TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Diplomarbeit

zum Thema

Visualization of Indicators in R with Application to EU-SILC

ausgeführt am
Institut für Statistik und Wahrscheinlichkeitstheorie
der Technischen Universität Wien

unter der Anleitung von
Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser
und
Univ.-Ass. Dipl.-Ing. Dr.techn. Matthias Templ
als
verantwortlich mitwirkendem Universitätsassistenten

durch

Stefan Zechner
Matrikelnr.: 0325638
Hasnerstrasse 99/41, 1160 Wien

Wien, am 27. Oktober

Stefan Zechner

Acknowledgments

First of all I would like to thank my supervisors Matthias Templ and Peter Filzmoser, who gave me a better understanding of statistics in different courses. Especially their teachings have been very inspiring where statistics was shown in practice as well. In our weekly meetings I had learned a lot about statistics, and whenever I had questions they took their time to answer them. In my points of view the fascinating topic in statistics is the visualization of certain statistical information. Applying suitable methods may get everyone a deeper knowledge of the data.

Furthermore I would like to thank my colleagues and friends Andreas Alfons, Angelika Meraner, Stefan Kraft and Josef Holzer, who helped me a lot with the statistical environment **R**, and for the friendly working environment in our shared office. Andreas had the hardest task of them, answering countless of questions about **R** for a very long period, and he had a helpful answer every time.

Last but not least I want to thank my family, who supported me during my whole study and Melanie Wurzer for her patience during exam times. The diploma thesis was partly funded by the European Union (represented by the European Commission) within the 7^{th} Framework Programme for Research (AMELI Project).

Stefan Zechner

Abstract

This diploma thesis covers a part of the workpackage 8 of the *Advanced Methodology for European Laeken Indicators* (AMELI) project, in which the Department of Statistics and Probability Theory of the Vienna University of Technology is involved. The main points are visualizing indicators in order to support policy decisions, regional indicators in maps, as well as visualization for a better understanding by the end user of the indicator values, including their quality.

The main task of the thesis is the development of new methods to visualize indicators, and the implementation of these methods in **R**. Several questions arise: which indicators are user friendly (easy understandable), when is the estimation value of an indicator good, bad, worsening or getting better? The thesis tries to answer these questions as well. The visualization of confidence intervals is also discussed in this thesis. The *sparkTable* package, which has been developed during this thesis, allows the presentation of graphical tables including *sparklines*.

Visualization of indicators in maps is another large part of this thesis. Maps should be presented in an interactive manner and their handling should be made user friendly. The colors of the maps should additionally be appealing. Detailed maps are needed, if possible a map of Europe, and the possibility to 'zoom' into the map to the level of NUTS3.

Furthermore different methods will be discussed how to calculate and display correlations between countries and their indicators. Alternative visualization methods like 'weather indicators' are also part of the thesis.

Within the AMELI project the methods are tested using real-world data from Europe (EU-SILC). For the visualization methods mainly the Gini coefficient is used, but the methods can also be applied to any other indicator.

Contents

Chapter 1

Introduction

1.1 Indicators

It is not easy to provide a formal definition of indicators. Indicators have a wide spread usage for different groups of people, may it be a statistician, politician or 'normal' inhabitants. As the name 'indicator' expresses, its sense is to indicate relevant information out of a complex data set, which can appear in ecology (e.g. ozone), economics (e.g. inflation), sociology (e.g. poverty rate) or any other field of science.

So an indicator can tell us something about the current state, the direction we are heading to, or how far away we are from a defined target. It is a small peace of information, which reflects the status of large data sets. Indicators are not made to tell us everything, but just enough to be a decision guidance.

Indicators are changing over time, so they should be visualized by numbers or graphics, which are made understandable for most people, who do not have the background of scientists. In other words, people should get a sense of the big picture by looking on a small part of the information.

The *International Institute for Sustainable Development* (IISD)[1] defines an indicator as follows:

> *An indicator quantifies and simplifies phenomena and helps us understand complex realities. Indicators are aggregates of raw and processed data but they can be further aggregated to form complex indices.*

Another definition is given by the *Organisation for Economic Co-operation and*

[1] www.iisd.org

Development (OECD)[2]

> *An indicator is a parameter, or a value derived from parameters, which points to, provides information about or describes the state of a phenomenon/environment/area, with a significance extending beyond that directly associated with a parameter value.*

In most countries institutions such as the national statistical offices (e.g. Statistics Austria), different ministries or others are responsible for collecting data and publishing this information. Nowadays institutions of different countries are working together in international programs. These bilateral relationships make it easier to develop new methods and to standardize them.

1.2 Selecting indicators

Indicators can be used on global, international or national levels. At the international level different organisations collect data from various countries, like the OECD or the European Union. At national levels the statistical offices are doing the same within their country.

Some points which should be considered if choosing an indicator (Horsch 1997) are now explained:

- Indicators should provide easy understandable information about the condition and result of the measured data. For example, if the wanted result is a reduction of poverty, achievement would be best measured by an outcome indicator, such as the at-risk-of-poverty rate.

- Over time the definition of the indicator should stay the same, and so should the process of collecting data. Decision makers have to be certain that the data they are looking at did not change over time. The data has to be collected frequently enough to be useful, most data is available on an annual base. Can new methods be developed to collect those data more cost effective?

- Is this indicator important to most people? Publicized indicators should have a high credibility. They are providing information which is easy to understand

[2] `www.oecd.org`

and accepted by decision makers. Highly technical indicators which require numerous explanations may not be useful for most people, but only for those who are working closer in the program for which the indicator is needed.

- Wrong or poorly measured indicators can lead decision makers down to a wrong path, faster than they would be without indicators. Indicators only represent a part of the picture, that should be kept in mind.
- The data quality is very important. Without good data the calculation or estimation of indicators may be biased, which could lead to wrong decisions.

1.3 Visualizing indicators

After indicators are calculated, the final step is the presentation of the results to the general public, in various media like newspapers, or to decision makers.

The visualization of the indicators should be clear to understand for everyone, awake interest and be graphically appealing. Clear to understand can also be expressed as simplicity, technical symbols or too many details will most likely confuse the audience. In which direction the chosen indicator is heading should also be clear, is it getting better or worse?

This is exactly what this diploma thesis is about. But why do we want to visualize indicators?

The most important point is the comparability. The globalization has lead to a standardization of most indicators. This makes indicators comparable with each other in an international context.

The Internet supplies us with enormous information. Especially online lexicons like Wikipedia[3] enjoy great popularity. The following examples of visualization methods are taken from this online encyclopaedia:

One of the most used indicators in the general public is the unemployment rate. Figure 1.1 shows a map of the European Union. The colors of each country represent the unemployment rate. That way the countries can be easily compared with each other.

A very contemporary issue during the last years was the greenhouse gas emission. One of the aims of the Kyoto Protocol is the reduction of fuel use. Someone could

[3] http://en.wikipedia.org

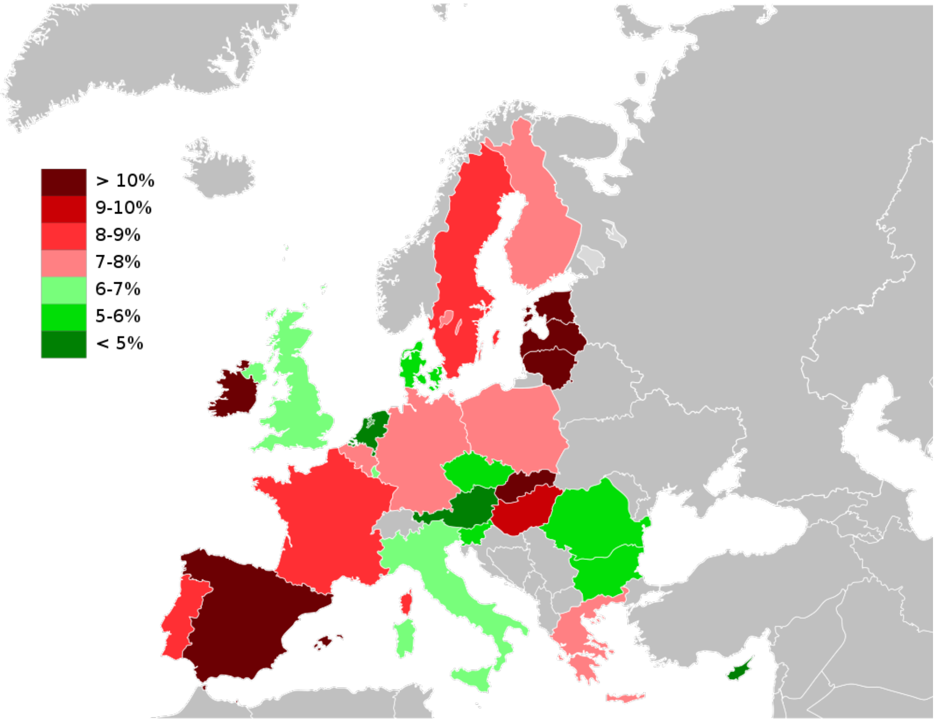


Figure 1.1: Unemployment rate in the European Union in March 2009.

think that new improvements in efficiency would help that goal, however it is the other way around. This effect is called the Jevons paradox. In short this paradox says: Improved efficiency lowers cost, which in turn increases demand.

Figure 1.2 presents the world market energy use by fuel by the Energy Information Administration (EIA). We can see that the EIA expects a steady increase of fuel use over the next years.

There are many other examples of visualization methods, and a few of them will be mentioned in the following chapters.

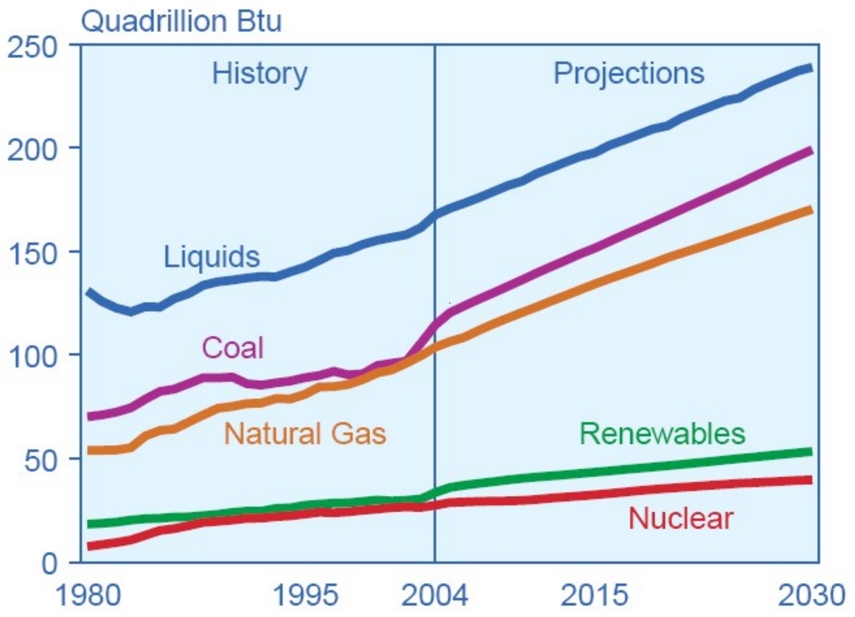


Figure 1.2: World marketed energy use by fuel.

1.4 Outline

All visualization methods mentioned in the following chapters have been implemented in the statistical computing environment **R** (see R Development Core Team 2010). Some helpful information on **R** can be found in Chambers (2008) and an introduction in data visualization can be found in Tufte (2006).

The diploma thesis is structured as follows:

Chapter 2: Laeken indicators

This chapter provides an overview on Laeken indicators which currently consist of 25 statistical indicators. It also includes the mathematical background of these indicators and which indicators are used for further study in this diploma thesis.

Chapter 3: Evaluation of indicators

The evaluation of indicators is an important topic. What defines an indicator as good, bad or neutral? How this could be calculated is mentioned in this chapter. Another section will approach the topic of graphical tables, and the development of a software package for that matter. This chapter also includes an alternative way of visualization, the *weather indicator*. It also mentions a way to display the often missing information of confidence intervals in graphics.

Chapter 4: Interactive maps
Visualization of national or international indicators is often done in maps. Countries or regions are easily comparable if they differ in colors or have additional information plotted in these maps. We will also discuss the need of projections in maps. This chapter also contains a short introduction in colors and their usage, and will describe why Figure 1.1 is a bad example for using colors. Furthermore different methods will be discussed how to calculate and display correlations between countries and their indicators.

Chapter 5: Summary and Conclusion
We are making a recapitulation of the diploma thesis in this chapter, including a brief outline of future work concerning the visualization of indicators.

Appendix A: R-Code
This chapter contains several for this diploma thesis developed functions in **R**, which have been used to visualize indicators in the previous chapters.

Chapter 2

The Laeken Indicators

The *Lisbon Strategy* is a development plan for the European Union (EU). Its goal is to turn the EU into the most competitive and dynamic economy in the world by 2010. The strategy was developed in the year 2000 for a period of ten years in Lisbon by the European Council. For this reason, the European Commission formed some social indicators for the measurement of poverty, inequality and social cohesion. This set of indicators is called Laeken indicators. More details about Laeken indicators can be found below.

The project *Advanced Methodology for European Laeken Indicators* (AMELI) started in April 2008. The AMELI project is a joint project of statistical offices and some universities in Germany, Switzerland, Finland and Austria. The project is split into more than 10 parts, called work packages. The main goal of the project is to increase the quality of the methodology for the estimation of the Laeken indicators, the development of new robust methods as well as new imputation methods which should increase the precision of the indicators. Another part of the AMELI project and the topic of this diploma thesis is the *visualization of indicators*, which is also an important issue. The analysis of the results of the developed methods and graphics should help the policy makers in their decisions. The project will finish in 2011.

The AMELI project is funded by the European Commission within the 7th Framework Programme for Research of the European Union.

The European Union has agreed a core set of poverty and social exclusion indicators which are regularly produced for every EU country on a comparable basis, called the 'Laeken indicators' (EUROSTAT 2003). For the estimation of the Laeken indicators the *European Survey on Income and Living Conditions* (EU-SILC) data set is used.

The EU-SILC data set is very important for the European social statistics and is taken as a basis for the estimation of the Laeken indicators. This data set includes personal and household data and other topics, it is collected every year in all 27 member states of the European Union (EU) as well as in Norway, Iceland, Turkey and Switzerland. More information about the EU-SILC data can be found in EUR (2007). There is a direct impact between the quality of the data and the quality of the Laeken indicators. Extreme outliers in the income components can have a huge influence on the estimation of the Laeken indicators.

The Laeken indicators are monitoring the multidimensional phenomena of poverty and social exclusion within a given area. In December 2000 at the Nice European council, Heads of State and Government confirmed their decision to 'fight against' poverty and social exclusion. The EU has named 2010 its year for fighting poverty and social exclusion.

At the time of definition the Laeken indicators consisted of 18 statistical indicators, divided into two parts, primary indicators (indicator 1 to 10) and secondary indicators (indicator 11 to 18). During the last years some definitions changed, some indicators were added and some others were dropped. The Laeken indicators contain now 11 primary indicators, 3 secondary indicators (with various breakdowns) and 11 context indicators. In practice, the primary list has been refocused to contain only the most important indicators that describe the various dimensions of poverty and social exclusion. A few indicators that were in the primary list became secondary indicators. More details can be found in EU-SILC (2009).

The Laeken indicators are presented in Table 2.1. Those which are marked by a '*' are using the EU-SILC data as a source. The other indicators use other sources like the EU Labour Force Survey (LFS) for the calculation. Some of the indicators in Table 2.1 are available from 2009 on, and some are still under development. The most relevant indicators which can be calculated using the EU-SILC data will be described below including the mathematical background.

Table 2.1: The Laeken indicators

Indicator SI-P1*	At-risk-of-poverty rate At-risk-of-poverty threshold
Indicator SI-P2*	At-persistent-risk-of-poverty rate
Indicator SI-P3*	Relative median at-risk-of-poverty gap
Indicator SI-P4	Long term unemployment rate
Indicator SI-P5	Population living in jobless households
Indicator SI-P6	Early school leavers not in education or training
Indicator SI-P7	Population living in jobless households
Indicator SI-P8*	Material deprivation rate
Indicator SI-P9*	Housing
Indicator SI-P10*	Unmet need for care by income quintiles - Inequalities in access to health care
Indicator SI-P11*	Child well-being
Indicator SI-S1*	At-risk-of-poverty rate
Indicator SI-S1a*	At-risk-of-poverty rate by household type
Indicator SI-S1b*	At-risk-of-poverty rate by work intensity of the households
Indicator SI-S1c*	At-risk-of-poverty rate by most frequent activity status
Indicator SI-S1d*	At-risk-of-poverty rate by accommodation tenure status
Indicator SI-S1e*	Dispersion around the at-risk-of-poverty threshold
Indicator SI-S2	Persons with low educational attainment
Indicator SI-S3	Low reading literacy performance of pupils
Indicator SI-S4*	Intensity of material deprivation
Indicator SI-C1*	S80/S20 income quintile share ratio
Indicator SI-C2*	Gini coefficient
Indicator SI-C3	Regional cohesion: dispersion in regional employment rates
Indicator SI-C4*	Healthy Life expectance and Life expectance at birth, and at age 65
Indicator SI-C5*	At-risk-of-poverty rate anchored at a fixed moment in time
Indicator SI-C6*	At-risk-of-poverty rate before social transfer
Indicator SI-C7	Jobless households by main household types
Indicator SI-C8*	In-work at-risk-of-poverty rate
Indicator SI-C9	Making work pay indicators
Indicator SI-C10	Net income of social assistance recipients as a % of the at-risk-of-poverty threshold for 3 jobless household types
Indicator SI-C11*	Self-perceived limitations in daily activities

2.1 'At-risk-of-poverty rate' by age and gender

2.1.1 Calculation of the 'equivalized household size'

The equivalized household size (EQ_SS) is a virtually weighting of each member of a household. This weighting works according to an OECD scale. This scale gives a weight of 1.0 to the first adult, 0.5 to other persons aged 14 or over who are living in the household (HM_{14+}) and 0.3 to each child aged less than 14 (HM_{13-}). Consider a household consisting of 2 adults and 2 children, in this case the equivalized household size is $2.1 = (1 + 0.5 + 0.3 + 0.3)$.

$$EQ\_SS = 1 + 0.5 \cdot (HM_{14+} - 1) + 0.3 \cdot HM_{13-} \quad . \tag{2.1}$$

2.1.2 Calculation of equivalized disposable income

For each person, the equivalized disposable income (EQ_INC) is defined as the total household disposable income divided by the equivalized household size. The total disposable household income (TDHI) is the sum for all household members of personal income components plus income components at household level,

$$EQ\_INC = \frac{TDHI}{EQ\_SS} \quad . \tag{2.2}$$

The equivalized disposable income is equal for each person who is living in the same household.

2.1.3 Calculation of national median equivalized disposable income

Persons have to be sorted by their 'equivalized disposable income', from lowest to highest value. Then the median of 'EQ_INC' is calculated as follows:

$$med(EQ\_INC) = \begin{cases} \frac{1}{2}(EQ\_INC_j + EQ\_INC_{j+1}) & \text{, if } \sum\limits_{i=1}^{j} u_i = \frac{1}{2}U \\ EQ\_INC_{j+1} & \text{, if } \sum\limits_{i=1}^{j} u_i < \frac{1}{2}U < \sum\limits_{i=1}^{j+1} u_i \quad , \end{cases}$$

where EQ_INC_j is the equivalized disposable income of person j, and $U = \sum\limits_{i=1}^{n} u_i$ is the estimated population size (u_i are the sample weights) and n is the number of household members in the sample.

2.1.4 Calculation of the at-risk-of-poverty threshold

With the calculated median of the equivalized income we can now calculate the at-risk-of-poverty threshold (ARPT)

$$ARPT = 60\% \cdot med(EQ\_INC) \tag{2.3}$$

2.1.5 Calculation of the 'at-risk-of-poverty rate'

In order to calculate the at-risk-of-poverty rate (ARPR) we just need an indicator function which flags households with an income lower that then ARPT.

$$y_i = \begin{cases} 1 & \text{, if} \quad EQ\_INC_i < ARPT \quad i = 1, \dots, n \\ 0 & \text{, otherwise} \quad . \end{cases} \tag{2.4}$$

Finally the estimated 'at-risk-of-poverty-rate' is given by

$$ARPR = \frac{\sum\limits_{i=1}^{n} u_i y_i}{\sum\limits_{i=1}^{n} u_i} * 100 \quad . \tag{2.5}$$

2.2 'At-risk-of-poverty rate' for domains

Usually, the ARPR is estimated for certain domains, such as different categories of:

- Age and gender
- Household type
- Work intensity of the household and by gender and selected age group
- Most frequent activity status and by gender and selected age group
- Accommodation tenure status and by gender and selected age group

ARPR with breakdown by age and gender

The age is divided into 3 groups (0 - 17 years, 18 - 64 years and more than 65 years), and by gender. In the 0 - 17 age group there is no breakdown by gender, since there is no difference expected in the poverty risk of children. The ARPR is then estimated for those domains. Other breakdowns for domains will not be mentioned here, details can be found at EU-SILC (2009).

