Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institute for Dynamic Systems and Control
IDSC
Institut für Dynamische Systeme und Regelungstechnik

Andreas Ritter, Dr. Philipp Elbert, Prof. Dr. Christopher Onder

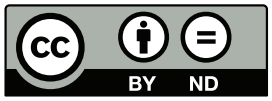
How to Use the IDSCreport LaTeX Class

Version 1.6.0

Technical Report

Institute for Dynamic Systems and Control
Swiss Federal Institute of Technology (ETH) Zurich

December 2018



Abstract

One important aspect of scientific research is the documentation of its results. In order to conserve newly found knowledge and ease the access to results, every researcher needs to acquire the ability to prepare reports in compliance with scientific standards. Only in this way, the new findings can be conserved and be communicated to supervisors, successors, and the scientific community. However, the task of preparing a high quality scientific report can represent a difficulty–especially for undergraduate students conducting their first research projects without having any experience to rely on, but also for graduate students who are beginning their scientific careers as PhD students. In order to facilitate the process of preparing high quality scientific reports, the Institute for Dynamic Systems and Control (IDSC) provides two items aiming to help young students and researchers in the process of learning the basics of scientific writing:

The first item is the *IDSCreport LaTeX Class*, which can serve as a template in the preparation of a report. The use of the IDSCreport template is mandatory for all student projects conducted under the supervision of Prof. Dr. Christopher Onder, such as Studies on Mechatronics, Bachelor's theses, semester projects, Master's theses and internships. This standardization aims at ensuring that all reports produced at IDSC share identical formatting, as well as that the documented results from different projects are compatible. The students benefit from the availability of a starting point when beginning to learn the LaTeX typesetting system.

The second item is this governing document, which is divided in three parts. After a general motivation, the first part provides a number of tips to consider when writing the text of a report. The second part introduces the functionalities of the IDSCreport LaTeX Class, while the third part covers the IDSC guidelines to consider when programming the typesetting of all items that are not regular text, such as figures, tables, citations, etc. These guidelines define the mandatory standards for all student reports of projects conducted under the supervision of Prof. Onder. The students benefit from a large knowledge base where they can quickly access a lot of information in case of doubt: What are the scientific standards? What is considered the most appropriate implementation? And where can further literature be found?

By providing these two items to our students, we hope to guide young researchers in taking their first steps of their career, maintain the quality of scientific reports written at IDSC at a high level with minimum effort, and keep students from struggling with the difficult task of preparing good reports.

Keywords: LaTeX, Template, Student projects, Institute for Dynamic Systems and Control, ETH Zurich.

Contents

Preface

This template can be seen as the third major revision of the original style file `ethimrt.sty` written by Eric A. Müller in 2003. In 2009, Soren Ebbesen adopted the style file and renamed it `ethidsc.sty` in accordance with the name change of the institute. Soren maintained the style file until he left the research group in 2014. When Prof. Dr. Lino Guzzella became President of ETH Zurich on January 1, 2015, Dr. Christopher Onder, formerly a senior scientist, was appointed professor and continued the research in automotive control and optimization with his own research group.

Having the honor of being one of his first doctoral students, I started rethinking the challenging tasks of supervising student projects and how the whole process could be standardized and therefore be simplified and streamlined. With this goal in mind, I started rewriting the LaTeX template for student reports and converted it from a style file to a class definition. In addition, I created this how-to guide in order to not only give students more information on the LaTeX-related instructions used by the template, but to provide guidelines for writing reports in accordance with scientific standards as well.

Much of my view on supervising student projects was inspired by my former supervisor, Dr. Philipp Elbert. On that account, I hereby express my sincere gratitude to him for teaching me how to improve my writing style and my presentation skills. Philipp was also involved in the development and revision of this text. Finally, I want to thank Severin Hänggi and other colleges of Prof. Onder's research group for helping me with the implementation of the LaTeX document class.

Zurich, December 2016 *Andreas Ritter*

Nomenclature

Mathematical Symbols

s	Distance variable	m

Subscripts

k	Iteration index

Superscripts

$\star$	Optimal solution

Acronyms and Abbreviations

abbr.	Abbreviation
AMS	American Mathematical Society
AMS	American Mathematical Society
CTAN	Comprehensive TEX Archive Network
DVI	Device-independent file format
EPS	Encapsulated PostScript
GUI	Graphical user interface
IDSC	Institute for Dynamic Systems and Control
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
JPEG	Joint Photographic Experts Group
PDF	Portable document format
PNG	Portable network graphics
SVG	Scalable vector format
URL	Uniform resource locator

WLTC	Worldwide light-duty test cycles
WLTP	Worldwide harmonized Light vehicles Test Procedure

Glossary

glossary	A collection of textual glosses or of specialized terms with their meanings, [1].