2.3 Inequality of income distribution: S80/S20 quintile share ratio

The S80/S20 quintile share ratio is the ratio of the total income received by the 20% of the country's population with the highest income (top quintile) to that received by the 20% of the country's population with the lowest income (lowest quintile) while 'income' is the equivalized disposable income. Persons are sorted according to their equivalized total net income (sorting order: lowest to highest value). The 20% of persons at the lower end of the distribution are defined as 'poorest' (first quintile). The 20 % of persons at the upper end of the distribution are defined as 'richest' (fifth quintile).

2.3.1 Calculation of the S80/S20 quintile share ratio

In theory the net equivalized income available to a quintile is the sum of the equivalized income of the individuals belonging to the quintile. In practice, the mean equivalized income of the quintile is used instead. S80/S20 is the quotient of the equivalized income available to the 5th quintile (richest) over the 1st quintile (poorest), it is defined as

$$QSR = \frac{E(x|x > Q_{EQ\_INC}(0.8))P(x > Q_{EQ\_INC}(0.8))}{E(x|x < Q_{EQ\_INC}(0.2))P(x < Q_{EQ\_INC}(0.2))} \quad , \tag{2.6}$$

The estimation of the quintile share ratio can be realized as follows:

$$\widehat{QSR} = \frac{\sum\limits_{i \in S} u_i \cdot EQ\_INC_i \cdot \mathbb{1}\{EQ\_INC_i > \hat{Q}_{EQ\_INC}(0.8)\}}{\sum\limits_{i \in S} u_i \cdot EQ\_INC_i \cdot \mathbb{1}\{EQ\_INC_i \leq \hat{Q}_{EQ\_INC}(0.2)\}} \quad , \tag{2.7}$$

where $\mathbb{1}$ is the indicator function and $\hat{Q}_{EQ\_INC}(0.2)$ and $\hat{Q}_{EQ\_INC}(0.8)$ are the estimated 0.2 and 0.8 quantiles of the equivalized disposable income and S is the set of all observations in the considered sample.

The indicator function $\mathbb{1}\{EQ\_INC_j \leq EQ\_INC_i\}$ can be written as

$$\mathbb{1}\{EQ\_INC_j \leq EQ\_INC_i\} = \begin{cases} 1 & \text{, if} \quad EQ\_INC_j \leq EQ\_INC_i \\ 0 & \text{, otherwise} \quad . \end{cases} \tag{2.8}$$

2.4 Inequality of income distribution: Gini coefficient

The Gini coefficient (developed by the Italian statistician Corrado Gini) is defined as relationship of cumulative shares of the population arranged according to the level of income, to the cumulative share of the equivalized total net income received by them.

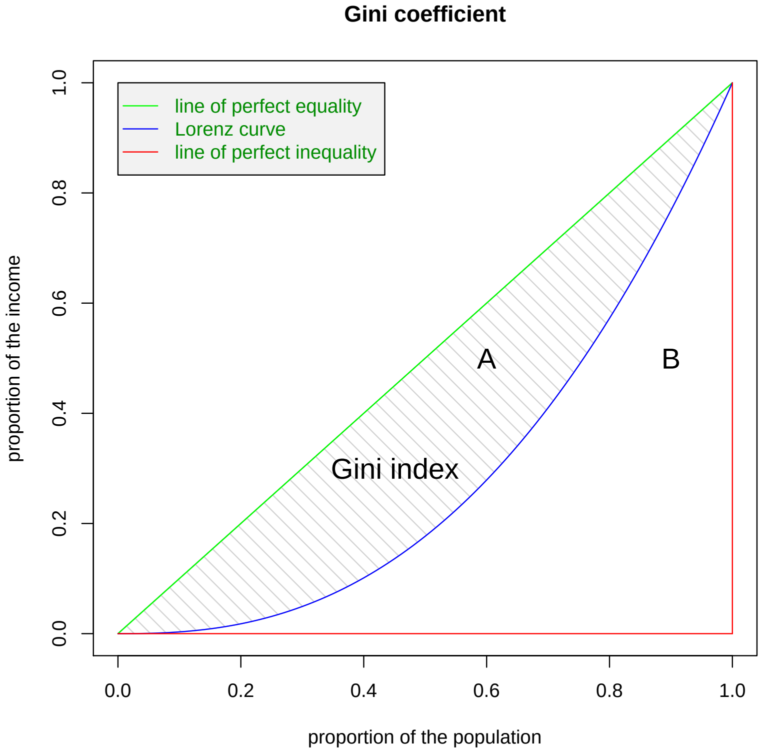


Figure 2.1: The Lorenz curve.

The easiest way to explain the Gini coefficient is with the help of the Lorenz curve. The Lorenz curve represents an income distribution. It can be seen as a way to visualize social inequality. Figure 2.1 shows one possible Lorenz curve. In that case around 40% of the population have only around 10% of the total net income. This can be seen if we follow the Lorenz curve to 0.4 on the x-axis and take a look at the y-value. If the income would be distributed equal among the people, the proportion of the population and the proportion of the income would be the same. The Gini coefficient is the ratio of the area that lies between the line of equality and the Lorenz curve (marked 'A' in the diagram) over the total area under the line of equality (marked 'A' and 'B' in the diagram),

$$G = \frac{A}{A+B} \tag{2.9}$$

For the estimation of the Gini coefficient the equivalized disposable income (EQ_INC) has to be sorted in increasing order.

The estimated Gini coefficient can be expressed as (see Hulliger and Münnich 2006)

$$\hat{G} \;=\; \frac{1}{\hat{\tau}} \cdot \sum_{i \in S} u_i \cdot \Big(2 \cdot \frac{1}{\hat{N}} \sum_{j \in S} u_j \cdot \mathbb{1}\{EQ\_INC_j \leq EQ\_INC_i\} - 1\Big) \cdot EQ\_INC_i$$

where $\hat{\tau}$ denotes the Horwitz-Thompson (HT) estimate for EQ_INC which is given by

$$\hat{\tau} = \frac{1}{\hat{N}} \sum_{i \in S} u_i \cdot EQ\_INC_i \tag{2.10}$$

with $\hat{N} = \sum\limits_{i \in S} u_i$ the number of individuals.

The visualization will focus on a few indicators like the Gini coefficient, but can also be used for any other kind of indicator.

Chapter 3

Evaluation of Indicators

Currently there are several different graphical systems in use for the evaluation of indicators like different smileys (from happy to sad) or colored tendency symbols. Different ways exist to evaluate an indicator. The four most common ways are:

- current status
- absolute change over time
- change over time relative to a past value
- change over time relative to a target value

Not every indicator aims at a target value. In case a target value is missing it has to be set by a political authority. As in every other statistical conclusion the variance and the accuracy of the indicator is an important issue.

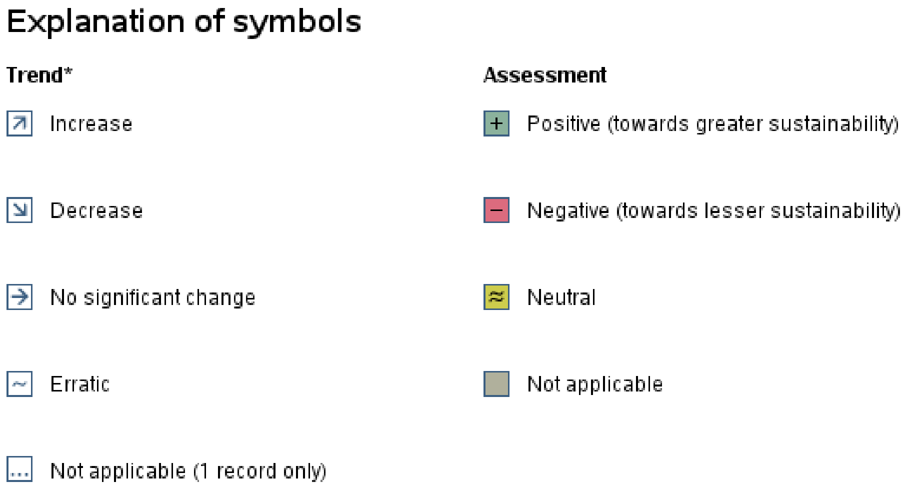
In Agency (1999) a semaphore code (traffic light colors) has been defined for indicators as a guideline by a scientific advisory group, consisting of 2300 European environment experts from all over the EU. The quality of the indicators can be divided into thee categories:

Relevancy refers to the closeness of the indicator to the environmental problem, also the chosen methodology and the relevancy of the breakdown published.

Overall accuracy represents issues such as comparability of the data, reliability of data sources and the used methodology, and if the result can be validated (e.g. by sensitivity analysis).

Comparability over time takes a look at the completeness of time series and the consistency of the used methodology over time.

The Federal Statistical Office, the Federal Office for the Environment and the Swiss Federal Office for Spatial Development developed the indicator system *MONET* (German abbreviation for Monitoring Sustainable Development). This indicator system is monitoring the sustainable development, and its aims are to provide information about the current situation and trends in social, economic and environmental aspects of sustainable development. It also allows the comparison to other countries and is designed as an information source for the public, politicians and the Swiss Federal Government.



Explanation of symbols

Trend*	Assessment
↗ Increase	+ Positive (towards greater sustainability)
↘ Decrease	− Negative (towards lesser sustainability)
→ No significant change	≈ Neutral
~ Erratic	▨ Not applicable
… Not applicable (1 record only)	

Figure 3.1: The *MONET* indicator system.

Figure 3.1 presents the *MONET* indicator system. The sustainable development can be directly seen by the trend and the assessment. An increasing trend of the at-risk-of-poverty rate for example will lead to a negative assessment.

In this chapter it will be discussed how to evaluate changes over time.

3.1 Example: Greenhouse gas

In the European Environment Agency (2005) another example is given for the evaluation of indicators. Every indicator is shown with an inaccuracy rate. The following example represents the measurement of greenhouse gases (short: GHG). The absolute values have an inaccuracy rate of ±20%, while the change to the past can be calculated with a variance of ±8%.

This indicator illustrates current trends in anthropogenic GHG emissions in relation to the EU and Member State targets. Emissions are presented by type of gas and weighted by their global warming potentials. There is a growing evidence that emissions of greenhouse gases are causing the global surface air temperatures to increase, resulting in a climate change. The potential consequences are fatal, rising sea levels which will lead to an increased frequency and intensity of floods, global warming will also lead to more droughts. Efforts to reduce or limit the effects of climate change are focused on limiting the emissions of all greenhouse gases covered by the Kyoto Protocol. This indicator supports the Commission's annual evaluation of progress in reducing emissions in the EU and the individual Member States to achieve the Kyoto Protocol targets.

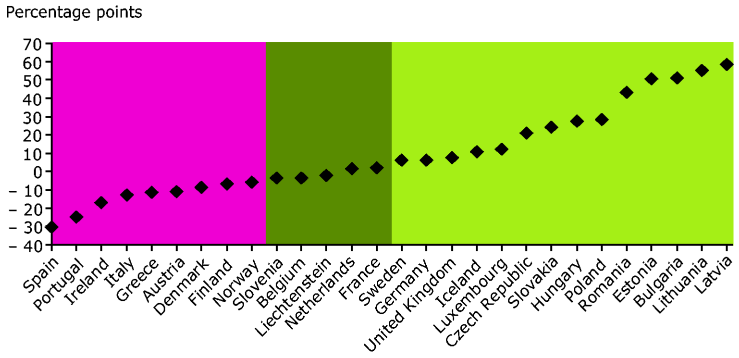


Figure 3.2: Emissions of ozone precursors, distance to NECD targets.

Figure 3.2 shows the information of GHG emissions of European countries. Those which are on the good side of the target-path are in the light-green section, countries with the indicator within a ±5% interval are dark-green and countries which are clearly on the bad side of the target-path are pink. As mentioned in Section 4.1, colors play an important role for visualization. This is one of the bad examples of how to choose colors because the colors are gaudy and the saturation differs too much between the colors.

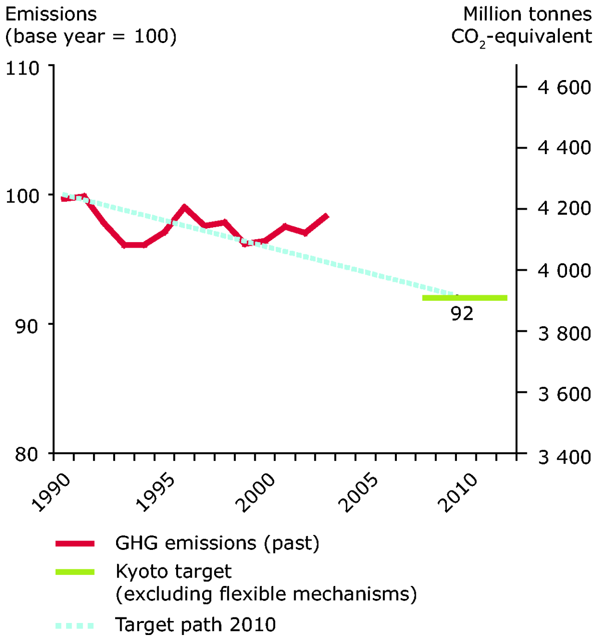


Figure 3.3: Development of EU-15 greenhouse gas emissions.

Figure 3.3[1] shows an example for the visualization of an indicator when a target value is given. From a base value of 100 in the year 1990 the target path aims at the Kyoto target of 92 in the year 2010. Or in other words the CO_2 emissions should decrease by 8% within 20 years.

3.2 Evaluation Plots

To present indicators with symbols two different problems have to be faced. The evaluation and the visualization of the indicator. For graphical solutions there are already different systems in use, like the one in the greenhouse gas example mentioned above.

The type of visualization depends also on the context. Are there different indicators in one concept, is there one indicator for different regions, or are we looking at one (or more) indicators over time?

[1] source: `http://www.eea.europa.eu/`

The main idea behind this evaluation method is based on a paper by Hulliger and Lussmann (2008). This diploma thesis generalizes and expands their approach.

The assumption is that there are three different evaluations for indicators:

- good
- neutral
- bad

The middle category is maybe better described as critical as it may drop in the bad zone. The definition of the parameter is a political process. The way from parameters to the evaluation should be transparent and should support political discussions.

3.2.1 Notation

An indicator is defined as a real-valued time series x_t. The indicator has been monitored from time t_1 until the current time t_N, or written in mathematical notation $t_1 \leq t \leq t_N$. The time t_S ($t_S \leq t_1$) is defined as a reference point or a start point in the past, while t_T ($t_T \geq t_N$) refers to a time in the future. Furthermore we assume without loss of generality to monitor the indicator once per year, so t usually represents the different years.

Reference values are denoted by x^*. The value at a given start time t_S is defined as x^*_S, while a target value at the time t_T is written as x^*_T.

3.2.2 Evaluation

As defined above the evaluation is working with three categories: good, neutral and bad. The evaluation of an indicator is defined as $Eval(x_t)$ with the following values

$$Eval(x_t) = \begin{cases} 1 & \text{good} \\ 0 & \text{neutral} \\ -1 & \text{bad} \end{cases} \tag{3.1}$$

Later on, the evaluation of an indicator will also depend on other quantities than on x_t. An increasing indicator can either be good or bad. An increasing indicator is assumed to be evaluated as negative, because for most Laeken indicators like the Gini coefficient, at-risk-of-poverty rate or the Quintile Share Ratio perform better if they have a low value.

Now different options of the evaluation of an indicator will be discussed, depending if the deviation from an initial value, a target value or from a given course is of interest.

3.2.3 Deviation from an initial value

The difference from the initial value x_S^* is defined as

$$\delta_t(S) = x_t - x_S^* \quad . \tag{3.2}$$

The initial value is either the mean value over a few years or can be defined by other criteria. The aim is an improvement of the indicator over time, and by the definition from above this refers to a decreasing value of the indicator.

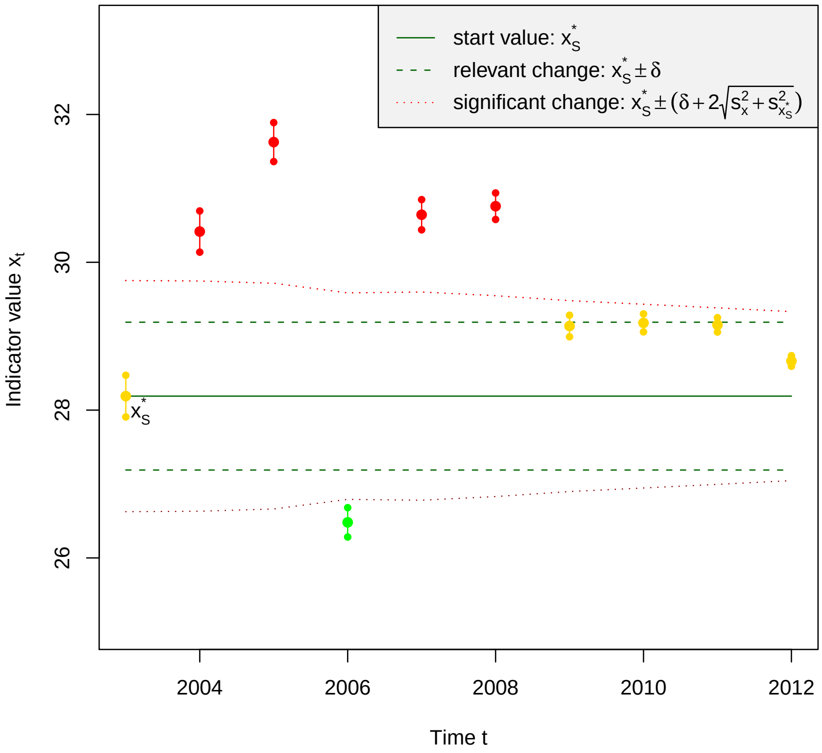


Figure 3.4: The evaluation plot for a given initial value.

For the evaluation of the indicator a reference value δ (the two dashed lines in Figure 3.5) is needed, which defines whether a change is relevant or not. This is one of the difficult decisions for the evaluation of an indicator, because it is mainly based on the knowledge of experts. However, the question arises what is meant by a relevant change? It might be possible to estimate s_x, the standard deviation of x_t. Furthermore it might also be possible to estimate $s_{x_S^*}$, the standard deviation of x_S^*. However, if the standard deviation $s_{x_S^*}$ cannot be estimated adequately, experts have to determine it. If s_x cannot be estimated, s_x has to be set at zero ($s_x = 0$) to prevent wrong evaluations. Note, it is possible that a relation between δ and $s_{x_S^*}$ exists. Experts would not choose a δ smaller than $s_{x_S^*}$ for example. Now 'relevant changes' have to be defined. It does not make any sense to name changes smaller than $s_{x_S^*}$ or even better, smaller than $2 \cdot s_{x_S^*}$ as relevant. This follows from the fact that if a data distribution is approximately normal then approximately 95% of the data values are within mean plus/minus two standard deviations of the mean.

The relevant deviation δ can also be seen as the real variability of the phenomena which got observed. The standard deviation s_δ could be seen as a measurement variability, which will be added to the real variability and the measurement variability may change over time. In Figure 3.5 the measurement variability is visualized via vertical lines at the indicator values. However, in practice the two variabilities are often hard or even impossible to differ.

The variance of $\delta_t(S)$ is defined as $s_{\delta(S)}^2 = s_x^2 + s_{x_S^*}^2$ if the covariance between x_t and x_S^* is zero, so $cov(x_t, x_S^*) = 0$. The 95% confidence interval for the deviation $\delta_t(S)$ is then approximately $[\delta_t(S) - 2 \cdot s_{\delta(S)}, \delta_t(S) + 2 \cdot s_{\delta(S)}]$. The dotted lines in Figure 3.5 illustrate this confidence interval.

If zero is not included in this confidence interval the deviation differs relevant from zero. If the confidence interval does not overlap with any part of the interval $[-\delta, \delta]$, then the deviation differs not only relevant from zero, but it is also significant.