Chapter 1

Introduction

This introductory chapter features a structure that is similar to regular technical reports such as theses reports of student projects. The chapter first depicts the initial motivation for the author to write this text. It then defines the objective of the new version of the template and declares the context within which it should be used. Subsequently, the structure of the document is described and the license associated with this template is given.

1.1 Motivation

For many students, the Bachelor's thesis is the first occasion within their career as a student, where they are expected to complete a small research project on their own—starting from scratch and finishing with the final thesis. Of course, having only little experience with the writing of scientific reports, many students repeat mistakes and adopt writing styles that are not compliant with scientific writing. Furthermore, since mathematical expressions, computer code, figures and tables are often part of the report, the use of the LaTeX typesetting system is a convenient choice. However, becoming familiar with the vast amount of functionalities and possibilities of LaTeX can be a burden, especially if the student project is to be finished in due time.

In order to help students in the process of writing their theses, it was decided years ago to prepare a LaTeX template to be used for any type of student project conducted within the Institute for Dynamic Systems and Control (IDSC), ETH Zurich, as well as this guiding document, which provides the students with some basic writing tips, a documentation of the functionalities of the IDSC LaTeX template, as well as some guidelines of how certain LaTeX features are to be used within reports related to the IDSC.

1.2 Objective

With the combination of the IDSC LaTeX template and this guiding document, we hope to achieve the following goals:

1. To guide students in the process of preparing their first, but also their consecutive scientific reports using the LaTeX typsetting system and thereby to streamline and simplify the process for both the students as well as for the supervisors.

2. To ensure that all material produced at the IDSC shares identical formatting and follows the same basic stylistic scheme, which ensures consistent quality and allows employees of the institute to quickly access and reuse the valuable material produced by the student.

1.3 Context

The IDSC LaTeX template should therefore be used for all kinds of generic reports and reports of student projects, including Studies on Mechatronics, Bachelor's theses, semester projects, Master's theses, as well as internships. Furthermore, this document should be considered as *the* guideline whenever possible, and it should be read carefully before starting to work towards a final report. In case of ambiguities, please contact your supervisor and/or the author of this document. Please keep in mind that the template and the guidelines are meant to make a student's daily work more structured and less complex. By this we hope to keep students from getting distracted by too many details about the special features of LaTeX, and therefore allow them to fully focus on their project instead. The guidelines can be seen as a compendium of the most important LaTeX functionalities relevant for our work and therefore can be used as a quick guide to many useful features that may have been unfamiliar to the student.

1.4 Structure of this Document

The remainder of this text is organized in the following way: Chapter 2 introduces the typical structure of a scientific report and summarizes the most important writing tips to keep in mind when preparing a report. It also lists some of the most common mistakes. Chapter 3 explains the functionalities of the IDSC LaTeX template, while chapter 4 sets up guidelines for the use of LaTeX in the context of student reports.

1.5 Licenses and Citation

This document is part of a distribution that includes the following parts, which are covered by different license agreements.

- This document entitled How to Use the IDSCreport LaTeX class is published under the creative commons license Attribution-NoDerivatives 4.0 International [2].
- The code of the LaTeX document class `IDSCreport.cls` is covered by the GNU General Public License [3].
- The PDF and EPS files of the logos of ETH Zurich and IDSC may not be altered in any way. If you intend to use them otherwise than designated by the document class IDSCreport, consult `https://rechtssammlung.sp.ethz.ch/Dokumente/202.4.pdf` for more information.
- Additional LaTeX packages contained in the directory `docstyle/packages` that are specifically for this document class and not available elsewhere (e.g. CTAN) are published under the GNU General Public License [3].
- Bibliography citation style definitions provided by IEEE (IEEEtran). All corresponding files in the directory `docstyle/bibtexstyle` are covered by The LaTeX Project Public License (LPPL 1.3) [4].
- A collection of manuals of LaTeX packages and guides for writing LaTeX documents. These documents are published under various licenses. Please check yourself for each document if you intend to share or modify these documents.
- A template for writing LaTeX documents with the document class `IDSCreport.cls`. The files `report.tex`, `exampleRegularChapter.tex`, and `exampleAppendixChapter.tex` are not covered by any license.

Copyright statements

This document is under the creative commons license, as stated on the copy right page on the verso of the title page. The license Attribution-NoDerivatives 4.0 International allows everyone to copy

and redistribute the material, even commercially. However, appropriate credit has to be given and the modified material may not be distributed. For detailed information, see [2] for a summary and `https://creativecommons.org/licenses/by-nd/4.0/legalcode` for the legal text.

The LaTeX class definition shipped with this document, `IDSCreport.cls`, is covered by the GNU General Public License. Thereby, anyone is allowed to redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

The class definition is distributed mainly for the purpose of writing reports of student projects and theses that are conducted at the Institute for Dynamic Systems and Control (IDSC) at ETH Zurich. However, it may be useful for other purposes as well. In any case it is distributed without any warranty, without even the implied warranty of merchantability or fitness for a particular purpose. See the GNU General Public License included in the package of this file, or `http://www.gnu.org/licenses/` for more details.

Citation

This document is freely available on the IDSC website. Everyone is free to use it also in the context work that is not related to IDSC. Please cite this document [5] in case you decide to use the IDSC LaTeX template or base your work on the code of the document class. The corresponding BibTeX record is given in the following.

```
@manual{IDSCreportClass,
author = {Andreas Ritter and Philipp Elbert and Christopher Onder},
title = {How to Use the {IDSCreport} {\LaTeX{}} Class},
language = {english},
howpublished = {Version 1.6.0},
organization = {Institute for Dynamic Systems and Control ({IDSC})},
address = {ETH Z\"{u}rich, Switzerland},
month = dec,
year = 2018
}
```

Chapter 2

Writing Tips

This chapter contains general hints that should assist the students in conducting any kind of research project. As part of the training plan of ETH Zurich and especially of studies in the Department of Mechanical and Process Engineering, the students have to conduct a Studies on Mechatronics, a Bachelor's thesis, a semester project, and a Master's thesis. This chapter starts by offering some advice that should be considered at the beginning of each student project. It then deals with some aspects of writing scientific reports and describes the proper structure of such documents. The chapter concludes by addressing the choice of the language and how a good writing style is achieved.

2.1 Where Should I Start?

Before starting with the actual work, the most important point in conducting a student project is to understand the assignment. For one project, the task might be clearly defined by the supervisor and thus be given for the student. For another, project the problem definition might be more open and is yet to be defined by either the supervisor or the student him- or herself. In the latter case, it is the responsibility of both the student and the supervisor to agree on a clear assignment in the process of the literature research.

The second step typically is a research study. The goal of that process is to fully understand the given problem and to identify potential solutions that have not been investigated as yet. It is highly recommended to keep track of the relevant sources and the information contained therein. There are many software applications available that assist you in organizing the source information, the associated documents, and your notes. As a member of ETH Zurich, almost every journal paper and conference paper is available for free. Furthermore, there are many books that can be downloaded for free, or they can be borrowed as hardcopies from the ETH Library.

After the literature research, the goal of the thesis should be clear and the actual work can start. During the work process the focus of the project may change, of course as the process of solving the problem always yields a better understanding of the problem itself and thus brings up new questions and also new answers. However, it is important that the goal always remains clearly defined.

When the duration of the project is at half over, student and supervisor should start working towards a thesis statement. This statement should reflect the main idea of your report and summarize what you want to show in your report. Most of the time, this reconsideration of the final goal leads to a slightly different focus on the problem. The remaining time should therefore be used to investigate the unresolved issues that are implied by the thesis statement.

Before starting to write the report, it is highly recommended to create an outline of the content. Similarly to the skeleton of a body, the outline will keep the text together and allows it to function. Thereby, important information is separated from unimportant information, and the contents of every chapter and section are connected, such that a common thread is created through the entire report.

In order to avoid spending time on topics that are not relevant in the final form of the report, it is often clever to start with the part on the results and the discussion. Based on the outcome of that discussion, the contents of the theoretical part of the report should be defined, while irrelevant material that is not necessary for understanding the results may be excluded. Having finalized the main parts of the report, i.e. the theoretical part and the results section, the introduction can be written on the basis of a definite understanding of the upcoming text. Finally, the abstract and the conclusion can be written.