In the case that high values of the indicator are bad, following valuation can be used:

$$Eval(x_t, x_S^*, \delta, s_{\delta(S)}) = \begin{cases} -1 & x_t > x_S^* + \delta + 2 \cdot s_{\delta(S)} \\ 1 & x_t < x_S^* - \delta - 2 \cdot s_{\delta(S)} \\ 0 & \text{else} \end{cases} \tag{3.3}$$

3.2.4 Deviation from a target value

A target value is mostly defined by politicians, but it could also be a natural value. Now the difference from the actual value to the target value can be evaluated. The target value at time t_T is defined as x_T^*, and the difference between that target value and the time series is defined as

$$\delta_t(T) = x_t - x_T^* \tag{3.4}$$

As mentioned in the previous section a reference value δ is needed which defines the relevant distance. $\delta_t(T)$ has again variance greater zero, and if x_T^* is a constant value without a zero variance then $s^2_{\delta(T)} = s^2_x$. The evaluation is the same as in the section before:

$$Eval(x_t, x_T^*, \delta, s_{\delta(T)}) = \begin{cases} -1 & x_t > x_T^* + \delta + 2 \cdot s_{\delta(T)} \\ 1 & x_t < x_T^* - \delta - 2 \cdot s_{\delta(T)} \\ 0 & \text{else} \end{cases} \tag{3.5}$$

So an indicator is defined as 'bad', if its value is above a threshold, where a positive deviance from the target value (in the 'bad' direction of the indicator) is significant.

3.2.5 Deviation from a given course

In this section starting value x_S^* and a given target value x_T^* is given, together with a period of time from t_S to t_T in which the target value should be reached. The interesting part is the distance between the indicator and the course which leads from the starting value to the target value.

The linear course from x_S^* to x_T^* for $t_S \leq t \leq t_T$ is

$$x_t^* = x_S^* + \frac{x_T^* - x_S^*}{t_T - t_S} \cdot (t - t_S) \tag{3.6}$$

Other courses are also possible, like asymptotic ones (e.g. halving of the distance every year), but here the focus is on the linear one (one possible non-linear course will be discussed later). The quotient

$$\beta = \frac{x_T^* - x_S^*}{t_T - t_S} \tag{3.7}$$

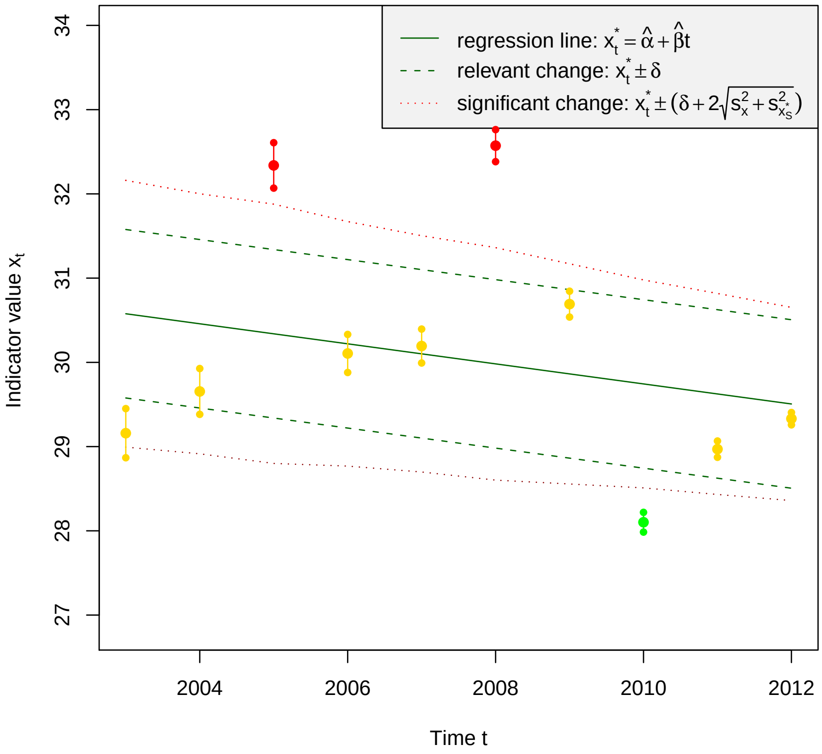


Figure 3.5: The evaluation plot for a given course.

is the slope of the line and describes the desired change in a time period. For every point in time a target value is given as a reference point.

The evaluation is calculated as follows:

$$Eval(x_t, x_S^*, \delta, \beta, s_\delta^2) = \begin{cases} -1 & x_t > x_S^* + \delta + \beta(t - t_S) + 2 \cdot s_\delta^2 \\ 1 & x_t < x_S^* - \delta + \beta(t - t_S) - 2 \cdot s_\delta^2 \\ 0 & \text{else} \end{cases} \tag{3.8}$$

If and only if there is an significant deviation from the course, the evaluation can be considered as good or bad. In the middle band around the course the evaluation is neutral, as the evaluation is neither significantly worse than the course, nor is it on the good side of the course.

3.2.6 Calculating the linear trend of an indicator

Not for every indicator or every region a course or target value is given. In order to get at least a better picture of the indicator and its trend a linear regression got implemented. One easy possibility is to estimate the trend $\hat{\beta}$ and the intercept $\hat{\alpha}$ of an indicator by least squares estimation. Let $\hat{\beta}$ be the trend estimated by minimizing the sum of squared distances between the observed indicator x_t and the regression line x_t^*, this distances are called residuals.

$$x_t = \hat{\alpha} + t \cdot \hat{\beta} + \varepsilon, \qquad t_1 \leq t \leq t_N \tag{3.9}$$

where ε is random variable (errors) which accounts for the discrepancy between the actually observed responses x_t and the predicted outcomes $\hat{\alpha} + t \cdot \hat{\beta}$.

The evaluation is calculated as follows:

$$Eval(x_t, \delta, s_\delta^2) = \begin{cases} -1 & x_t > \hat{\alpha} + t \cdot \hat{\beta} + \delta + 2 \cdot s_\delta^2 \\ 1 & x_t < \hat{\alpha} + t \cdot \hat{\beta} - \delta - 2 \cdot s_\delta^2 \\ 0 & \text{else} \end{cases} \tag{3.10}$$

This approach has one great disadvantage, its non-robustness. Robustness is needed to face outliers. Outliers are observations that severely deviate from the linear data trend. The aim is to apply a regression method which is not sensitive if outliers occur. For this reason the MM-estimator got chosen for the robust regression. For more information on robust regression take a look at Yohai (1987).

Figure 3.6 presents the difference between robust and non-robust regression methods. Outliers can have a strong influence on the OLS regression line.

3.2.7 Calculating a non-linear trend of an indicator

For a nonlinear regression the LOESS model is used. LOESS (locally weighted scatterplot smoothing) was proposed by Cleveland (1979) as a model which uses locally weighted polynomial regression. For every point in the data a polynomial of low degree is fit to a subset of the data. The observations closer to the point have a higher weight than points further away. The default weight function for LOESS is usually a tri-cube weight function:

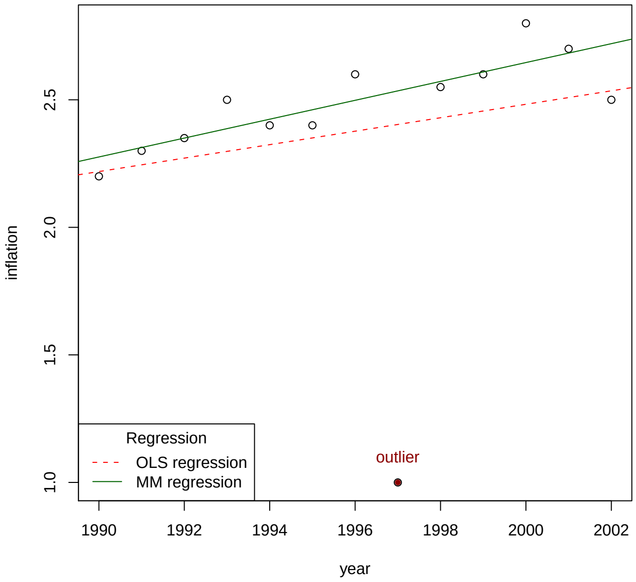


Figure 3.6: Comparison between robust and non-robust regression. The point in dark red is an outlier.

$$w(x) = \begin{cases} (1 - |x|^3)^3 & \text{for } |x| < 1 \\ 0 & \text{for } |x| \geq 1 \end{cases} .$$

For every point in the data x a polynomial of low degree is fit to a subset of the data. Points closer to the estimated response get more weight than points farther away. The degree of that polynomial as well as a smoothing parameter can be chosen by the user. How this works in practice can be seen in Figure 3.9.

3.3 Implementation in R

Based on a function written by Hulliger and Lussmann (2008, Appendix B) an extended function was written to evaluate and visualize indicators. The input for the function *indEval()* (Appendix A) can either be a time series or a vector with values and an additional time vector.

For the following visualizations the Gini coefficient of the equivalized income for Austria[2] in the years 1995-2008 got used. The indicator for 2002 is missing, for the visualization the value was imputed by the median of the time series, and for the years 2004-2008 the results from the *EU-SILC* data are used.

Table 3.1: Gini coefficient for Austria in the years 1995-2008

year	1995	1996	1997	1998	1999	2000	2001
Gini	27.0	26.0	25.0	24.0	26.0	24.0	24.0

year	2002	2003	2004	2005	2006	2007	2008
Gini	26.0	27.7	25.8	26.1	25.3	26.1	26.2

This is the *indEval()* function including all the parameters which can be set by the user:

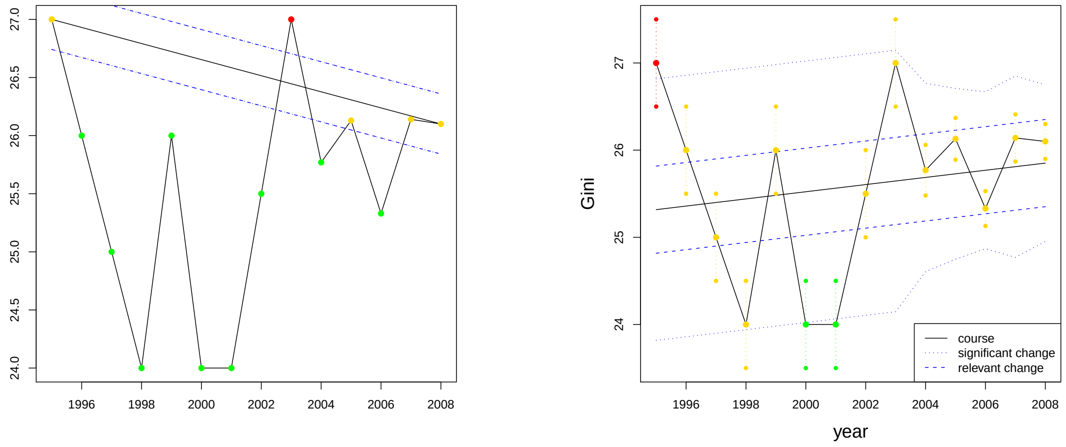
```
`indEval` <-
function(x, x_time=1:length(x), sderr=0, dr=0.01*median(x),
x0=x[1], t0=x_time[1], x1=x[length(x)], t1=x_time[length(x_time)],
betax, eval.labels=TRUE, good.ind="low", ind.size=1,
Legend=FALSE, placeLegend="topleft",
show.axes=TRUE, x.lab="Time", y.lab="Indicator",
parList=list(cex.lab=1.5, pch=19), sparkline=FALSE, regression="none",
plot.title=title(main = NULL, sub = NULL, xlab = NULL, ylab = NULL),
...)
```

The variable x is the data, which can either be a vector or a time series. The other default settings are sensitive to changes.

[2] http://appsso.eurostat.ec.europa.eu/nui/show.do?dataset=ilc_sic2&lang=en

3.3.1 Default visualization

With the above data given in Table 3.1 the default plot of the time series in Figure 3.7(a) would look only slightly different from a normal *plot* function. Additional information in the default plot of the *indEval* function is represented as the dotted lines and the different colors of the Gini coefficient. The dotted lines determine the region where the indicator will be called 'neutral', it is the region around the line reaching from the first value to the last value in our time series. By default this marks a 1% margin of the median of the indicator around the course. Outside this corridor an indicator is called 'bad' if it has a higher value, or 'good' if it has a lower value.



(a) Default visualization

(b) Visualization of the robust regression and standard deviation of the indicator

Figure 3.7: Two plots with the *indEval* function of the Gini coefficient of Austria in the years 1995-2008.

There are other indicators of interest which are called 'better' if they have a high value, or are staying within/outside a corridor. This can be changed via the option for the parameter `good.ind`:

- "high"
- "in.funnel"
- "out.funnel"

Figure 3.7(b) represents an advanced plot of the *indEval()* function, using a robust regression. There are three different settings available for the `regression` parameter:

- `"none"` - linear trend from the first to the last value
- `"LM"` - robust linear regression
- `"LOESS"` - non-linear regression

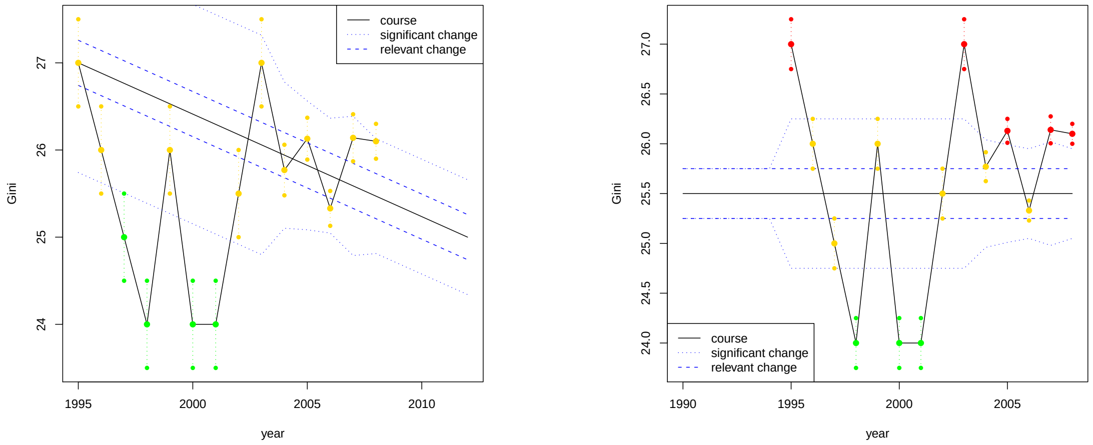
Additionally the measuring inaccuracy of the indicators are added to the plot, which have been calculated with bootstrap resampling (take a look at Section 3.6 for details) More graphical parameters for the *indEval()* function will be explained in the following subsections.

3.3.2 Visualization of a target value

This is probably the most used way to visualize indicators. Target values are used in almost all parts of science. There are several examples for this case:

Ecologists use target values for greenhouse gas, which should decrease over time and reach a small value in future, like shown in Figure 3.3. Economists use them for economic growth, the higher the better, on the other side a low value for the unemployment rate is aimed at.

The Gini coefficients in Table 3.1 should be as low as possible, since a low Gini coefficient indicates a more equal income distribution. An artificial long term target value x_T^* of the Gini coefficient of 25 in the year $t_T = 2012$ would lead to following visualization in Figure 3.8(a).



(a) Target value plot

(b) Initial value plot

Figure 3.8: Gini coefficient with a specified target value in the future (a) and a given initial value (b)

3.3.3 Visualization of a given initial value

If x_S^* is given the visualization of the development since that time can be made. Assume that politicians have defined a target for the Gini coefficient in the year 1990 of 25.5 for the following years. The plot shown in Figure 3.8(b) would present the outcome of such a setting. Note that in this plot the measuring inaccuracy (standard deviation of the estimated Gini coefficient) was halved. If the first known value is too far away in the past from the time series, the figure could look dispersed. In that case the visualization could lead to wrong conclusions.

3.3.4 Nonlinear Regression

Taking a look at the last option for the `regression` parameter for the *indEval()* function: `LOESS`. The LOESS function mentioned in Section 3.2.7 performs a nonlinear regression.

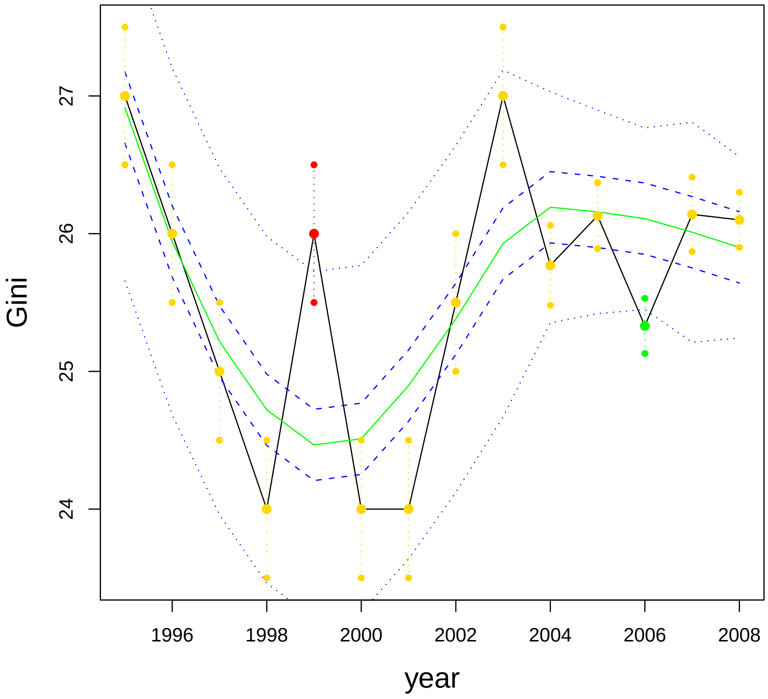


Figure 3.9: LOESS regression.

Figure 3.9 presents the outcome of the LOESS regression for our given data. With the non-linear regression it is possible to see some kind of cycle which could be connected to the business cycle for example. However, the focus is on the visualization the data, the interpretation of the results should be left over for experts.

3.4 Sparklines

The term sparkline was developed by Edward R. Tufte (Tufte 2006). A sparkline is a graphic of small size (which fits in a normal text) but containing a lot of information. The idea behind sparklines is the placement of the graphic in the text, for example, 'The Gini coefficient has increased the last years '.

Sparklines are especially useful in tables, if as much information as possible should be visible. The majority of sparklines consists either of polygons like in Table 3.2 or barplots. Last but not least, Alfons et al. (2009) proposed such tables to fit more information about indicators within a table.

Concerning sparklines, Tufte (2006) mentions another important issue, the *aspect ratio*. The *aspect ratio* is the ratio between width and length of a sparkline. Sparklines have a short dimension (the width is mostly specified by the font size of the text beside the sparklines) and a long dimension. In general, sparklines should have a median slope of 45°. This technique is called *banking to 45 degree* and got implemented by Cleveland (1993). For a given time series $x_t, t = 1, \ldots, T$, the median slope of the sparklines

$$median(|\alpha_t|) = median(|arctan(x_t - x_{t-1})|) = 45° \tag{3.11}$$

where α_t denotes the slope of the sparkline between two points x_t and x_{t-1} for $t = 1, \ldots, T$. The shape varies a lot when the aspect ratio gets changed. In practice the height of the sparkline is fixed, and the horizontal length of the sparkline is adjusted by considering the 45° rule. However, in tables this rule is not that easy to implement because the 45° rule will lead to sparklines of different length. Therefor, a fixed aspect ratio of 5:1 is often used especially for sparklines in financial applications.

Table 3.2 shows a practical example for the usage of sparklines in tables. For the purpose to have a user friendly way to produce tables like Table 3.2 the function *indTable()* (Appendix A) got developed. It is an **R** function which generates a LaTeX table.

One can easily generate sparklines out of a given data set. Table 3.2 shows an example with artificial data. The sparkline can either be a normal plot or one of the *indEval()*-function plots like in Figure 3.7(b) or 3.9. The user can select one or more of these columns: min, max, current, median, mean, trend, sparkline as well as the order of the columns.