2.2 Structure of a Report

A scientific report not only contains all relevant information, but also follows a well-defined structure. In order to ensure that the reports of all student projects are structured alike, the present LaTeX template is implemented such that the following order of document elements is enforced:

- Title page
- *Copyright page*
- Abstract including keywords
- *Acknowledgment*
- Table of contents
- *Preface*
- *Nomenclature*
- Regular chapters (introduction, theoretical part, results and discussion, conclusion)
- *Appendices (additional material, mathematical derivations, etc.)*
- Bibliography
- Information sheet

The underlined elements are created automatically during the compilation process. The elements written in italics are optional.

In order to ensure the proper document structure, all elements except for the regular chapters and the appendix are either created automatically during compilation or have to be written in a way that is specific for this template. The body of the document, i.e. the regular chapters and the appendices, is defined by the author. More information on how this structure is realized in terms of typesetting with LaTeX is given in section 3.5. The following sections describe what the individual elements should comprise.

2.2.1 Mandatory Elements

The body of a report of a student project typically contains an introduction, a main part describing the theoretical investigations, a second main part that discusses the results, and a conclusion. This body is framed by a preceding abstract and the table of content and a succeeding bibliography.

Abstract

The abstract is a brief summary of the thesis and is used to help the reader to quickly capture the content of the text. A good abstract contains between ten and twenty sentences, where on average the first third of the abstract is used to describe the broad spectrum of the problem and introduces the reader to the more specific issue that is addressed in the thesis. The second third sketches the methods that are used in order to tackle the given problem, and the remainder of the abstract discusses the results and their contribution and limitations.

Introduction

Although the introduction does not have to follow any given guidelines, it typically contains the following sections:

- Motivation,
- Objective,
- Context,
- State of research, and
- Structure of the report.

The motivation reflects the author's interest in the topic in general and in the given problem in particular. The objective declares the goal of the student project and the reason for its necessity. The context describes the overall project to which the content of this report contributes. This section should also focus on the connection between the specific work of the student and the overall project. The state of research reflects the literature research that is conducted by the student at the beginning of the project. It shows the solutions that are available for the given problem or for similar problems and discusses why those solutions are not appropriate or not sufficient. As a result of this section, the reader should have a clear understanding of why the content of the report is important. The structure of the report briefly describes each chapter of the report and explains why they were chosen to appear in this specific order and what the connections are from one chapter to the next.

Theoretical Part

The theoretical part includes a rigorous problem analysis that yields the reasons for the subsequent theoretical investigations and of course the elaboration of the material that is used to understand and solve the problem. This part typically consists of two or three chapters, where each chapter treats a different subject.

Results and Discussions

The theoretical part is followed by a section on results, where the theoretical knowledge acquired is applied to the specific problem or certain use cases. Depending on the topic, the results may comprise measurement data or simulation data or both, in order to demonstrate how well the theory matches the reality.

The author should always comment the results obtained with his or her knowledge of the topic. The explanations should refer to the theory presented in the previous chapters, but they can also incorporate the personal valuation of the results. The goal is to share the knowledge acquired because at the end of the project, when the report is written, the author should be the expert.

Conclusion

The conclusion contains two elements, namely the contributions and an outlook. For certain reports it might be advisable to have a third element in-between that describes the limitations of the subject matter elaborated.

2.2.2 Elements Created Automatically

The title page, the copyright page, the table of contents, the nomenclature, the bibliography, and the information sheet are created automatically. However, since the implementation of the typesetting systems of LaTeX dates back to the 1980s where computers had a limited amount of memory, the process from LaTeX source code to the final PDF document requires several executions of build commands in a specific order. These so-called build recipes can differ from one document to another. Recommendations for this template are given in section 3.8.

The content of the title page is declared using the designated commands listed in section 3.5. The appearance of the the information sheet with or without the declaration of originality is toggled by the layout-related document class options.

The creation of the table of contents is performed by LaTeX internally and requires the document to be compiled multiple times.

The nomenclature is created if at least one abbreviation, symbol, or index is used. The list is sorted automatically and cannot be modified. Its creation requires an additional step in the "build" recipe and adding elements to the nomenclature via the special commands that are described in section 4.10.

The bibliography depends on a source file given within the preamble of the root file (see section 3.5), and on the citations that are used in the text (see section 4.12).

2.2.3 Optional Elements

In addition to the basic structure, the report may contain a copyright page on the verso of the title page, an acknowledgment just after the abstract, or a preface just after the table of contents.

The copyright page contains the copyright notice, publication information, printing history, legal notices, etc. It can also contain a dedication.

The acknowledgment is a written form of expressing the author's gratitude. In scientific research it usually serves to thank the supervisors and supporters for their help and their preliminary work.

The preface is typically used in books, where the author can explain his or her motivation for working on the topic and describe the origination process of the text. If the introductory chapter is written by a different person, it is called "Foreword" and precedes the author's preface. The preface often closes with acknowledgments of those who assisted in the literary work. Therefore, if the text contains a preface, an additional introductory chapter for the acknowledgments is usually omitted.

2.3 Language

Student reports can either be written in English or in German. Although the use of the German language might be preferable for certain industrial partners, we recommend to use English, as it is the main language in scientific research and thus addresses a larger audience. Moreover, writing an English text offers a great opportunity to improve the writing style for subsequent projects and the post-graduate career. Note that ETH Zurich published its own *English Style Guide* [6] to be used in corporate communications.

2.3.1 Technical Words in German

Writing the report in German might be attractive in particular for German native speakers that conduct Bachelor's theses. Although it seems to be a lot of effort to find the correct words to express thoughts and to reason about ideas and results, the benefit from the learning process is not to be underestimated.

Another reason for switching to English is the fact that often students are more familiar with the English version of many technical words and expressions than with the corresponding German version. Finding the German equivalent takes time and effort and sometimes can be the cause of confusions. The use of English expressions in a German text is not recommended, most importantly because it makes the writer look lazy and imprecise. If the report is going to be in German anyway, two sources for translating specific technical expressions are given here: The appendix in Analysis and Synthesis of Single-Input Single-Output Control Systems by L. Guzzella [7] contains a list that can be used to translate technical terms frequently used in control engineering between English

and German. An extensive list of terms used in automotive engineering in English, German, and French can be found in [8]. Both books can be provided by the supervisors.

2.3.2 English Grammar

This section provides useful information about recurring mistakes, differences between British English and American English, and capitalization rules. Appendix A provides additional information and is referenced at the corresponding points in text in the remainder of this section.

Top Ten Grammatical Mistakes

Some grammatical mistakes happen again and again. The following listing is neither derived by counting mistakes nor is the order of particular importance.

1. The words *which* and *that* are used to introduce restrictive relative clauses and nonrestrictive relative clauses. In general, both words can be used to introduce restrictive relative clauses, which contain essential information about the sentence and thus cannot be omitted. For the same reason there is no comma. Nonrestrictive relative clauses cannot be introduced with that. The additional information is not required for the understanding of the sentence. Relative clauses are preceded by a comma [9].
2. *Transition words* such as *therefore*, *however*, *as a result*, etc. at the beginning of a sentence are always followed by a comma [10]. Avoid using them within a sentence.
3. A *compound word* is a combination of two or more words that function as a single unit of meaning [11]. The decision of whether a series of words has to be hyphenated is very tricky and follows quite some rules. One approach to detect a compound word is to check whether the sense of the sentence is ambiguous without the hyphenation, e.g. *high-risk students* clearly has another meaning than *high risk students*. We must also use compound words when we create single adjectives with past participles such as *well-known*, *so-called*, *model-based*, etc. Furthermore, compounds that are created with *-ly* adverbs are never hyphenated. Closed compounds are written as single words, e.g. *footpath*, *railway*, *smalltalk*, etc. They are used when the first part of the word acts only as a modifier of the second part [12]. Appendix A.1 presents a list of common compound words that are often misspelled.
4. Avoid neglecting words in series of similar things, but rather write out the complete expression multiple times. It is clearer to write “the heat equation and the mass equation” than “the heat and mass equations”. Likewise, avoid suspended hyphens.
5. The expression “yield to” does not exist. Either use “leads to sth.” or “yields sth.” Similarly, the expression “allow to” cannot exist without a noun, see [13].
6. Use a comma in a series of items, even before the last item, which is usually preceded by *and* or *or* and is called a serial comma [14]. In addition, use a comma after certain introductory words or terms, such as *namely*, *that is*, *i.e.*, *e.g.*, and *for instance*, when they are followed by a series of items [15].
7. Place a comma to separate sentences with two independent clauses that are joined by connectors such as *and*, *or*, *but*, etc. If the subject is neglected in the second clause, a comma is usually not required [15].
8. When starting a sentence with a *dependent clause*, a comma is required to separate it from the second clause. When the sentence starts with an independent clause followed by a dependent clause, a comma is often unnecessary. For more information, see [16] and [15].
9. The abbreviations *i.e.* and *e.g.* have different rules for using commas. *E.g.* (abbreviation for *exempli gratia*, which is Latin and means “for example”) should generally be followed by a list of examples. Therefore, *e.g.* usually requires a subsequent comma to delimit the beginning of that list. *I.e.* (abbreviation for *id est*, which is Latin and means “that is”) is used to recapture

the meaning of an antecedent clause by rephrasing. Typically, it is only followed by a clause describing a singular entity, and so does not require a subsequent comma [17]. Do not use them at the beginning of a sentence.