An alternative way is to set the option `output="plot"` which generates a single

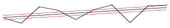
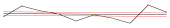
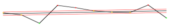
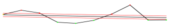
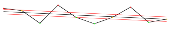
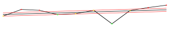
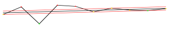
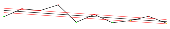
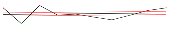
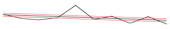
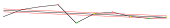
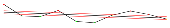
	sparkline	trend	min	max	current	median	mean
AT		0.27	25.56	34.28	34.22	32.00	30.90
BE		0.03	22.51	31.45	27.97	27.19	27.06
CY		-0.03	24.50	33.70	27.22	29.94	29.71
CZ		-0.12	25.67	35.22	27.42	27.42	29.24
DE		-0.11	26.88	32.78	28.37	28.89	29.44
DK		0.20	23.22	32.84	32.84	30.19	29.39
EE		0.01	23.71	32.97	31.27	31.27	30.15
ES		-0.16	27.17	33.34	27.17	29.36	29.49
FI		0.10	21.43	32.15	30.72	27.04	27.46
FR		-0.15	27.18	37.68	27.18	30.27	30.74
GR		-0.29	21.71	36.96	26.77	27.17	29.04
HU		-0.16	24.58	37.32	27.20	29.36	29.84

Table 3.2: Table of sparklines with the usage of the 'indTable' function.

PDF file. However, this function allows only one plot type which gets applied to the whole table. That this functions are not modifiable is a disadvantage, but the advantage is clearly the easy usability. A more complex approach is given in the next section, the *sparkTable* package for **R**.

3.5 The sparkTable package

sparkTable (Kowarik et al. 2010) is a package for **R**. It is an extension of the *sparkTable()*-function and includes various features. While the first two authors of the package did most of the programming part, my job was testing the **R** functions, finding possible errors and to propose modifications. The presentation of this package took place in Vienna at the *Workshop on Exploratory Data Analysis and Visualisation* in May 2010. Within the *indTable()* function the parameters are fixed for every column. The *sparkTable* package completely changes this approach by having various parameter settings for every single cell. Colors, width and length of a specific cell of a n-dimensional table can be changed in an easy manner, leaving all other cells unchanged. This gets especially handy when facing complex tables.

3.5.1 Aims of the package

The following points are considered:

- Providing quick access to additional insights by the use of *graphical tables.*
- Presentation of numerous data in a well-arranged way.
- Improving data density by using spark-graphs.
- Results should be easy to modify.
- Development of a tool to create graphical tables easily.
- Sparktables for multidimensional data.

Based on these ideas and aims the work started on the package, and some of the functions will be described below:

- spark_init(): This functions allows to set a list of properties depending on the type of the desired spark-graph for any given numerical input data vector.
- sparkTable_Config(): Using this function, meta-objects are created in order to generate multiple spark-graphs for multidimensional input data. This function produces the base frame of the table.

- setPara(): Gives the possibility to set or change (graphical) parameters for data objects generated with sparkTable_Config().
- print.sparkTable(): Output of graphical tables in *EPS* or *HTML*-format of given objects generated with sparkTable_Config().

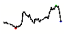
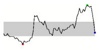
As mentioned above the output format can be chosen between *EPS* or *HTML*, and afterwards included on web pages or documentations. In the current version there are three types of plots to choose from, the time series plot, the bar plot and the boxplot.

Indexing of the meta object

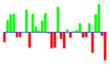
Tables often consist of higher dimensions. For example, k indicators for different countries got measured for T years, which denotes a *three way* table. This example leads to a table with $n \cdot k \cdot T$ cells, which can either be displayed as a k x T table for every group, tables of dimension k x n for every year, etc. The data preparation should construct a table like Table 3.3, which shows the first few lines of the `gini` data set available in the *sparkTable* package. The *sparkTable* package can currently handle *3-dimensional* tables, whereas such a table is structured within the package in a special manner. Further development is aiming for the usage of higher dimensions.

The *sparkTable* package formats three-way tables into a list of lists after calling the *sparkTable_config()* function. First of all it splits the meta-object into two parts, a list containing the information (`metaInfo`) and a list containing the data (`metaData`). The first one consists of a list (i.e. it is a list of a list) with information about which variables are used, what should be calculated with them, what groups have been chosen and so on. The `metaData` list contains information about each cell of the data which gets displayed in the table. The first index represents a list containing the rows of the table, the second index is a list within the previous list representing the columns (for example *metaData$[[1]][[3]]* is the third cell in the first row). The user is able to specify the different columns of the table by an object called `typeNumeric`, which can either be a list or a vector. This list or vector can contain every function that returns a single value, like the mean or the maximum. The order of this object defines the column number (element of the second list in `metaData`). All information of the sparkline is added to this list. Note that the names of the rows and columns are excluded from the indexing. If the chosen list element is a sparkline, the element consists of another list of parameters with data and all necessary graphical parameters for the sparkline. These parameters can be changed by the user. An application example is given in Section 3.5.3.

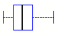
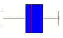
Time series plots

The time series plots  provided bythe *sparkTable* package are common time series plots with certain additional options. It is possible to highlight specific values (like the minimum, maximum, mean) in different colors, sizes or symbols. In addition, the 50% of the inner data points, determined by the interquartile range (IQR), can be additionally visualized in the plots, see .

Bar plots

These are classical bar plots with the possibility to change the bar widths, heights, borders or colors . This kind of bar plot is mostly used to visualize differences or changes over time.

Boxplots

The visualization of outliers in the boxplots  can be deactivated if the data are too distorted, and different colors of the boxplots can be chosen. .

3.5.2 An application of sparkTable with EU-SILC data

Gini data set

Table 3.3 presents the first few entries of the Gini data set. For most countries data from 2004-2007 is given, for some countries the data for 2004 are missing because they were not present in the EU-SILC data set for that year.

The Gini data set is available in the *sparkTable* package, and can be loaded with `data(gini)`. The values have been calculated within the *AMELI* project.

3.5.3 A guided tour

Table 3.3 shows a part of the data set which is used for demonstrating the package in the following. First of all, a meta object with the *sparkTable_config()* function is produced by the following code:

	year	country	gini
1	2004	AT	25.77
2	2005	AT	26.13
3	2006	AT	25.33
4	2007	AT	26.15
5	2004	BE	27.01
6	2005	BE	28.53

Table 3.3: First 6 observations of the Gini data set, which is available in the sparkTable package.

```
meta <- sparkTable_Config(gini,
groups=c("AT","DE","IT"), groupVar="country", vars=c("gini"),
     typeNumeric=c("1--4","mean", "max"),
     typePlot="line", output="eps")
```

Austria, Germany and Italy have been chosen, which act as our groups, and the group variable is called `"country"`. Furthermore we want to take a look at the variable `"gini"` which is the only variable in this data set. From the value of this variable `gini` the mean and the maximum is of interest over the years, as well as the values of every year. These values are displayed by the `"1--4"` code. The output should be a time series plot `typePlot="line"` and it should be exported in `"eps"` format, to be easily included it into this diploma thesis. The row names look better in the output if the full names of the countries are used.

```
rowVec <- c("Austria", "Germany", "Italy")
colVec <- c("Mean", "Maximum","Line-Plot")
```

The column names are modified similarly (`colVec <- c("...")`). Now only a final step is needed which produces Table 3.4:

```
eps.text <- print.sparkTable(meta, outdir="examples",
                             rowVec=rowVec, colVec=colVec, outfile=NULL)
```

With the last command the code generating Table 3.4 gets printed in **R**, and the result just needs to transfered into LaTeX. In addition, if also an `outfile` is defined in the previous code line, a standard LaTeX-file with the table is produced.

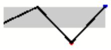
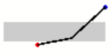
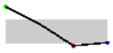
	Line-Plot	2004	2005	2006	2007	Mean	Maximum
Austria		25.77	26.13	25.33	26.15	25.84	26.15
Germany		*NA*	26.26	27.03	30.63	27.97	30.63
Italy		33.24	32.75	32.13	32.22	32.59	33.24

Table 3.4: Table of Gini coefficients for Austria, Germany and Italy, produced by the sparkTable package.

Figure 3.4 indicates that no data is available for Germany in the year 2004. Therefore, the sparkline for Germany is starting one year later as Austria or Italy.

Assume that the representation of Table 3.4 does not look like properly, because of the small size of the points. By changing the parameter `pointWidth` in the function `setPara()`, the size of the points can be increased.

```
meta$metaData <- setPara(meta$metaData, "pointWidth",3 )
```

If this is not satisfying, for example, if the point size specifying the Gini coefficient for Germany is too small, the settings for one specific sparkline can be modified. In this example the IQR box will be removed and the color of the last observation is changed to red. The **R**-code for this changes looks like this:

```
meta$metaData[[2]][[7]]$showIQR <- FALSE
meta$metaData[[2]][[7]]$colVals <- c("#f00","#0f0","#f00")
```

`meta$metaData[[2]][[7]]` are the 'coordinates' for the German sparkline. `[[2]]` represents the second row of the table, and `[[7]]` is the column of the sparkline. Colors in the *sparkTable* package are encoded as hexadecimal. To get a picture of all possible sparkline settings take a look at the help files of the package (`help("setPara")`). Now just call

```
eps.text <- print.sparkTable(meta, outdir="examples",
rowVec=rowVec, colVec=colVec)
```

again. The result is shown in Table 3.5.

This is just a very basic example, various more complex tables can be made with this package, which is still under development.

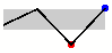
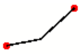
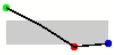
	Line-Plot	2004	2005	2006	2007	Mean	Maximum
Austria		25.77	26.13	25.33	26.15	25.84	26.15
Germany		NA	26.26	27.03	30.63	27.97	30.63
Italy		33.24	32.75	32.13	32.22	32.59	33.24

Table 3.5: Modified Table of the Gini coefficients for Austria, Germany and Italy.

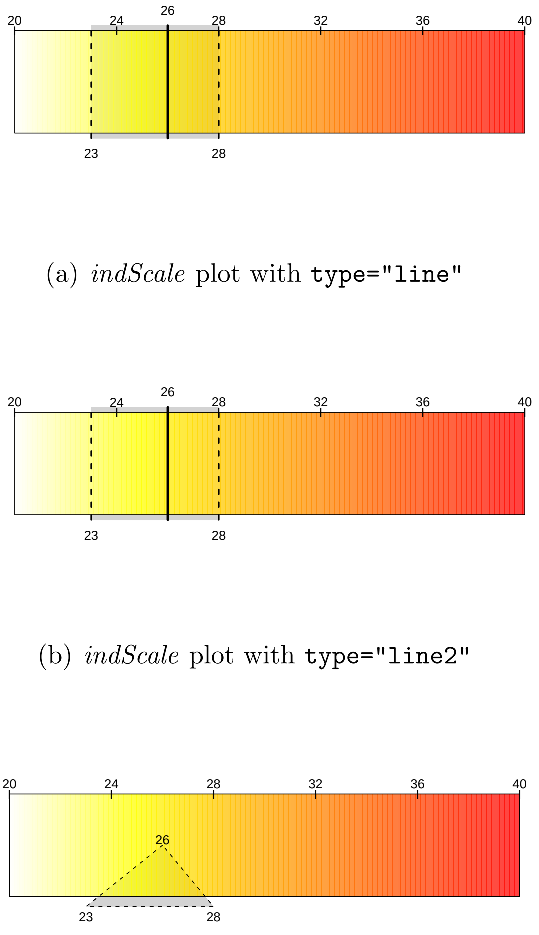
3.6 Confidence Intervals

Various graphical methods have been described so far to visualize indicators. Now the disadvantages of some of these methods are described. The main drawback is that only the point estimates are visualized in the previous figures (like in Figure 3.8(a)), but their uncertainty is not reported. From a statistical point of view this is not enough, a way to visualize confidence intervals has to be found.

Figure 3.6 shows different ways of visualization, the point estimate of an indicator value of 26, and the following confidence interval of $[23, 28]$. This is just an example for illustration and our estimations of the Laeken indicators have significantly smaller confidence intervals. In order to be still able to see the intervals the scale got changed. Figure 3.10(a) shows a shaded confidence interval on the scale. This kind of visualization is handy for plots which get scaled-down. This option is used for the sparklines in Table 3.6. Figure 3.10(b) illustrates the confidence intervals above and below the scale, but not within the scale like in Figure 3.10(a). This way the color range of the indicator scale does not get distorted. The last Figure 3.10(c) shows another possibility of visualization, by using a triangle for the confidence intervals.

3.6.1 Gini coefficients for Austria

In Table 3.6 the Gini coefficients for Austria as well as for the three NUTS1 levels of Austria are displayed. In the region AT1 (Eastern Austria, consisting of Vienna, Lower Austria and Burgenland) the Gini coefficient has the highest value. The Gini coefficient measures the inequality of the income distribution (the higher the inequality, the higher the Gini coefficient), and the high value for Eastern Austria is explained by the difference of income between Vienna, Lower Austria and Burgenland. The gross regional product per inhabitant of Vienna was 43.300 in the year 2007, while it



(a) *indScale* plot with `type="line"`

(b) *indScale* plot with `type="line2"`

(c) *indScale* plot with `type="tri"`

Figure 3.10: Different plot methods for the *indScale* function to visualize confidence intervals for indicators.

was just 21.600 for Burgenland, so around the half. Compared to the others, region AT2 (South Austria) has a low Gini coefficient. South Austria consists of Styria and Carinthia, both states have nearly the same gross regional product per inhabitant, 27.800 for Carinthia and 28.200 for Styria[3].

Someone could now interpret these numbers as 'South Austria is the richest region in Austria because it has the lowest Gini coefficient'. This would be a misinterpreta-

[3] source: `http://www.statistik.at/web_de/services/wirtschaftsatlas_oesterreich/oesterreich_und_seine_bundeslaender/021513.html`

tion of the Gini coefficient, because for a region in which everyone has a low income the Gini coefficient would be very low.

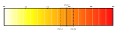
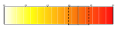
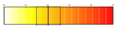
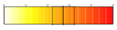
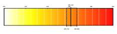
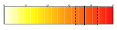
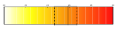
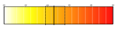
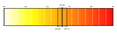
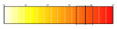
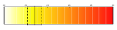
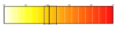
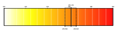
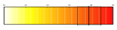
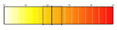
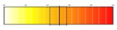
	region	year	ind	lower	upper	sparkline
1	AT	2004	25.77	25.14	26.35	
2	AT1	2004	26.79	25.94	27.79	
3	AT2	2004	24.05	22.97	25.16	
4	AT3	2004	25.42	24.43	26.45	
5	AT	2005	26.13	25.74	26.68	
6	AT1	2005	27.38	26.54	28.56	
7	AT2	2005	25.87	24.61	26.69	
8	AT3	2005	24.58	23.8	25.58	
9	AT	2006	25.33	24.93	25.77	
10	AT1	2006	27.47	26.63	28.14	
11	AT2	2006	22.83	22.15	23.47	
12	AT3	2006	24.14	23.68	24.82	
13	AT	2007	26.15	25.59	26.62	
14	AT1	2007	27.78	26.75	28.89	
15	AT2	2007	24.38	23.57	25.30	
16	AT3	2007	25.08	24.21	25.75	

Table 3.6: Table of the Austrian Gini coefficient and the Austrian NUTS1 level, including the confidence intervals for the Gini.

The confidence intervals in Table 3.6 have been calculated in **R** with the function *bootVar* from the package *laeken* (Alfons et al. 2010). It computes the confidence interval estimation based on a bootstrap resampling method, to be more precise it is a bootstrap percentile interval. This works as follows (Efron and Tibshirani 1986):

Given the data X, n samples of the data are taken with replacement. For every sample X_{boot_i} the unknown parameter $\hat{\theta}$ of interest is estimated, in this case the Gini coefficient. That way n indicator values $\hat{G}_1, \ldots \hat{G}_n$ are calculated. The bootstrap percentile confidence interval is determined by percentiles of the bootstrap distribution. That means leaving off $\alpha/2 \cdot 100\%$ of each tail of the distribution. α is the given significance level which is 5% in this case.

The percentile method was used because it cannot be assumed that the income is normal distributed. This way the confidence interval is asymmetric around the point estimator of the Gini coefficient. The smaller confidence intervals for AT compared to the NUTS1 regions AT1-AT3 are explained by the larger sample which is available for Austria.

3.7 Weather Indicators

Nearly everyone will associate 'good weather' with sunshine. People who live in or near deserts will probably disagree, but this thesis assumes that a majority will consider sunshine as something better than rain. Sunshine is most important for several parts in the human life, e.g. the production of Vitamin D. Vitamin D is produced by our skin in response to exposure to ultraviolet radiation from natural sunlight. The more clouds are seen in the sky, the worse the weather gets in the minds. If it is raining as well it is called beastly weather, with ending in the worst case, a thunderstorm.

The idea is to use this point of view of 'good' and 'bad' weather to visualize indicators. Development over the last years or forecast for the next years are only two examples how to use the 'weather indicators'. Figure 3.11 shows one possible visualization method of this indicators.

The example shown in Figure 3.11 visualizes the import, export, employment, inflation, Gini coefficient and S80/S20 Quintile Share Ratio. It shows the trend for those indicators for the next years using artificial data. At the first look a mostly positive forecast for Austria and a relatively negative one for Spain is seen. Notice that “import” has a more optimistic prediction in Central Europe than in Spain, and that the employment indicator for Spain looks alerting. The S80/S20 Quintile Share Ratio appears bad for all countries in this forecast, so the inequality of the income distribution will increase according to our artificial data.

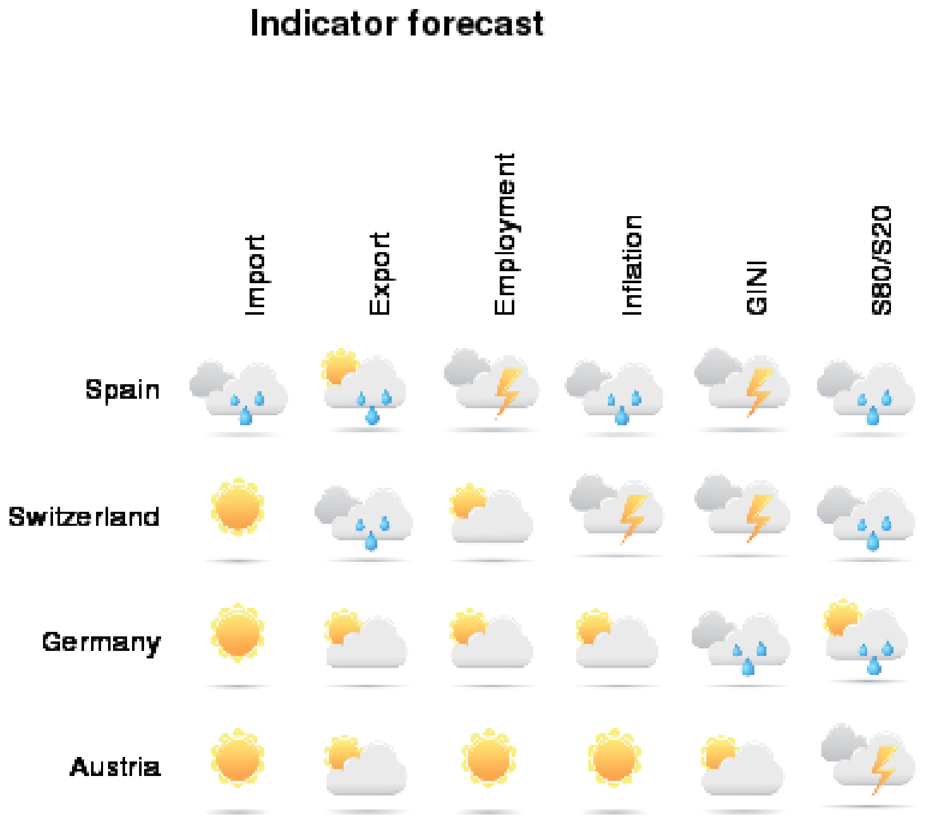


Figure 3.11: Prediction of Indicators for different countries.

Chapter 4

Interactive Maps

In this chapter several topics will be addressed. The main part is the visualization of indicators in maps. This maps should also be interactively usable by the users. The chapter starts with a short introduction into the right usage of colors, which is an important issue when it comes to the visualization of maps. Another section in this chapter be about NUTS levels, which are a key topic in European statistics when it comes to regionalized statistics. Several approaches how to calculate connections between countries and indicators are given in the next section about correlation. The last part of this chapter presents an interactive map about the visualization of correlations.

4.1 Usage of colors

Color is an important part of most graphical displays. Especially statistical and geographical graphics are dependent on colors. Wrong color settings can lead to distortions in the perception of the colors. But how to choose appropriate colors? Zeileis et al. (2009) outline three main points which should be fulfilled when choosing colors:

- Colors should be appealing:

  Statistical graphics do not need to be trendy, but the used software should provide the user with an easy way to select colors. A color is percepted by three properties: hue, brightness and saturation. Most software packages specify colors in the RGB (red-green-blue) space. Computer and TV screens work with

RGB colors, but the human eye does not. Another possibility to describe a color is in the HSV or HSL color space. HSV stands for hue, saturation and value while in HSL the term value is replaced by lightness. The usage of HSV colors motivates for using highly saturated colors. Because of that another perceptually-based color model has been developed, the HCL (Hue-Chroma-Luminance) space. This is the color space we are aiming at because this model mitigates problems of perceptual properties in the HSV space.