10. Adverbs and adverb phrases can be put at the front, the middle or at the end of a clause. The mid position is between the subject and the main verb. Where there is more than one verb, or if the main verb *be* is used, mid position means after the first auxiliary verb or after the *be* verb. For more information, see [18] and [19].

British English vs. American English

This section gives an overview of the differences between British English and American English. The majority of the publications are written in American English, as this text is as well. Although the Swiss Education System prescribes British English as part of its curriculum, most people tend to switch to American English after school or keep a mixture of both styles.

The most obvious difference between British English and American English is probably the letter *s* that becomes a *z* as for example in *optimize*, *analyze*, *catalyze*, etc. But there are many other differences that also include more than just spelling differences. An overview is given in appendix A.2.

Capitalization Rules

In regular sentences only names, the pronoun *I*, and the first letter of the sentence are capitalized. This is the so-called *sentence case* style. For titles it is possible to use sentence case as well. However, many authors use *title case* instead. There are many different style guides for the capitalization of titles, see e.g. [20]. The one recommended by IDSC is given by the U.S. Government Printing Office Style Manual. It recommends that all words in titles be capitalized, except for *a*, *an*, *the*, *at*, *by*, *for*, *in*, *of*, *on*, *to*, *up*, *and*, *as*, *but*, *or*, and *nor*.

2.4 Writing Style

When reviewing student reports and other literature contributions, a number of stylistic shortcomings can be noticed again and again. Most of the time, these shortcomings are not errors in the sense that something is strictly wrong, but errors in the sense that they do not help to present the student's work in the most precise way possible and thus hinder the reader in understanding the work quickly. These shortcomings make a text cumbersome to read because the reader has to go back and forth through the paragraphs in order to be able to follow the argumentation.

Therefore, this section provides a list of the most prominent "mistakes". While many of the items listed might seem obvious, students may still find one or the other point that they have not thought of before, and therefore keeping the list in mind while preparing a report might still be worthwhile.

2.4.1 Addressing the Reader

Before starting to write a text, the author should first make up his mind about whom he or she wants to address. Of course, many scientific and technical reports address a very specific audience, and undoubtedly many of the readers themselves are experts in the given field. However, any text benefits from trying to address the broadest possible audience. By doing so, the author has no choice but to abstain from assuming that readers are familiar with certain concepts already, or that they are in possession of all the pieces of information necessary to understand the text. A text written in this way not only becomes understandable by non-experts as well, but also readers find the chain of arguments easier to follow. A number of rules has to be considered when aiming to formulate a text that is easy to understand.

- Avoid making things more complex and complicated than they already are. Go for the most simple explanation possible, always. Instead of trying to sound like an expert yourself, try to translate the complicated technical expressions of your topic and the cutting edge research concepts developed into simple, small pieces that the reader can digest more easily. Being able to give the reader a clear impression of your complicated work will impress more than being a show-off. Keep the sentences short.
- Avoid "tech slang" whenever possible. If the use of certain special words is unavoidable, take the reader by the hand and explain what the word means before diving into your own explanations.

2.4.2 Tense and Voice

Avoid passive voice. Expressions like "It is useful to ...", "One can think of ...", or "In Figure XY it can be seen that..." are possible in German while in English they sound weird and strange. Better ways to formulate the latter example are "Figure XY shows ...", or "The results in Figure XY indicate ...".

Technical reports are always written in present tense. The past and future forms should only be used occasionally when the action is related to a specific point in the past or in the future. In general, the text refers to events that are not time-dependent and thus are valid at all times.

2.4.3 Precision

Always be as precise as possible in a scientific text. For example, the sentence "The electric vehicle was able to travel a long range before the battery was depleted" is a poor sentence, because it leaves the reader with many open questions and lacks a precise and objective measure to quantify the performance. A better sentence would be "On the New European Driving Cycle, an all-electric range of 100 km is achieved."