- Colors should cooperate with each other:

  In statistical graphics the purpose of using colors is to make it easier for users to distinguish between different symbols, areas or levels. Normally, plots will have several colors, which should be related to each other. In a perceptually-based color space (like the HSV space) that means to move along a path in the space and selecting colors from it. While it is hard in the HSV space to find colors which are harmonic to each other, this problem is solved in the HCL space. The HCL space makes it easier to understand the movement within the color space and the distances between colors.

- Colors should work everywhere:

  Well chosen color sets should work in different situations. If the graphic is plotted on a gray scale printer, or if it is projected by a video projector, the areas should still be distinguishable, even if the viewer is color blind. In Harrower and Brewer (2003) an online tool is presented for selecting color schemes. It provides a lot of information, but the color palettes are fixed. Zeileis et al. (2009) suggest methods how to move on paths along perceptual axes in the HCL color space, leaving the user the final choice of the color settings.

- Color blindness:

  While choosing colors especially color-blind persons have to be taken into account. Approximately $5-10\%$ of males have some kind of color-blindness. Males are more affected because they have just one X-chromosome, and females can compensate possible mutations with the second X-chromosome. The most common type is a red-green-blindness. So especially graphics with red-green scales should be avoided.

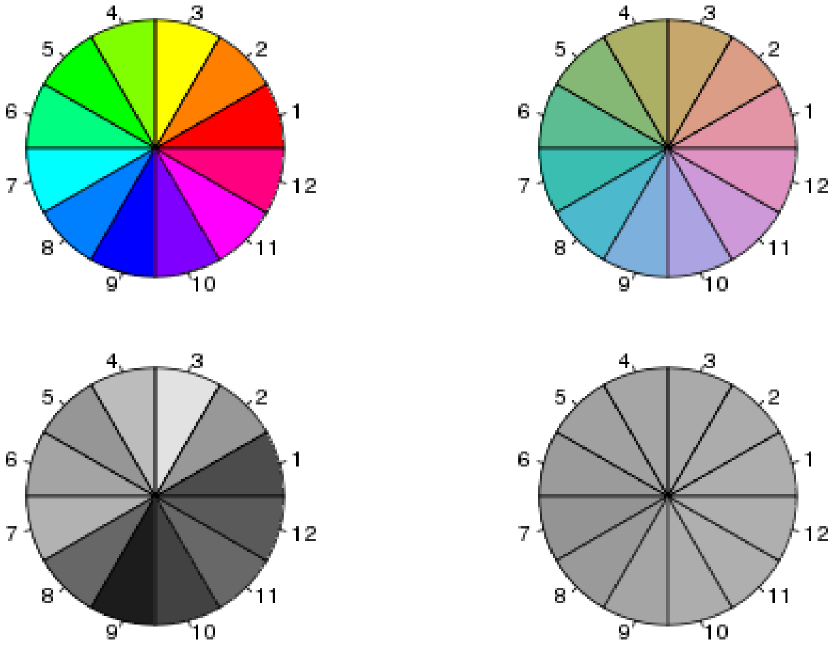


Figure 4.1: The two top circles are filled with 12 colors, the left one is made with the default *rainbow*() function, the right one with *rainbow_hcl*() one. The bottom circles visualize how colorblind people would see them, or how they look like in gray-scale.

There exists a hypothesis that the human eye visions color in three different stages, which are exactly the three dimensions of the CIELAB color space (Wikipedia 2010):

- perception of the light/dark contrast
- yellow/blue contrast
- green/red contrast

Colors are typically described in three-dimensional spaces, the three dimensions used by humans to describe colors are typically:

- hue (dominant wavelength)
- chroma (colorfulness, intensity of color compared to gray)
- luminance (brightness, amount of gray)

The two spaces (HCL, HSV) mentioned above do not fit to each other, so a transformation from one to the other is needed. The most common implementation of this color space in statistics software are HSV colors. HSV colors are the transformation of RGB to a triplet (H, S, V) with $H \in [0, 360]$ and S,$V \in [0, 100]$.

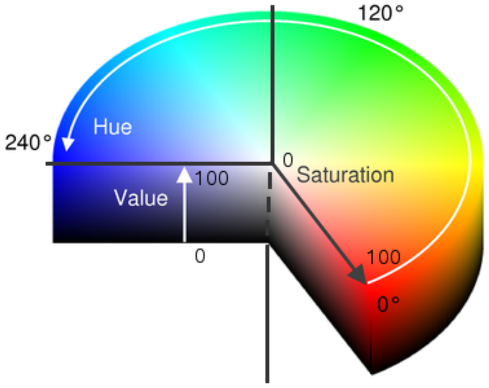


Figure 4.2: The HSV colors, visualized in a cylindrical space.

Figure 4.2 presents the HSV color space, where hue has an angular dimension, starting with red at 0°, passing green and blue at 120° and 240° respectively. The vertical axis represents the gray-scale, black has a value of 0 and white has value 100. The saturation reaches from 0 in the middle of the cylinder to 100 at the outer boarder, where the color has the highest saturation.

However the three dimensions of HSV colors do not match the human color perception very well. For example, the brightness of colors is not uniform over hues and saturations at a given value. Therefore, other color spaces were developed like CIELUV or CIELAB in Commission Internationale de L'Eclairage (2004) which consider this fact. However, a perfect color space will never exist due to the different perception of every individual.

In the next step of development the HCL colors are implemented. They are described by a triplet (H, C, L) with $H \in [0, 360]$ and $C, L \in [0, 100]$. How HCL looks in comparison to HSV is shown in Figure 4.1. There the difference between HCL and HSV based colors can be noticed. The HCL circle has the same luminance level, so all colors resulting from different combinations of the hue and chroma have the same level of brightness, and they look identical in a gray scale. This can be tested by the **R**-function *col2gray()* in the package *TeachingDemos* (Snow 2009), and the usage of HCL colors requires the package *colorspace* (Ihaka et al. 2009).

In the following, the transformation from sRGB color space to the HCL color space is described. The sRGB color space is a standardized RGB color space created by Microsoft and HP in 1996, it approximates the color gamut of the most common computer device displays. It has also become a standard color space for displaying graphics on the Internet. The sRGB color space can only visualize around 35% of other CIE color spaces, but that is considered as good enough for most applications. The XYZ color space mentioned below was designed to be able to describe all colors which are visible for the human eye. A detailed transformation is given in an article written by Juckett (2010). Only a short summary about the color transformations is given, which pass three different color spaces.

Transformation from sRGB to XYZ color space

The CIE XYZ color space is one of the oldest color spaces, it got defined by the CIE (Commission Internationale d'Eclairage) in 1931. The X, Y and Z values can roughly be seen as derived red, green and blue values. To be exact this are the tristimulus values of a color, which are are the amounts of the three primary colors in an additive color model.

Transformation XYZ to CIELAB color space

CIE L*a*b* (CIELAB) is another color space specified by the International Commission on Illumination (Commission Internationale d'Eclairage). All colors visible to the human eye are described by this space. It approximates the human vision, unlike in the RGB color space. The three coordinates of CIELAB represent the lightness of the color (L* = 0 yields black and L* = 100 indicates white), a* is the position between red/magenta and green (negative values indicate green while positive values indicate magenta) and b* defines the position between yellow and blue (negative values indicate blue and positive values indicate yellow). Any two dimensional illustrations use a fixed lightness factor.

CIELAB to CIELCH

The LCh color space is the same as the CIELAB color space, but is expressed via cylindrical coordinates (Wikipedia 2010). Finally this leads to the hue (H), chroma (C) and luminance (L) values, which generates a color space. This space is the same as the HCL color space mentioned in Zeileis et al. (2009).

4.2 NUTS levels

One basic topic of the European Statistical System is the regional statistics[1]. The European Union developed NUTS as a geocode standard for statistical purposes. *NUTS* is the abbreviation for 'Nomenclature d'Units Territoriales Statistiques'. All 27 EU member countries (EU-27), all candidate countries (Croatia, Macedonia and Turkey) as well as the EFTA countries Island, Norway, Liechtenstein and Switzerland are divided into four NUTS levels. Every statistics transmitted to the European Commission should be broken down by NUTS classification, as well as the analysis of this statistic. The NUTS2 level is of most interest to the European Union. To ensure comparability of regional statistics, the regions should be built on the same thresholds, like shown in Table 4.1.

Table 4.1: Population Threshold for NUTS levels regarding to people living there.

Level	Minimum	Maximum
NUTS0	the whole country	
NUTS1	3 000 000	7 000 000
NUTS2	800 000	3 000 000
NUTS3	150 000	800 000

The *NUTS* code starts with two letters referring to the country. This country is subdivided into several regions, described by one number. Another number expresses the second and third subdivision. An example is given in Table 4.2 where the first two letters correspond to the country code (AT for Austria)

Table 4.2: Example: NUTS hierarchy of Carinthia(AT)

	NUTS0	NUTS1	NUTS2	NUTS3
AT	Austria			
AT2		South-Austria		
AT21			Carinthia	
AT211				Klagenfurt-Villach
AT212				Oberkaernten
AT213				Unterkaernten

[1] http://epp.eurostat.ec.europa.eu/portal/page/portal/nuts_nomenclature/introduction

There is even a lower administrative division of a country, the so called LAU levels. LAU stands for 'Local Administrative Unit'. For every EU country two LAU levels are defined. Note that before 2003 LAU-1 and LAU-2 have been known as NUTS4 and NUTS5. In some countries LAU-1 is the same classification as NUTS3 (mostly smaller countries) and LAU-2 are mainly defined as municipalities.

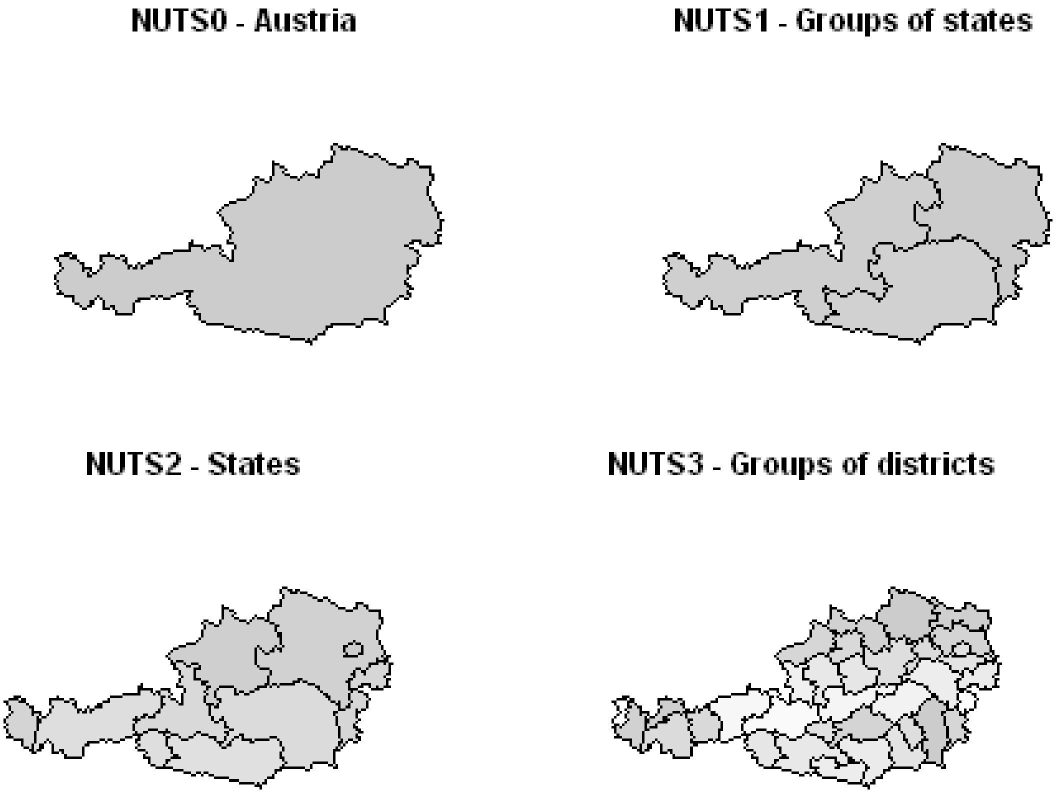


Figure 4.3: Map of Austria, showing regions of NUTS0 to NUTS3.

Dividing to specified NUTS levels allows to compare different NUTS regions. Figure 4.3 shows the NUTS classification of Austria from NUTS0 to NUTS3. Note that most Eurostat statistics are based on NUTS2 levels. For policy issues it would be of interest to estimate the Laeken indicators at NUTS2 level. However, the EU-SILC data almost only includes NUTS1 geocodes.

4.3 Visualizing Maps

4.3.1 About GIS and spatial data [1]

Spatial data are also known as data including geospatial or geographic information. They contain the geographic location of the observations, and probably the boundaries on the surface on Earth, such as natural or constructed features, oceans and so on. Such data are stored as coordinates and topology and can also be mapped.

GIS is a short form for Geographic Information Systems. These are tools used to gather, transform, manipulate, analyze, and produce information related to the surface of the Earth. Such data may exist as maps, 3D virtual models, tables, and/or lists.

Often the spatial data are stored as shapefiles. Shapefiles contain a lot of other information too. To be exact, a shapefile is just an umbrella term for several data files. The main file (the *.shp* file) contains the primary geographic reference data, like points, lines or polygons. The other two required files have a $.shx$ and $.dbf$ extension. The index file ($.shx$) stores the index of the geometry (that was more useful when computers had limited memory) and is a connection to the data in the $.dbf$-file, which stores the variable information of the observations. These three files are the main files of spatial data, additional files can contain projections ($.prj$) or other properties of the shapefile.

4.3.2 Projections

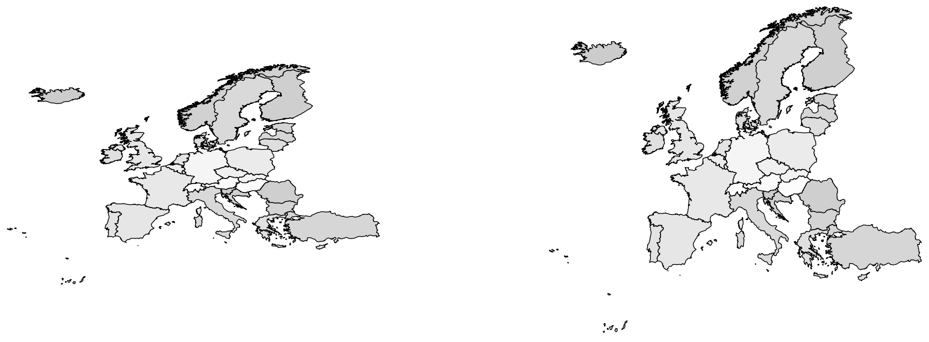
The visualization of shape files in the correct way is an important issue. For that reason a short introduction to map projections will be made. A map projection is a mathematically described technique to represent the curved surface of the Earth on the flat surface of a map. The aim is an accurate map which represents the surface of the earth. Before getting to projections various coordinate systems in use will be explained:

- **Latitude and longitude:** Latitude tells us the position of a place on earth north or south of the equator in degrees (north/south pole are at 90°N/90°S, while longitude is the angular distance of this point to the Greenwich Meridian. The latitude/longitude coordinates are given by (ϕ, λ)

[1] Definitions of Spatial data and GIS from www.webopedia.com

- **Universal Transverse Mercator (UTM):** The UTM system divides the surface of earth into 60 zones. Each of them has a width of 6 longitudes, and the zones are defined between 80°S and 84°N latitude.

- **The Mercator projection:** This is a cylindrical map projection which became the standard map projection for nautical purposes. The Mercator projection causes a big distortion of the size and shape of large objects, especially of objects which are far from the equator. For example Greenland and Africa have about the same size with the Mercator projection, while in reality Greenland is only around 1/14 of the size of Africa.



(a) Plot of the map of Europe with latlong projection

(b) Plot of the map of Europe with LAEA projection

Figure 4.4: Difference of the plot output: (a) is the default plot and (b) uses a projection function.

For using such maps it is important to have a minimum of distortion like in Figure 4.4(b). There are dozens if not hundreds of map projections in use, and every projection includes some kind of distortion, for example in shape, distance, area, direction or any combination of them. The aim is to find a projection that maintains accurate relative sizes. This is called an equal area, or equivalent projection. These projections are used for maps where accurate displayed maps are important. In other fields of science like physics conformal map projections are more important, these are projections which preserve angles locally. One example is the Lambert Azimuthal

Equal-Area (LAEA) projection. Azimuthal means that the projection maintains accurate directions, when a central point is given. In plots like Figure 4.3 or all figures about Europa in the following sections the LAEA projection for minimal distorted maps is used. The recommended European standard grid is called *Grid_ETRS89-LAEA5210.* ETRS89 is the abbreviation for European Terrestrial Reference System from the year 1989. LAEA5210 means a Lambert Azimuthal Equal-Area with the central point 52°N and 10°E.

This coordinate transformation of a point uses intricate trigonometrical functions. The formulation of this transformation is not the aim of this thesis since these transformations are filling pages. However, if someone is interested in those formulas, they can be looked up at the following website: `http://www.epsg.org/guides/docs/G7-1.pdf` at page 61.

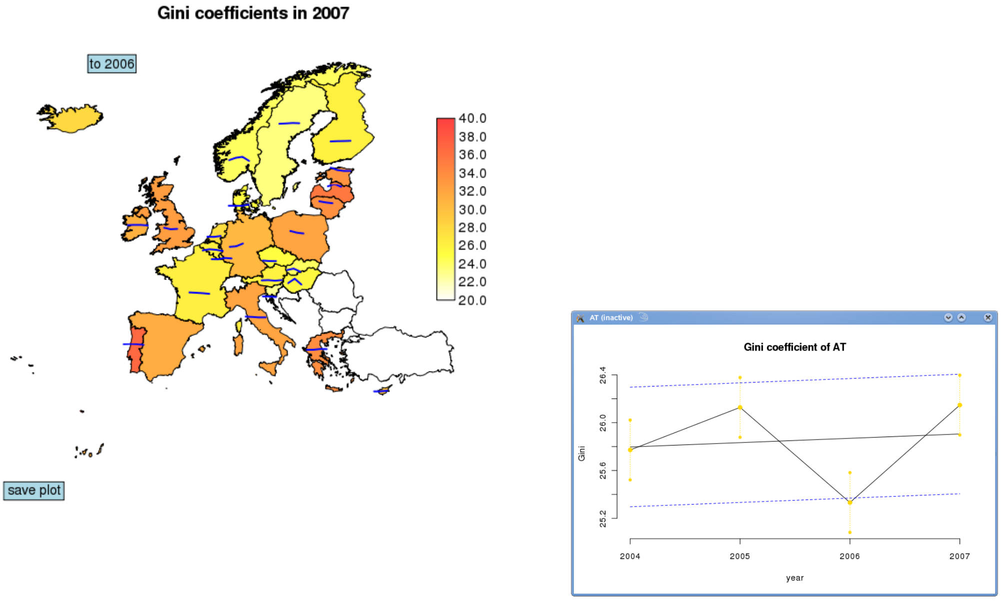
For Figure 4.3 such a transformation from latitude/longitude coordinates to LAEA coordinates for the map of Austria is used. The central point for this map is the middle point of the bounding box. The bounding box of a map is a rectangular, which touches the most north/east/south/west points of the map. In **R** this projection works as follows:

```
laea.austria <- spTransform(austria,CRS("+proj=laea
+lat_0=bbox(austria)[2]+(bbox(austria)[4]-bbox(austria)[2])/2
+lon_0=bbox(austria)[1]+(bbox(austria)[3]-bbox(austria)[1])/2
+x_0=400000 +y_0=4000000 "))}
```

where `austria` is the map of Austria in datum lat/long coordinates, `lat_0` and `lon_0` represent the central point, calculated as mentioned above. `bbox[2]` and `bbox[4]` are the southernmost and northernmost points of Austria and `bbox[1]` and `bbox[3]` are the westernmost and easternmost points. `x_0` and `y_0` represent False Easting and False Northing, which prevents negative x/y values after the transformation. This is important because some geographic software products can not handle negative numbers. **R** provides various packages about maps like *maptools*, *maps*, *rgdal* and many more. For our purposes the package *rgdal* (Timothy et al. 2010) has been used. For detailed information about projection visit `http://trac.osgeo.org/proj/`, the homepage of the *Cartographic Projections Library.*

4.3.3 Visualization of the Gini in a map of Europe

In the previous sections the usage of colors in maps, different NUTS levels and projections of maps have been mentioned. Now this knowledge gets combined with the theory of Chapter 3. One result of this is the *giniVis()* function. The **R**-code can be found in Appendix A, this section focuses on the output.