Precision of Numbers

Sometimes, when stating a result in form of a number, students simply export the full numeric floating point number from their measurement equipment or from their simulation. Thereby, they forget that the number of digits should give the reader an idea of the precision of the result. For example, a statement like "A fuel consumption of $4.637495\,{}^{\mathrm{l}}/_{100\,\mathrm{km}}$ is achieved" would indicate that the measurement or simulation is precise up to $10^{-3}\,{}^{\mathrm{ml}}/_{100\,\mathrm{km}}$, which is obviously impossible to achieve using real measurement equipment or any kind of sensible simulation. To avoid this problem simply round numeric results according to the actual precision of the measurement or simulation.

2.4.4 Acronyms

While the introduction of acronyms for widely used but lengthy expressions like *GPS* for *global positioning system* is arguably useful (because it takes up less space and may even improve readability) the use of too many acronyms should be avoided. Most readers are be familiar with general acronyms that are often used throughout many fields of research and engineering. Consequently, the use of these general acronyms is fine. However, there exist even more acronyms for very specific things in very specific fields of research, and only experts of that specific field are familiar with these acronyms. The regular reader, however, has to revisit the nomenclature section every time the acronym is used. Of course, this circumstance distracts and annoys the reader and therefore causes a bad reading experience. The problems worsen if a great number of special acronyms is used, and they get unbearable if the writers start to introduce their own acronym definitions.

Therefore, whenever being tempted to use an acronym, the writer should think of the following questions: Is space a limited quantity for the text I am writing? Who do I want to address with my

text, and is the typical reader be familiar with the acronym already? If any of the above questions can be answered with *no*, then do not use the acronym, but write the full expression instead.

Newly Defined Acronyms

There exists only one reason to introduce a new definition of an acronym—namely to label a newly developed methodology, algorithm, or idea with a concise name that eventually is to be remembered by other researchers in the community. A good example for such an acronym is "ECMS", which stands for "equivalent consumption minimization strategy", an energy management strategy for hybrid electric vehicles. Obviously, such acronym definitions are useful in journal and conference publications, while in student reports they are superfluous. More importantly, these special acronyms should be granted only to ideas that potentially will have a big impact on the scientific community. Of course, this statement is not meant to devalue scientific findings of a lesser impact. There is just no need to try to make something small look big by randomly assigning fancy names in the hope to be remembered by other researchers.

Note that the IDSCreport class provides a special command `\typeabbr` used to add acronyms and abbreviations to the nomenclature of the document. For more information, the reader is referred to section 4.10.

2.4.5 Abbreviations

Similarly to acronyms, the use of abbreviations can interrupt the reading flow and therefore distract the reader. Thus, abbreviations should be avoided whenever possible. Exceptions are standard expressions like *e.g.*, *i.e.*, *etc.*, and so on. Also, depending on the template used, references to figures, tables and other numbered items in a text may be abbreviated. The special command `\typeabbr` should be used to add the abbreviation to the nomenclature the first time it is defined in the text, see section 4.10.

2.4.6 Statements of Judgment

Of course, the purpose of a report is to explain to the reader the relevance of the research conducted within the project and to express the importance of the findings. However, when writing scientific reports, one important thing to keep in mind is that readers most likely are researcher themselves, and that they likely are able to make up their own minds about the presented work. Therefore, avoid judging your own work by telling the reader how "good", "simple", or "useful" the new methodology is. Rather than that, try to convince the reader that the presented way of solving a problem is the right way. To achieve this goal, using plain arguments rather than statements of judgment is way more effective. In fact, the only places in a scientific text where judging statements are not out of place are the introduction, when reviewing the current state of the art, and the conclusion.

2.4.7 Parentheses and Footnotes

Other than in German language, English written texts use parentheses very sparsely. Especially, the German habit of adding a large number of relative clauses in parentheses to the text should be avoided in scientific texts. Also avoid adding too many footnotes. Whenever being tempted to use parentheses or footnotes to add additional information to your text ask yourself whether the added piece of information is relevant and essential for the reader to understand the text. If the information is relevant or essential, simply add it to the main body of the text. Probably, a new sentence has to be added. If the information to be added is neither relevant nor essential, then omit it altogether.