(a) Gini coefficients for European countries (b) Gini coefficient for Austria

Figure 4.5: Interactive map of Europe to visualize the Gini coefficient.

Figure 4.5(a) and Figure 4.5(b) display the two output plots of the *giniVis()* function. The left figure shows the Gini coefficients for most European countries. The background color for each country represents the current value of the Gini coefficient in the year 2007. Yellow to red is the used scale for this plot, while yellow indicates a small Gini coefficient and red indicates a high one. In the foreground of the countries a small sparkline presenting the trend of the last years can be seen. It should give a first small picture of the development of the indicator in that year over the last years. All sparklines in this map use the same scale of the y-axis. This is made to get the user focused on large changes in the time series, like Hungary in Figure 4.5(a). If only one value is present for a specific country no sparkline is shown. There can be various reasons for having only one value. Either this is the only year available in the *EU-SILC* data set, or in the previous years there is a problem with the calculation of the Gini coefficient. The scale on the right hand side of Figure 4.5(a) is variable, and should be default reach from the *minimum* to the *maximum* of the indicator values of all countries in the presented year. In order to get consistent scales and plots over the years we could also use a scale which reaches from the *minimum* to the *maximum* of all indicators in every year.

The interactivity in this plot consists of two parts. One can move between the different years using the *"to 2006"* button in Figure 4.5(a), which will change the color of each country. To get a closer look at the time series of one country one has to click on it. This opens Figure 4.5(b) in a new window (in this case a closer look at Austria is taken), presenting the time series with the *indEval()* function from Chapter 3.

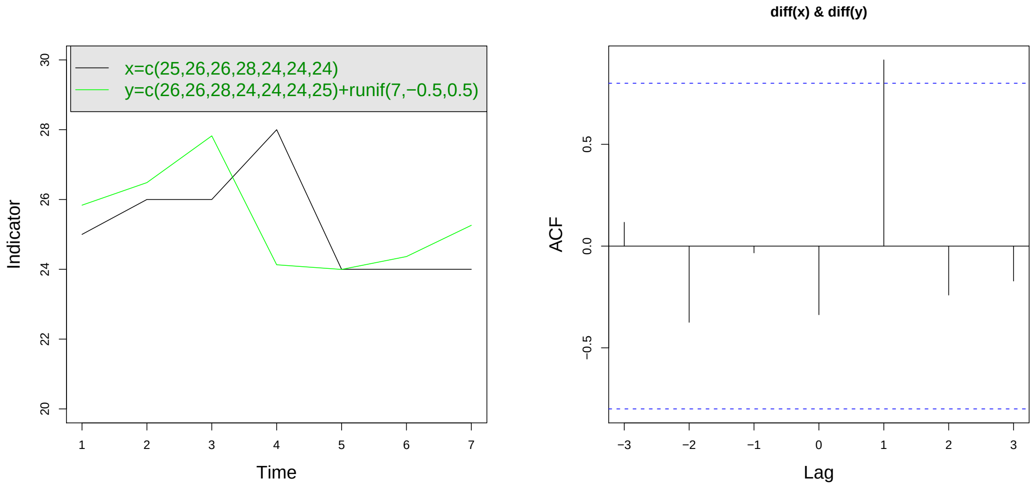
4.4 Correlation between indicators over time

The data bases (like the World Bank Database) are getting larger every year, providing better information. But this also makes it harder to visualize that huge amount of data. Gunawardane et al. (2007) developed an interactive interface for correlation analysis to visualize the relationship between various global indicators and countries. A part of this paper focuses on the visualization of correlation on a world map.

To be able to present correlations in a map, the estimation of the correlation is needed. In the following sections several approaches to estimate correlation are discussed. At the beginning cross correlation is introduced. However, these formulas should only be applied on stationary time series, which is mentioned in the next section. An extension to this is the estimation of weighted correlations, which gives higher weights to actual observations. An example of these methods is given by using **R** to explain these methods in practice.

4.4.1 Cross correlation

The relations between different countries concerning a single indicator are discussed in this section. This relation is expressed by the cross correlation. Time series can be correlated at the current time, or being correlated at any lag, i.e. shifting the time of a given time series. Let us take a look at an example. The economies of Austria and Germany are closely connected to each other, which has historical and cultural reasons. It can be assumed that the economy in Germany influences the economy in Austria with a delay of one year, a usual correlation (like shown in Figure 4.6(b) at *lag 0*) could not notice that, or produce even misleading results.



(a) Two similar time series.

(b) Cross-correlogram of the two time series.

Figure 4.6: Cross correlation analysis of two time series.

To face that problem the cross correlation function comes into use. The cross correlation function is a standard method for estimating the degree to which two time series are correlated. The cross covariance function between x_t and y_t (for $t = 1, \ldots, T$) is defined as

$$\gamma_{xy}(k) = \begin{cases} E[(x_t - \mu_x)(y_{t+k} - \mu_y)] & \text{for } 0 \leq k \leq T-1 \\ \gamma_{xy}(-k) & \text{for } -T+1 \leq k < 0 \end{cases} \tag{4.1}$$

for $k = 0, \pm 1, \ldots, \pm T$. (see, e.g., Wei (1990) and Brockwell and Davis (1996)) Here, μ_x and μ_y are the expectations of the x and y vectors, respectively. k denotes the time difference between the two time series. This leads to the following cross correlation function:

$$\rho_{xy}(k) = \frac{\gamma_{xy}(k)}{\sigma_x \sigma_y} \tag{4.2}$$

for $k = 0, \pm 1, \ldots, \pm T$, where σ_x and σ_y are the standard deviations of the x and y vectors, respectively.

In **R** the cross correlation function `ccf()` can be used. Generally, the above functions can only be used if the time series are stationary, which will be discussed in the next section.

4.4.2 Stationarity

The cross correlation function should only be used, if the time series are stationary. Brockwell and Davis (1996) defines multivariate stationary time series as follows:

A m-variate time series $X_t = (x_{t1}, \ldots, x_{tm})^T$ is (weakly) stationary if

- $\mu_X(t)$ is independent of t and
- $\Gamma_X(t+k, t)$ is independent of t for each k ,

where $\mu_X = (\mu_{t1}, \ldots, \mu_{tm})^T$ is the vector of means. Γ_X denotes the covariance matrix

$$\Gamma_X(k) = \begin{pmatrix} \gamma_{11}(k) & \ldots & \gamma_{1m}(k) \\ \vdots & \ddots & \vdots \\ \gamma_{m1}(k) & \ldots & \gamma_{mm}(k) \end{pmatrix} \tag{4.3}$$

consisting of covariance functions $\gamma_{ij}(k)$ as defined in Equation 4.1.

In order to get an accurate estimation of the cross correlation mentioned below, stationarity of the time series is needed. This is obtained by *prewhitening*, a process which removes systematic or deterministic effects over time. The general idea is to transform the time series to white noise based on one pre-chosen time series, thus the term *prewhitening*. *White noise* is a random process, which has a mean of zero, a constant variance over time, and covariance equals zero between different time points.

Prewhitening can be used in different ways, and there is still a discussion ongoing between researchers about the best method. Most of them make use of an *ARIMA* model (Hayes et al. 2007). Trends can be removed by an ARIMA(0,d,0) model which just applies differencing of order d on the time series, this is also called de-trending. This approach is common to remove deterministic components of time series. The disadvantage of this method is the destruction of the information about the absolute value. Other approaches are using an ARIMA(1,0,0) model, so just focusing on the autoregression. This means that each point of the time series is dependent on the previous.

This method applies suitable filters to the time series, which transforms them into white noise. White noise means that $\mu(t)$ is independent of t and $\rho_{xy}(k)$ is independent of t for each k. Prewhitening generally applies the following procedure:

1. First an *ARIMA* (autoregressive integrated moving average) model is fit on the first time series.

2. Transformation of the correlated input series x_t to the uncorrelated white noise series α_t, which consists in fact of the residuals of the fitted time series.

3. The same transformation is applied on the second time series using the fitted parameters from modelling the first time series, which leads to the second process β_t.

4. Estimate the cross correlation between the two new series.

Prewhitening is a wide spread topic but we will not go too much into details in this thesis. More details about prewhitening and transfer models are discussed in Brockwell and Davis (1996), Hayes et al. (2007) or Box et al. (2008). In **R** the function *prewhiten()* can be used to prewhiten the time series. This function can be found in the package *TSA* (Kung-Sik 2010).

4.4.3 Cross correlation with prewhitening

When non-stationary behaviour is expected for the time series, we can assume that differencing the time series by degree d is inducing stationarity. In the following Δ denotes the differencing operator, i.e. $\Delta(x_t) = x_{t+1} - x_t$ for $t = 1, \dots, T-1$. Hence Δ^d is the differencing operator applied d times on the time series x_t, i.e. $x_{t_d} = \Delta^d X_t$. Note that this way the dimension of the time series also gets reduced by the degree of differentiation. In practice, d is usually 0, 1 or 2. After the time series x_t has been made stationary, an ARIMA model can be fit to the new time series. This transforms the times series to uncorrelated white noise α_t. The same transformation is applied on the second time series y_t, leading to the residuals β_t. Box et al. (2008) explains the usage of prewhitening with an application, where more complex transfer models are used as in our example below.

4.4.4 Weighted correlation

In most cases the actual values of indicators are of a higher interest. Thus a weighted correlation function is needed, which weights the last known values higher than the ones in the past.

The standard correlation is shown in the top left plot of Figure 4.7, where each observation has the same weight. The upper right plot shows linearly decreasing weights over time, when moving into the past. Two non-linear weighting functions are shown in the bottom plots, the left one shows a quadratic decreasing weight while the right one shows an exponential decreasing weight.

For the estimation of a weighted correlation function the `cov.wt()` function is used in **R** which is included in the *stats* package. This function will be explained below:

Let x_{t_d} and y_{t_d}, $t = 1, \ldots, T-d$ define two already differentiated time series. The same method as in the previous section is applied on the time series leading to prewithened time series α_{t_d} and β_{t_d}. Furthermore the weighting vector $w = (w_1, \ldots, w_{T-d})$ of dimension $T-d$, is known. The sum of the weights should be one ($\sum_{t=1}^{T-d} w_t = 1$), if this is not the case the weights have to be normalized by dividing by the sum of weights. The difference to the previous section is the different handling of the residuals. The focus is on the most recent observations, residuals from older time points get lower weights.

The next step is the calculation of the weighted mean of the residuals α_{t_d} and β_{t_d}:

$$\bar{\alpha}_{t_d} = \sum_{t=1}^{T-d} w_t \cdot \alpha_{t_d} \tag{4.4}$$

$$\bar{\beta}_{t_d} = \sum_{t=1}^{T-d} w_t \cdot \beta_{t_d} \tag{4.5}$$

After that the subtraction of those values from residuals of each time series has to be done, so centering them, and multiplying the result by the square root of the weights:

$$\begin{aligned} \alpha^*_{t_d} &= \sqrt{w_t} \cdot (\alpha_{t_d} - \bar{\alpha}_{t_d}) \qquad t = 1, \ldots, T-d \\ \beta^*_{t_d} &= \sqrt{w_t} \cdot (\beta_{t_d} - \bar{\beta}_{t_d}) \qquad t = 1, \ldots, T-d \end{aligned}$$

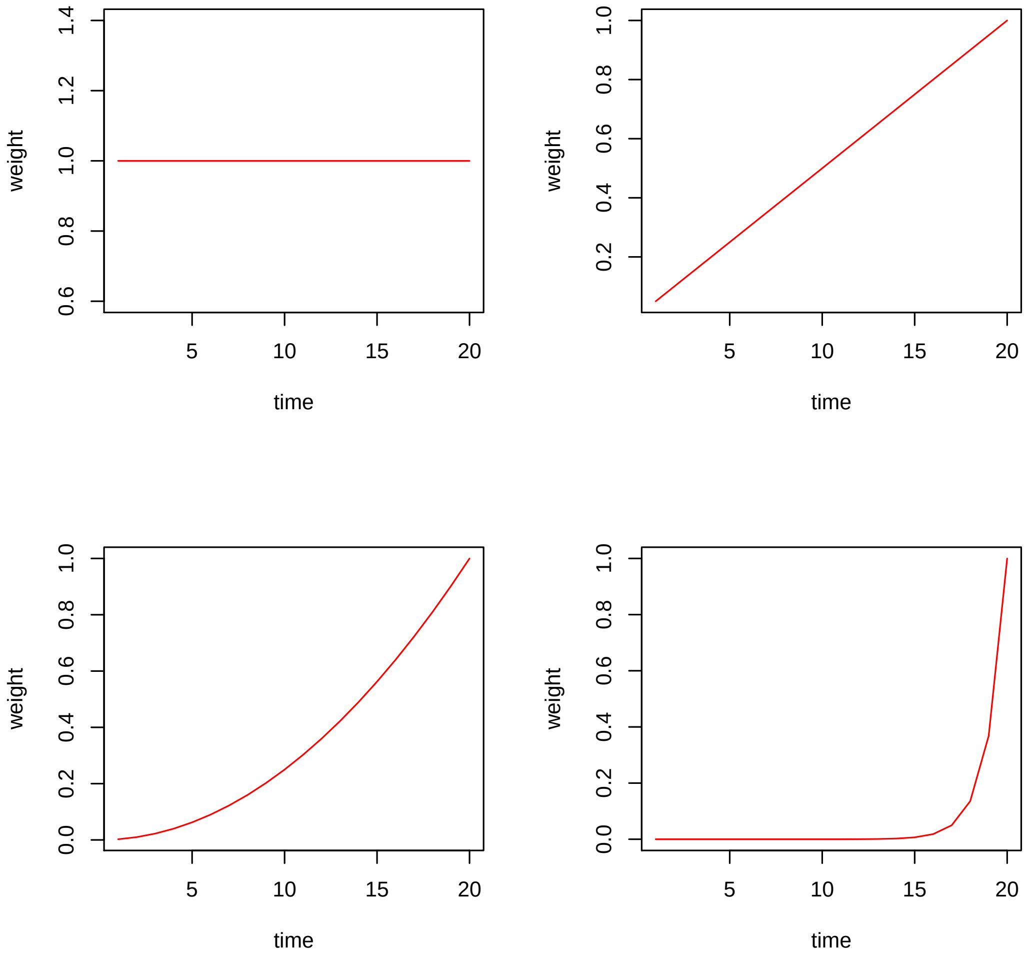


Figure 4.7: Various weighting functions.

Now the weighted covariance can be calculated:

$$s^2(\alpha_{t_d}, \beta_{t_d}) = \sum_{t=1}^{T-d} \alpha^*_{t_d} \cdot \beta^*_{t_d} \tag{4.6}$$

In order to get an unbiased estimation of the covariance the result has to be divided by the scalar

$$1 - \sum_{t=1}^{T-d} w_t^2 \quad .$$

For default weights (all weights have the same value $1/(T-d)$) the conventional unbiased estimate of the covariance with divisor $(T-d)$ is obtained. For calculating the correlation the standard formula is used:

$$r(\alpha_{t_d}, \beta_{t_d}) = \frac{s^2(\alpha_{t_d}, \beta_{t_d})}{\sqrt{s^2(\alpha_{t_d}, \alpha_{t_d}) \cdot s^2(\beta_{t_d}, \beta_{t_d})}} \tag{4.7}$$

where $s^2(\alpha_{t_d}, \alpha_{t_d}) = \sum_{t=1}^{T-d} \alpha_{t_d}^{*2}$ and $s^2(\beta_{t_d}, \beta_{t_d}) = \sum_{t=1}^{T-d} \beta_{t_d}^{*2}$. This $r(\alpha_{t_d}, \beta_{t_d})$ is then an estimation for the correlation between x_t and y_t.

4.4.5 Example of application in R

The discussed points in the previous section are now explained in an example. The time series which gets compared to all others represents the Gini coefficients for Austria as mentioned in Table 3.1. The other used time series in this example are artificial, because the time series of the *EU-SILC* data are too short for this purpose. The following four time series will be compared to each other, which are presented in Figure 4.8.

```
set.seed(1234)
# original time series:
ts1 <- ts(c(27, 26, 25, 24, 26, 24, 24, 25.5, 27, 25.77, 26.13,
    25.33, 26.14, 26.1), start=1995)
# highly correlated time series:
ts2 <- ts(c(27, 26, 25, 24, 26, 24, 24, 25.5, 27, 25.77, 26.13,
    25.33, 26.14, 26.1)+rnorm(14,5,0.5), start=1995)
# highly correlated time series at lag 1
ts3 <- ts(c(26, 25, 24, 26, 24, 24, 25.5, 27, 25.77, 26.13, 25.33,
     26.14, 26.1, 27)+rnorm(14,-3,0.5), start=1995)
# highly correlated time series at the last observations:
ts4 <- ts(c(23, 28, 26, 25, 24, 23, 24, 25.5, 27, 25.77, 26.13,
    25.33, 26.14, 26.1)+rnorm(14,2,0.5), start=1995)
```

The time series are prepared for use by differentiating of order 1 (`diff()` function)

```
ts1d <- diff(ts1)
ts2d <- diff(ts2)
ts3d <- diff(ts3)
ts4d <- diff(ts4)
```

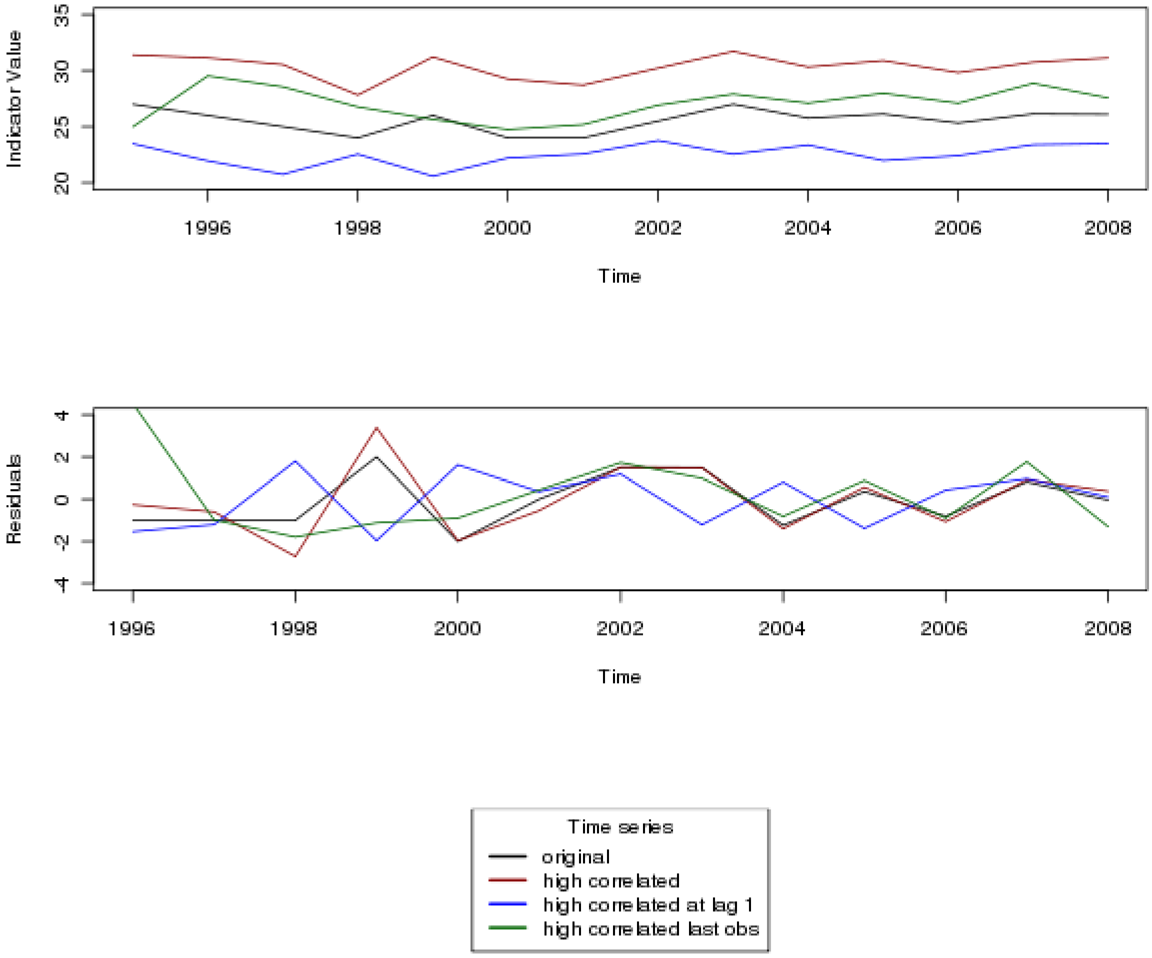


Figure 4.8: Example for time series, which have different forms of correlation to each other. The upper plot presents the original time series, the lower plot the time series after differentiating.

The relation between the first time series to all others is of interest in this example. Since the time series are already differentiated, an ARMA model is used for prewhitening. The ARMA model is now applied on the first time series. This is realized with the `arma()` function in **R**.

```
ts1a <- arma(ts1d)
```

This leads to an ARMA(1,1) model with coefficients $\varphi_1 = 0.35$ for the AR process and $\theta_1 = -2.62$ for the MA process (for details about AR and MA processes have a look at Brockwell and Davis (1996)). The model parameters are then applied to the other time series by using the `prewhiten()` function in the package *TSA*.

```
ts12<-prewhiten(ts1d,ts2d,ts1a)
ts13<-prewhiten(ts1d,ts3d,ts1a)
ts14<-prewhiten(ts1d,ts4d,ts1a)
```

As expected, the first time series has a high correlation of 0.91 with the second one, while the correlation between the first and third time series has a correlation of 0.82 at lag 1. The correlation between the first and the forth time series is low with a value of 0.21. However, we assume that the last observations of the time series are more important for policy. In this example the last observations between the first and the forth time series are higher correlated, which has to be considered. Here comes the weighted correlation into mind. In this example a linearly decreasing weighting function is used, which is then multiplied with the residuals of the two time series after prewhitening. The default output of the `prewhiten()` function does not include the residuals which are getting cross correlated, so a slightly modified `mod.prewhiten()` function was written which includes the corresponding residuals in the output. Those are called α and β like mentioned in the previous section:

```
n <- length(ts1d)
w <- 1:n
w <- w/sum(w)
ts14m <- mod.prewhiten(ts1d,ts4d,ts1a)
ccf(ts14m$alpha*w,ts14m$beta*w)
```

This leads to a better result of 0.68 for the correlation between the first and the forth time series at lag 0. However, this value strongly depends on the weighting function, which should be chosen with care.

4.4.6 Visualization of the correlation

The function `mapCor()` (see Appendix A) provides an EU map with the visualization of correlations between countries. Note that the input for this function is either a matrix or a data frame. The `mapCor()` function can handle three different visualization methods by changing the parameter `chart`, as explained below:

- `"correlation"`: A normal correlation matrix with values between -1 and +1. The visualization will use the whole scale.
- `"minmax"`: instead of the whole scale from -1 to +1 this option reduces the scale. It reaches from the minimum of the correlation to the maximum of the

correlation for the chosen country. This is useful if correlations are close to each other, and the colors are hard to differ with the ”correlation” option.

- `"distance"`: An option to present a distance matrix, consisting of values between 0 and +1.

Calculating the correlation matrix can be done by any of the methods mentioned in the previous section, or in any other way. The correlation of an indicator is shown after picking a single country in the map of Europe.

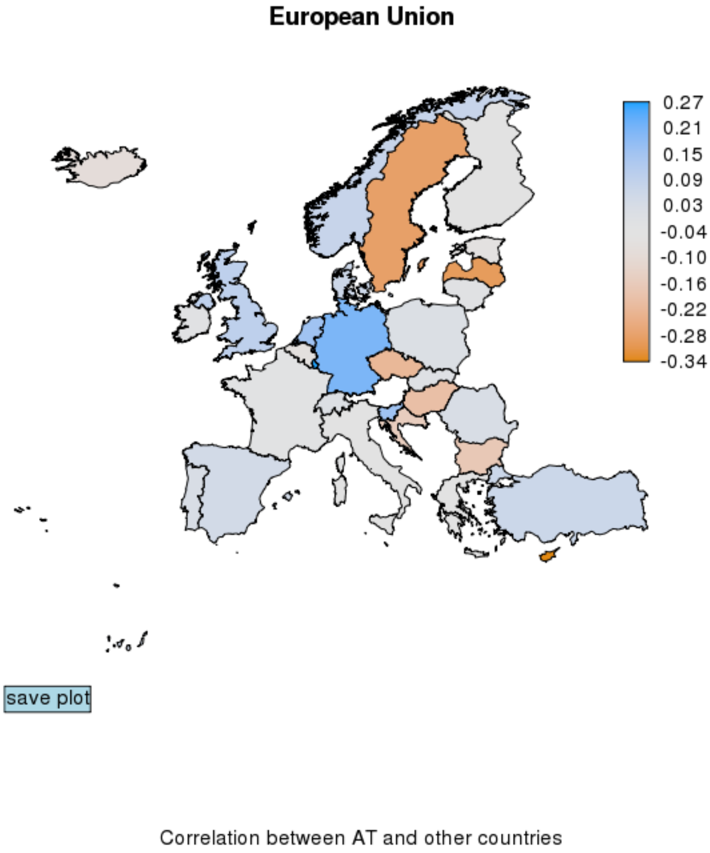


Figure 4.9: Correlation using the HCL color scheme.

Like mentioned above different color schemes are useful for the visualization. Three predefined color schemes are available for the function, in addition there is an option to use user defined color schemes with the parameter `"colorScheme"`.

Interactivity for this function is divided into two parts. On the one hand one can switch between different countries of interest by just clicking into them. The map is then plotted again, using the correlation between the new country to all other countries. On the other hand one can switch between three (respectively four, if a user color scheme got defined) different color schemes and use the one which fits best to the data. Figure 4.9 is an example for such a plot where an artificial correlation matrix for the input is used. In this figure the correlations between Austria and all other countries can be recognized. Most countries do not have any significant correlation with Austria, but Germany has the highest positive correlation, and Latvia is the country with the most negative correlation.

Chapter 5

Summary and Conclusions

The primary focus and motivation for this diploma thesis was the investigation of current visualization methods and the development of new ways to present data. We started with the definition of a set of Laeken indicators in Chapter 2, and methods to estimate them. The focus for the visualizations was on the Gini coefficient, an indicator which measures the inequality of the income distribution. In Chapter 3 the evaluation of indicators has become the main task, based on a paper from Hulliger and Lussmann (2008). We have taken a look on various important question: When is an indicator good/bad, when is it getting better/worse and what influence has the variance of the indicators to the evaluation? The development of presentation methods finally ended in the **R**-package *sparkTable* (Kowarik et al. 2010). The package *sparkTable* is one step towards the direction of presenting both figures and graphics in nice tables. It got developed for the free statistical environment **R** and is freely available.

We also focus on rather new methods for visualization for policy needs. We present the *Weather Indicator*, which is an easy understandable way to visualize the development or forecast of an indicator to the general public. In most communication media, indicators are expressed by single values. In addition reports rarely include confidence intervals. We present a way to visualize confidence levels for indicators in plots, and also the usage of them in tables as small sparklines.

In Chapter 4 we moved on with the presentation of data to visualize them in maps. The chapter starts with a section about the meaningful usage of colors, and what is important when selecting color palettes. The visualization of indicators in maps can be done on different levels, the so called NUTS levels. Politicians are mostly interested in data of areas which they are administrating. Therefore, one important facility is to 'zoom' into the area of interest. However, the problem is the lack of data for smaller regions. Even though the estimation of indicators at NUTS2 level would be of most interest to the European Union, unfortunately the *EU-SILC* data set provided to us mostly includes only information at NUTS1 level. The possibility to 'zoom' into an area of interest leads into a small section about projections of maps. Projections into other coordinate systems are important to avoid distorted maps.

Another interesting question is the correlation between different countries/regions. For example, if the value of one indicator increases in region A, do the values of this indicator also increase in some manner in region B? Which methods should be used to compare time series? One important issue is the *prewhitening* of time series to make them comparable. We focused on the cross correlation, and modified it by introducing weights for observations. The chapter ends with an application example in **R** to explain the methods mentioned before in practice. Especially a robustification of the correlation measures could be very useful in practice and should be investigated in further research

The visualization of indicators for end-users and policy support will get more and more of interest in the future. New methods to present indicators should include easy understandable graphics or tables, moving away from pages full of only numbers.

Open source software environments such as like **R** will get interesting for end-users because of the development of packages which are not hard to handle but still produce nice looking outputs.

Appendix A

R-Code

The code related to this thesis was written in **R**. Some helpful information about **R** can be found in Chambers (2008).

A.1 Evaluation of Indicators

A.1.1 indEval function

This function evaluates indicators like mentioned in Chapter 3.

```
library(robustbase)

`indEval` <-
function(x, x_time=1:length(x), sderr=0, dr=0.01*median(x),
x0=x[1],t0=x_time[1], x1=x[length(x)], t1=x_time[length(x_time)], betax,
eval.labels=TRUE, good.ind=”low”, ind.size=1, Legend=FALSE,
placeLegend=”topleft”, show.axes=TRUE, x.lab=”Time”, y.lab=”Indicator”,
parList=list(cex.lab=1.5, pch=19), sparkline=FALSE, regression=”none”,
plot.title=title(main = NULL, sub = NULL, xlab = NULL, ylab = NULL),
    ...)

# Function indev - Evaluation of indicators
# Beat Hulliger, FHNW
# Modifications by Stefan Zechner
# x: indicator
# x_time: time
# sderr: standard deviation of indicator
# dr: relevant difference of indicator (>0)
# x0: start value
# t0: start time
# x1: target value
# t1: target time
# betax: course
# eval.labels: switch for printing labels in graph
# good.ind: a good indicator is either
#        ”high”,”low”,”infunnel”,”outfunnel”
# ind.size: size of the plotted indicator
# Legend: TRUE/FALSE of plotted legend
# placeLegend - where the legend should be plotted
# show.axes: TRUE/FALSE of plotted axes
# x.lab: lab of x-axis
# y.lab: lab of y-axis
# sparkline: TRUE/FALSE - prepares the graphic for later
#         use as a sparkline
# regression: none, LM or LOESS
# plot.title(): add additional title settings

{
   if(!missing(x0) | !missing(x1))
   {
      regression=”LM”
      print(”regression will only be needed if x0 and x1 is missing”)
   }
   if (missing(x)) stop(”pass parameter x”)

   if (is.ts(x))
   {
      x_time <- as.numeric(time(x))
      x <- as.numeric(x)
   }
   if(missing(t0))    t0 <- x_time[1]
   if(missing(t1))    t1 <- x_time[length(x_time)]
   if( length(x) != length(x_time)) stop(”length of x and x_time is
       different”)
   nobs <- length(x)
   if (length(sderr)==1) sderr <- rep(sderr, nobs)
   if (length(sderr) != length(x)) stop(”length of x and sderr is
       different”)
   dr <- abs(dr)
```

```
if(missing(betax) & t1 != t0 & missing(x0) & missing(x1))
{
   if(regression == "LM")
   {
      betax<-coefficients(lmrob(x~x_time))[2]  # slope of LM (change
          to RLM)
      x0 <- coefficients(lmrob(x~x_time))[1] + betax*t0  # intercept
          of LM
   }
   if(regression == "LOESS")
   {
      zloess<-loess(x~x_time)
      z.predict <- predict(zloess, data.frame(x_time=x_time))
      target.path <- z.predict
   }
   if(regression == "none")
   {
      betax <- (x[nobs]-x[1])/(x_time[nobs]-x_time[1])
      x0=x[1]-betax*x_time[1] + betax*t0
   }

}
if (missing(betax) & (!missing(x0) | !missing(x1)))
   betax <- (x1-x0)/(t1-t0) # try to calculate betax


# evaluation
if(regression=="LM" | regression=="none") target.path <- x0 + (x_time
    - t0) * betax
evaluation <- ifelse(x-target.path > dr+2*sderr,1,
         ifelse(x-target.path < (-dr-2*sderr),-1,0))
if(good.ind=="low") evaluation <- evaluation*(-1)
if(good.ind=="infunnel")
   evaluation <- ifelse(abs(x-target.path) > abs(dr+2*sderr),-1,
            ifelse(abs(x-target.path) > abs(dr)
            & abs(x-target.path) < abs(dr+2*sderr),0,1))
if(good.ind=="outfunnel")
   evaluation <- ifelse(abs(x-target.path) > (dr+2*sderr),1,
            ifelse(abs(x-target.path) > abs(dr)
            & abs(x-target.path) < abs(dr+2*sderr),0,-1))
```

```
xrange<-c(min(x_time,t0),max(x_time,t1))
yrange<-c(min(x-sderr,x0,x1),max(x+sderr,x0,x1))

# plotting of plot, title, legend
plot(x~x_time,type=”l”,
      xlim=xrange, ylim=yrange, axes=show.axes,xlab=””,ylab=””,
      par(parList))
title(plot.title)

if (Legend){
   legendText <- c(”course”,”significant change”,”relevant change”)
   (dr+2sderr) cols <- c(”black”, ”blue”, ”blue”)
   ltys <- c(1,3,2)
   legend(placeLegend, col=cols, lty=ltys, legend=legendText)
}
# x-values for target path
tx<-xrange[1]:xrange[2]

if(regression==”LM” | regression==”none” | (regression==”LOESS” &
!missing(betax) & t1 != t0 & !missing(x0) & !missing(x1))) { long.
    target.path<-x0+(tx-t0)*betax
   lines(long.target.path~tx,col=”black”)
}
if(regression==”LOESS”)
{
   zloess<-loess(x~x_time,control=loess.control(surface=”direct”))
   z.predict<-predict(zloess,data.frame(x_time=c(xrange[1]:xrange[2])
      ))
   long.target.path<-z.predict
   lines(z.predict~seq(t0,t1,1),col=”green”)
   betax=NULL
}

# error lines dr around target.path
lines((dr+long.target.path)~tx,col=”blue”,lty=2)
lines((-dr+long.target.path)~tx,col=”blue”,lty=2)
# error lines sderr around target path (first pad sderr)
if (sparkline==FALSE)
{
   sderrL <- sderr
```

```
        if (min(x_time)>min(tx)) sderrL<-c(rep(0,min(x_time)-min(tx)),
            sderrL)
        if (max(tx)>max(x_time)) sderrL<-c(sderrL,rep(sderrL[length(sderrL
            )],max(tx)-max(x_time)))

        lines((dr+2*sderrL+long.target.path)~tx,col="blue",lty=3)
        lines((-dr-2*sderrL+long.target.path)~tx,col="blue",lty=3)
    }
    #limits (dr+2sderr): dotted")


    if (eval.labels)
        points(x~x_time,col=c("red","gold","green")[evaluation+2],
                cex=ind.size)

    segments(x0=x_time, y0=x-sderr,x1=x_time,y1=x+sderr, col=c("red","
        gold","green")[evaluation+2], lty=3)
    points(x+sderr~x_time,col=c("red","gold","green")[evaluation+2],
            cex=ind.size, pch=20)
    points(x-sderr~x_time,col=c("red","gold","green")[evaluation+2],
            cex=ind.size, pch=20)

    # output
    list(indicator=x,time=x_time,evaluation=evaluation,
            sderr=sderr,dr=dr,x0=x0,t0=t0,x1=x1,t1=t1,betax=betax,
            eval.labels=eval.labels)

}
```

A.1.2 indTable function

The indTable function generates tables like Table 3.2.

```
library(TeachingDemos)
library(xtable)
library(robustbase)

indTable <- function(data,
plottype="normal",region="country",output="plot",columns=c("min","max","
    current","sparkline")) {
```

```
datadim <- dim(data)[1]
time <- dim(data)[2]
plotscale=2
yrange <- datadim/plotscale
xrange <- 8
posRegion <- 0
posSpark <- 2
posMin <- 5
posMax <- 6
posCur <- 7

if(output==”plot”)
{
  #X11(height=yrange,width=xrange,xpos=800,ypos=0,title=””)
  plot(0,xlim=c(-plotscale/8,xrange),ylim=c(-plotscale/8,yrange+
     plotscale/8),cex=0,axes=FALSE,xlab=””,ylab=””,mar=c(0,0,0,0),
     omi=c(0,0,0,0))

  text(c(posRegion+.75,posSpark+1,posMin,posMax,posCur),y=rep(yrange
     ,5),c(region,”sparkline”,”min”,”max”,”curent”))

  for (i in 1:datadim)
  {
    ypos <- yrange-i/plotscale
    text(posRegion,ypos,rownames(data)[i],pos=4)
    if(plottype==”normal”)
      subplot(plot(as.numeric(data[i,]),axes=FALSE,xlab=””,ylab
         =””,type=”o”),x=posSpark,y=ypos,hadj=0,size=c(1.5,0.25))
    if(plottype==”indEval”)
      subplot(indEval(as.numeric(data[i,]), good.ind=”low”,
      parList=list(pch=19,cex.lab=0.75,cex=0.75), show.axes=FALSE)
         ,x=posSpark,y=ypos,hadj=0,size=c(1.5,0.23))
    text(posMin,ypos,format(min(data[i,]),digits=2,nsmall=2))
    text(posMax,ypos,format(max(data[i,]),digits=2,nsmall=2))
    text(posCur,ypos,format(data[i,length(data[i,])],digits=2,
       nsmall=2))
    lines(x=c(0,xrange),y=rep(ypos+plotscale/8,2))

  }
  rect(xleft=0,xright=xrange,ybottom=0-plotscale/8,ytop=yrange+
     plotscale/8)
```

```
}

if(output==”latex”)
{
    dir.create(”Figures”)
    sparkplot <- function(datats,plottype)
    {
        pdf(paste(”Figures/spark_”,rownames(datats),”.pdf”,sep=””),
             width=20, height=5)
        if(plottype==”normal”)
            plot(as.numeric(datats),axes=FALSE,xlab=””,ylab=””,type=”o”)
        if(plottype==”indEval”)
            indEval(as.numeric(datats), good.ind=”low”,
            parList=list(pch=19,cex.lab=1,cex=1), show.axes=FALSE) dev.
                off()
    }
    data[,time+1]<-apply(data,1,min)
    data[,time+2]<-apply(data,1,max)
    data[,time+3]<-data[time]
    data[,time+4]<-apply(data,1,median)
    data[,time+5]<-apply(data,1,mean)
    trend <- function(data)
    {
        betax<-rep(0,dim(data)[1])
        for (i in 1:dim(data)[1])
        {
            lmmod <- lmrob(as.numeric(data[i,])~time(as.numeric(data[i
                ,])))
            betax[i]<-coefficients(lmmod)[2]
        }
        betax
    }
    data[,time+6]<-trend(data)

for(i in 1:dim(data)[1])
{
    sparkplot(data[i,1:time],plottype=plottype)
    a<-'\\raisebox{-.6mm}{\\includegraphics[height=1em,    width=6em]'
    b<-paste(”{Figures/spark_”,rownames(data[i,]),”.pdf}}”,sep=””)
    data[i,time+7]<-paste(a,b,sep=””)
    }

    colnames(data)<-c(colnames(data)[1:time],"min","max","current","
        median","mean","trend","sparkline")
    datanew <- data[(time+1):(dim(data)[2])]
    print(xtable(datanew[columns]), sanitize.text.function=I)

    }
}
```

A.2 Map related Functions

A.2.1 giniVis function

The `giniVis` function presents an interactive map like in Figure 4.5(a).

```
library(mvtnorm)
library(rgdal)
library(laeken)
library(plotrix)
library(colorspace)
library(TeachingDemos)
library(pixmap)

data(mapNUTS)

giniVis <- function (map,  year=2004, regression="none")
{

    load("countrycoords.RDATA",.GlobalEnv)
    #load("euGini.RDATA",.GlobalEnv)
    load("gc_nuts0.RDATA")
    load("gc_plotCol.RDATA")    # colors
    EU<-  c("IS","FI","NO","EE","LV","SE","DK","LT","IE","NL","UK","LU",
           "PL","BE","CZ","SK","DE","LI","AT","HU","CH","SI","RO","HR",
           "BG","TR","IT","MT","CY","GR","PT","ES","FR")

    all.countries <- c("AT","BE","CY","CZ","DE","DK","EE","ES","FI","FR
        ","GR","HU","IE","IS","IT","LT","LU","LV","NL","NO","PL","PT","SE
        ","SI","SK","UK")
```

```
corColorHCL <- rev(heat.colors(201,alpha=0.75))

locatorVIM <- function(error = FALSE)
{
    pt <- try(locator(1), silent=TRUE)
    if(class(pt) == "try-error" && !error) pt <- NULL
    pt
}

eumap <- subsetNUTS(map,'',0)
proj4string(eumap) <- CRS("+proj=longlat")
eumap@bbox[1,1] = -27    # changed map boarders
eumap@bbox[1,2] = 50
eumap@bbox[2,1] = 25
eumap@bbox[2,2] = 70

while(1)
{
    plot(eumap,col=plotCol[,as.character(year)])#, col=corColor[runif
        (33,150,180)])
    gradient.rect(xleft=42,ybottom=45,xright=45,ytop=65,col=
        corColorHCL,gradient="y")
    for (i in 0:10)
        text(48,45+2*i,format((20+2*i),nsmall=1))

    title(paste("Gini coefficients in",year))
    for (i in 1:length(gc_nuts0))
    {
        gc <- gc_nuts0[[i]]
        an <- all.countries[i]
        coords <- a[all.countries[i],]
        subplot(plot(gc,xlim=c(1,4),ylim=c(22,38),pch=19,cex=0.2,lwd=2,
            type="l",col=c("blue"),xlab="",ylab="",axes=FALSE),
               x=coords[1],y=coords[2],size=c(0.25,0.15))
    #   subplot(plot(gc,xlim=c(1,4),ylim=c(22,38),pch=19,cex=0.2,lwd=2,
        type="p",col=c("darkblue"),xlab="",ylab="",axes=FALSE),
    #          x=coords[1],y=coords[2],size=c(0.25,0.15))

    }
```

```
gradient.rect(xleft=-30,ybottom=23,xright=-20,ytop=25,col=”
    lightblue”,gradient=”y”)
text(-25,24,”save plot”)
if(year>2004)
{
   gradient.rect(xleft=-16,ybottom=70,xright=-8,ytop=72,col=”
       lightblue”,gradient=”y”)
   text(-12,71,paste(”to”,as.numeric(year)-1,sep=” ”))
}
if(year<2007)
{
   gradient.rect(xleft=35,ybottom=70,xright=43,ytop=72,col=”
       lightblue”,gradient=”y”)
   text(39,71,paste(”to”,as.numeric(year)+1,sep=” ”))
}
pt<- locatorVIM()
con=TRUE
while((!(pt$x>-30 & pt$x < -20 & pt$y>23 & pt$y<25) |
       !((pt$x>-16 & pt$x < -8 & pt$y>70 & pt$y<72) & (year > 2004
           )) |
       !((pt$x>35 & pt$x < 43 & pt$y>70 & pt$y<72) & (year < 2007))
           ) & con==TRUE)
{
   subId<-overlay(SpatialPoints(cbind(pt$x,pt$y)),eumap)
   if(any(EU[subId]==all.countries) && (pt$x>-27 & pt$x < 50 &
       pt$y>25 & pt$y<70) && !is.na(subId))
   {
       if(length(dev.list())==2)
          dev.off(dev.list()[2])
       print(EU[subId])
       print(gc_nuts0[[which(EU[subId]==all.countries)]])
       X11(height=5,width=8,xpos=800,ypos=0,title=EU[subId])
       gini_ts <- ts(gc_nuts0[[which(EU[subId]==all.countries)]],
           start=as.numeric(names(gc_nuts0[[which(EU[subId]==all.
           countries)]])[1]))
       indev(gini_ts, sder=0.25, regression=regression, dr=0.5,
             #gc_nuts0[[which(EU[subId]==all.countries)]],
             good.ind=”low”,    plot.title=title(xlab=”year”,ylab=”
                 Gini”,main=paste(”Gini coefficient of”,EU[subId],
                 sep=” ”)),
                        parList=list(bg="white",pch=19,cex.lab=1,cex=1),show.
                            axes=FALSE)
                axis(1, at=start(gini_ts):end(gini_ts), lab=c(start(gini_ts)
                    :end(gini_ts)))
                axis(2)#, at=(min(gini_ts)-sd(gini_ts)):(max(gini_ts)+sd(s))
                    ,lab=(min(gini_ts)-sd(gini_ts)):(max(gini_ts)+sd(s)))

                dev.set(dev.list()[1])

            }

        if((pt$x>-30 & pt$x < -20 & pt$y>23 & pt$y<25))
        {
            savePlot(filename = paste(paste("gini",year,sep="_"),"png",sep
                =".") , type="png", device = dev.cur())
            print(paste("Plot saved as ",paste(paste("gini",year,sep="_"),"
                png",sep="."),sep=""))
            con=TRUE
        }
        if((pt$x>-16 & pt$x < -8 & pt$y>70 & pt$y<72) & (year > 2004 ))
        {
            year <- year - 1
            con=FALSE
        }
        if((pt$x>35 & pt$x < 43 & pt$y>70 & pt$y<72) & (year < 2007))
        {
            year <- year + 1
            con=FALSE
        }
        if(con==TRUE)
        {
            pt<- locatorVIM()
        }
    }
    }

}
```

A.2.2 mapCor function

This function visualizes correlations in a map of Europe. One possible plot can be seen in Figure 4.9

```
mapCor <- function (map, cCor, chart="correlation", colScheme="HCL")

{

  if(chart=="distance" & min(cCor) < 0 )
  {
    cCor <- abs(cCor)
    warning("chart=distance requires non-negative numbers, numbers
        changed to absolute values")
  }

  corColorHCL2 <- rev(diverge_hcl(201, h = c(246, 40), c = 96, l = c
      (65, 90)))
  corColorRGB <- rainbow(201, s = 1, v = 0.9, start = 0, end = 2/6,
      gamma = 0.8, alpha = 0.8)
  corColorHCL <- rev(diverge_hcl(201, h = c(130, 43), c = 100, l = c
      (70, 90)))
  corColorUser <- corColorHCL
  if(length(colScheme) != 201 && (colScheme != "HCL" && colScheme !="
      RGB"))
  {
    print("please use a Color Scheme with 201 colors")
  }
  if(length(colScheme) == 201)
  {
    corColorUser <- colScheme
    corColor <- corColorUser
  }
  if(colScheme=="HCL") corColor <- corColorHCL
  if(colScheme=="RGB") corColor <- corColorRGB

  eumap <- subsetNUTS(map,'',0)
  proj4string(eumap) <- CRS("+proj=longlat")
  eumap@bbox[1,1] = -27   # changed map boarders
  eumap@bbox[1,2] = 50
  eumap@bbox[2,1] = 25
```

```
eumap@bbox[2,2] = 70

plot(eumap, col=corColorRGB[runif(33,150,180)])
title("European Union",sub="please select country")

locatorVIM <- function(error = FALSE)
{
    pt <- try(locator(1), silent=TRUE)
    if(class(pt) == "try-error" && !error) pt <- NULL
    pt
}

while(1)
{

    pt<- locatorVIM()
    print(pt)

    subId<-overlay(SpatialPoints(cbind(pt$x,pt$y)),map)

    if(!is.na(subId))
    {
        xcor <- cCor[subId,]
        if(chart=="correlation")
        {
            plotCol <- round((xcor+1)*100)
            plotCol <- corColor[plotCol+1]
        }
        if(chart=="minmax" | chart=="distance")
        {
            minCor <- min(xcor)
            if(chart=="distance")
                minCor <- 0
            maxCor <- max(xcor[-subId])
            range <- maxCor - minCor
            #xcor - minCor
            plotCol <- round((xcor-minCor)*200/range)
            plotCol <- corColor[plotCol+1]
        }
        subId<-map$NUTS_ID[subId]
        subIdP<-subId
        plot(eumap, col=plotCol)

        title(main="European Union",sub=paste("Correlation between ",
            subId," and other countries",sep=""))
        gradient.rect(xleft=42,ybottom=50,xright=45,ytop=70,col=
            corColor,gradient="y")
        if(chart=="correlation")
        {
            for (i in 0:10)
                text(48,50+2*i,format((2*i-10)/10,nsmall=1))



        }
        if(chart=="minmax" | chart=="distance")
        {
            for (i in 0:10)
                text(49,50+2*i,format(round(minCor+i*range/10,2),nsmall
                    =2))
            #print(format(minCor+i*range/10,nsmall=1))
        }
        gradient.rect(xleft=-30,ybottom=23,xright=-20,ytop=25,col="
            lightblue",gradient="y")
        text(-25,24,"save plot")


    }

    if((pt$x>42 & pt$x<45 & pt$y>50 & pt$y<70))
    {
        colChange=FALSE
        if(identical(corColor,corColorRGB) & colChange==FALSE)
        {
            corColorN<-corColorHCL
            colChange=TRUE
            print("Color Scheme changed to HCL")
        }
        if(identical(corColor,corColorHCL) & colChange==FALSE)
        {
            corColorN<-corColorHCL2
            colChange=TRUE
            print("Color Scheme changed to HCL2")
         }
         if(identical(corColor,corColorHCL2) & colChange==FALSE)
         {
            corColorN<-corColorUser
            colChange=TRUE
            print("Color Scheme changed to User specified Color Scheme")
         }
         if(identical(corColor,corColorUser) & colChange==FALSE)
         {
            corColorN<-corColorRGB
            colChange=TRUE
            print("Color Scheme changed to RGB")
         }
         corColor<-corColorN
      }
      if((pt$x>-32 & pt$x < -20 & pt$y>22 & pt$y<25))
      {
         savePlot(filename = paste(subIdP,"png",sep="."),   type="png",
            device = dev.cur())
         print(paste("Plot saved as ",paste(subIdP,"png",sep="."),sep
            =""))
      }
   }

} # end mapCor
```

A.3 Miscellaneous Functions

A.3.1 indweather function

The weather indicators, presenting an outlook of the development of variouse indicators. One example is given in Figure 3.11.

```
weather_ind <- function (matr)
{
   rows <- nrow(matr)
   cols <- ncol(matr)

      indw <- list(rep(0,rows*cols))
```

```
    k<-1
    for (i in 1:(rows))
    {

        for (j in 1:(cols))
        {

            ind<-paste("weather0",matr[i,j],sep="")
            file <- paste(ind,"ppm",sep=".")
            indw[k]  <- read.pnm(file,bbox=c((j-1)*200+20, (rows-i)
                *200+20, j*200-20, (rows-i+1)*200-20))
            if(i==1 && j==1)
            {
                plot(indw[[1]],xlim=c(-500,cols*200),ylim=c(0,rows*200),
                    axes=FALSE,cex=3)
            }
            text(100+200*(j-1),200*rows+20,colnames(matr)[j],srt=90,pos
                =4) # x-axis names
            text(0,100+(i-1)*200,rownames(matr)[i],pos=2)        # y-axis
                names

            k<-k+1
        }

    }

    for (i in 2:(rows*cols))
        plot(indw[[i]],add=TRUE)

    title(main="Indicator forecast")

}
```

A.3.2 indTableCI function

This function is used to present confidence intervals in tables, presenting special sparklines.

```
library(TeachingDemos)
library(xtable)
library(robustbase)
library(plotrix)



indTableCI <− function(data, plottype=”line”,scaleRange=c(20,40))
{
   years <− data$years
   country <− substr(data$strata[1],1,2)
   countryInd <− data$value
   dir.create(”FiguresCI”)
   dataLatex <−matrix(nrow=length(data$years)∗(length(data$strata)+1),
       ncol=6)
   colnames(dataLatex) <− c(”region”,”year”,”ind”,”lower”,”upper”,”
       sparkline”)
   k=0
   for (i in as.character(years))
   {
      data$ci[i,]
      indCI <− as.numeric(round(c(countryInd[i], data$ci[i,]),2))

      pdf(paste(”FiguresCI/ind_”,country,”−”,i,”.pdf”,sep=””),width=20,
          height=5)
         indScale(ind=indCI, type=plottype, scaleRange=scaleRange,
             fillColor=”lightgrey”)
      dev.off()
      k<−k+1
      a<−'\\raisebox{−.6mm}{\\includegraphics[height=1em,     width=6em]'
      b<−paste(”{FiguresCI/ind_”,country,”−”,i,”.pdf}}”,sep=””)
      dataLatex[k,] <− c(country,as.numeric(i),indCI, paste(a,b,sep=””))


      strataVal <− data$valueByStratum[data$valueByStratum==i,]
      strataCI <−  data$ciByStratum[data$ciByStratum$year==i,]

      for(j in data$strata)
      {
         indStrataCI <− c(strataVal[strataVal$stratum==j,]$value,
               strataCI[strataCI$stratum==j,]$lower,
               strataCI[strataCI$stratum==j,]$upper)
      indStrataCI <- round(indStrataCI,2)
      k<-k+1
      a<-'\\raisebox{-.6mm}{\\includegraphics[height=1em,     width=6em
        ]'
      b<-paste(”{FiguresCI/ind_”,j,”-”,i,”.pdf}}”,sep=””)
      dataLatex[k,] <- c(j,as.numeric(i),indStrataCI, paste(a,b,sep
        =””))
      pdf(paste(”FiguresCI/ind_”,j,”-”,i,”.pdf”,sep=””),width=20,
        height=5)
        indScale(ind=indStrataCI, type=plottype, scaleRange=
          scaleRange, fillColor=”lightgrey”, showValues=FALSE)
      dev.off()

    }

  }
  print(xtable(dataLatex), sanitize.text.function=I)

}
```

The indTableCI function uses the indScale function for the creation of the sparklines:

```
indScale <- function(ind, scaleColor=rev(heat.colors(201,alpha=0.75)),
    scaleRange = c(0,100), scaleLabs = 5, type=”line”, fillColor = NA,
       showValues=TRUE)
{

  if(length(ind)==1)
    ind <- rep(ind,3)
  if(length(ind)==2)
    ind <- c(ind[1], ind[1] - ind[2], ind[1] + ind[2])

plot(0,type=”n”,xlim=c(scaleRange[1]-5,scaleRange[2]+5), ylim=c(0,30),
  axes=FALSE, xlab=””, ylab=””)

if(type==”line”) gradient.rect(xleft=ind[2],ybottom=4, xright=ind[3],
  ytop=26, col=fillColor, border=fillColor)

if(type==”line2”)
{
  gradient.rect(xleft=ind[2],ybottom=4, xright=ind[3], ytop=5, col=
    fillColor, border=fillColor)
    gradient.rect(xleft=ind[2],ybottom=25, xright=ind[3], ytop=26, col=
        fillColor, border=fillColor)
}

if(type==”tri”) polygon(ind, c(15, 3, 3), col=fillColor, lty=c(2,1,2))

gradient.rect(xleft=scaleRange[1],ybottom=5,xright=scaleRange[2],ytop
    =25,col=scaleColor,gradient=”x”)

if(showValues==TRUE)text(ind[2:3],1,ind[2:3])

for (i in 0:scaleLabs)
{
    text(scaleRange[1]+i*(scaleRange[2]-scaleRange[1])/scaleLabs,27,
        format((scaleRange[1]+i*(scaleRange[2]-scaleRange[1])/scaleLabs),
        nsmall=0))
    text(scaleRange[1]+i*(scaleRange[2]-scaleRange[1])/scaleLabs,25,”I”)
}

if(type==”line” | type==”line2”){
    lines(c(ind[1],ind[1]),c(4,26),lwd=3)
    lines(c(ind[2],ind[2]),c(4,26),lwd=2, lty=2)
    lines(c(ind[3],ind[3]),c(4,26),lwd=2, lty=2)

    if(showValues==TRUE)text(ind[1],29,ind[1])
}

if(type==”tri”)
{
    if(showValues==TRUE)text(ind[1],16,ind[1])
    polygon(ind, c(15, 3, 3), lty=c(2,1,2))
}

}
```

List of Tables

List of Figures

Bibliography

European Environment Agency. Towards environmental pressure indicators for the EU. Technical report, European Commission, Office for Official Publications of the European Communities, Luxembourg, 1999. URL `esl.jrc.it/envind/tepi99rp.pdf`. [Online; accessed November-2009].

A. Alfons, P. Filzmoser, B. Hulliger, B. Meindl, T. Schoch, and M. Templ. State-of-the-art in visualization of indicators in survey statistics. Technical Report CS-2009-4, Vienna University of Technology, 2009.

A. Alfons, J. Holzer, and M. Templ. *laeken: Laeken indicators for measuring social cohesion*, 2010. R package version 0.1.1.

G.E.P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. John Wiley Sons Inc, New Jersey, 2008.

P. J. Brockwell and R. A. Davis. *Introduction to Time Series and Forecasting*. Springer, New York, 1996.

J. M. Chambers. *Software for Data Analysis: Programming with R*. Springer, New York, 2008.

W. S. Cleveland. *Visualizing Data*. AT&T Bell Labs, New Jersey, 1993.

W. S. Cleveland. Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, 74:829–836, 1979.

Commission Internationale de L'Eclairage. *Colorimetry*. CIE, Vienna, 2004.

B. Efron and R. Tibshirani. Bootstrap Methods for Standard Errors, Confidence Intervals, and Other Measures of Statistical Accuracy. *Statistical Science*, 1:54–75, 1986.

EU-SILC. Algorithms to compute Social Inclusion Indicators based on EU-SILC and adopted under the Open Method of Coordination (OMC). EU-SILC LC-ILC/39/09/EN-rev.1, Directorate F: Social and information society statistics Unit F-3: Living conditions and social protection, EUROPEAN COMMISSION, EUROSTAT, Luxembourg, 2009.

European Environment Agency. The European environment - State and outlook 2005. Technical report, BFS/BUWAL/ARE, 2005.

Comparative EU statistics on Income and Living Conditions: Issues and Challenges, Methodologies and working papers, Luxembourg, 2007. EUROSTAT.

EUROSTAT. 'laeken' indicators - detailed calculation methodology. Technical Report E2/IPSE/2003, Working group 'Statistics on income, poverty & social exclusion', 2003.

P. Gunawardane, J. Feng, S. K. Lodha, B. Crow, B. Fulfrost, and J. Davis. Visualizing Relationships between Global Indicators. Technical report, UC Santa Cruz: Center for Global, International and Regional Studies., 2007.

M. Harrower and C. A. Brewer. ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. *The Cartographic Journal*, 40:27–37, 2003.

A. F. Hayes, M. Slater, and L. B. Snyder. *The SAGE Sourcebook of Advanced Data Analysis Methods for Communication Research*. Sage Publications, Thousand Oaks, California, 2007.

K. Horsch. Indicators: Definition and Use in a Results-Based Accountability System, 1997. URL `http://www.hfrp.org/publications-resources/browse-our-publications/indicators-definition-and-use-in-a-results-based-accountability-system`. [Online; accessed December-2009].

B. Hulliger and D. Lussmann. Bewertung der Nachhaltigkeits- und Umwelt-Indikatoren. Technical report, Institute for Competitiveness and Communication, University of Applied Sciences, Northwestern Switzerland, 2008.

B. Hulliger and R. Münnich. Variance estimation for complex surveys in the presence of outliers. In *Proceedings of the Section on Survey Research Methods*, pages 3153–3161. American Statistical Association, 2006.

R. Ihaka, P. Murrell, K. Hornik, and A. Zeileis. *colorspace: Color Space Manipulation*, 2009. URL `http://CRAN.R-project.org/package=colorspace`. R package version 1.0-1.

R. Juckett. RGB Color Space Conversion , October 2010. URL `http://www.ryanjuckett.com`. [Online; accessed August-2010].

A. Kowarik, B. Meindl, and S. Zechner. *sparkTable: Sparklines and graphical tables for tex and html*, 2010. URL `http://CRAN.R-project.org/package=sparkTable`. R package version 0.1.1.

C. Kung-Sik. *TSA: Time Series Analysis*, 2010. URL `http://CRAN.R-project.org/package=TSA`. R package version 0.98.

R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2010. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

G. Snow. *TeachingDemos: Demonstrations for teaching and learning*, 2009. URL `http://CRAN.R-project.org/package=TeachingDemos`. R package version 2.5.

H. K. Timothy, R. Bivand, E. Pebesma, and B. Rowlingson. *rgdal: Bindings for the Geospatial Data Abstraction Library*, 2010. URL `http://CRAN.R-project.org/package=rgdal`. R package version 0.6-25.

E. Tufte. *Beautiful Evidence.* Graphics Press, Cheshire, 2006.

W. W. S. Wei. *Time series analysis, Univariate and Multivariate Methods.* Addison-Wesley Publishing Company, 1990.

Wikipedia. Lab color space — wikipedia, the free encyclopedia, 2010. URL `http://en.wikipedia.org/w/index.php?title=Lab_color_space&oldid=367435096`. [Online; accessed 2-August-2010].

V.J. Yohai. High breakdown-point and high efficiency estimates for regresssion. *The Annals of Statistics*, 15:642–656, 1987.

A. Zeileis, K. Hornik, and P. Murrell. Escaping RGBland: Selecting colors for statistical graphics. *Computational Statistics Data Analysis*, 53:3259–3270, 2009.