Chapter 3

How to Use the Template

The purpose of this chapter is to familiarize the reader with the IDSCreport *template. Up to this point, this document focused on the general aspects of scientific writing. Starting with this chapter, the focus of this document shifts towards the actual use of the LaTeX typesetting system and the template. The chapter begins with general information about LaTeX and the file structure that is typically pursued. It then describes the document-related features of the LaTeX document class* IDSCreport *and finishes with instructions and background information on the compilation process. Please note that the author assumes that the reader is familiar with the basic functionalities of LaTeX already, e.g., functional commands, command definitions, etc. If readers have never worked with LaTeX before, they are referred to Appendix B and the references therein.*

3.1 Background Information on LaTeX

The acronym LaTeX represents a family of typesetting programs that follow the philosophy of separating the content from the presentation. Thereby, the author does not directly see the final document while writing, but uses special commands to *program* the appearance of the content.

The abbreviation LaTeX stands for Lamport TeX, where T, E, and X (pronounced "ch" as in "loch") come from the capital Greek letters *tau*, *epsilon*, and *chi*, and Lamport is the last name of one of the program developers.

The history of LaTeX goes back to 1978, when Donald Knuth implemented the first typesetting program of the LaTeX family which he called TeX. Since programming with basic TeX commands is very complicated, Leslie Lamport developed a wrapper around TeX in the early 1980s, which he called LaTeX. This higher level language can then be converted into the original basic TeX commands, using an appropriate compiler.

In principle, TeX and LaTeX are not software applications but special-purpose programming languages of a typesetting system. This multilingual system contains several programs that are executed via command-line interfaces. Due to the limited computational power and the limited memory back in the days when the TeX programming language was developed, the process used to convert the source files into the final graphical document consists of several individual steps, where each step is conducted by a separate program. The central program that is typically used within this typesetting routine is called `latex`. This program translates the plain-text source code into a device-independent file format (DVI). This file format is not intended to be human-readable, but represents the visual layout of the document as binary data which is independent of specific image formats and printers. DVI files are then further converted via so-called DVI drivers to printable formats such as PostScript.

When the PDF format came along in the 1990s, the need for a direct build process from source code to PDF without the DVI and PostScript stages emerged. During his PhD thesis Hàn Thế Thành developed the program `pdftex`, which (similar to the relation of the programs `tex` and `latex`) is the underlying code of the program `pdflatex`. These newer programs run faster and offer the ability to include image formats such as PNG or JPEG and PDF documents. As a downside, it is not possible to include Encapsulated PostScript (EPS) documents, which is straightforward in the indirect way with DVI-to-PostScript drivers and a final PostScript-to-PDF conversion.

3.2 Installation of LaTeX

LaTeX is not an application, but a part of the TeX typesetting system that is executed via command-line interfaces. In order to simplify the installation of LaTeX and its major dependencies, several distributions of the TeX typesetting system are available:

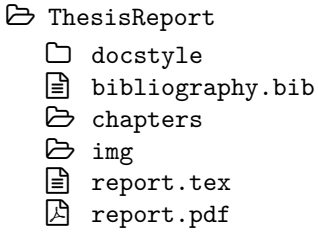
- TeX Live [21] is a free software distribution for the TeX typesetting system that includes major TeX-related programs, macro packages, and fonts.
- MacTeX [22] is a distribution for MacOS that contains TeX Live and a set of front end applications such as the TeX editor TeXShop, the package manager TeX Live Utility [23], the reference manager BibDesk, etc.
- MiKTeX [24] is a distribution for Microsoft Windows that contains TeX Live and a set of front end applications such as the TeX editor TeXworks and the MiKTeX Package Manager [25].

Whether you have installed the plain TeX Live distribution or a package that includes additional software is not relevant. More important is that the software is up to date. Experience has shown that many compilation problems with advanced document structures can be solved by updating the typesetting system. As suggestion, we recommend for Windows users to install MiKTeX and for MacOS users to install MacTeX, and manually uninstall the unused applications such as TeXworks and TeXShop afterwards.

3.3 Structure of the Template

This template basically consists of one class definition, the LaTeX document class `IDSCreport.cls`. As this main file implicitly includes several other files, the document definition is contained in a separate folder called `docstyle`. The template is used by copying the folder `docstyle` to the same directory as the root file of the document which contains the document class declaration `\documentclass{docstyle/IDSCreport}`. This root file is written in plain text and is usually given the extension TEX to indicate that it is a LaTeX source file. In the example below, this root file is named `report.tex`.

For reasons of clarity and comprehensibility, chapters are usually written in separate LaTeX source files that are included by the root file. These files are stored in the folder `chapters`. Likewise, images and source files graphics are stored in another supplementary folder called `img`. The folder structure on the top level of the template may look like the following example:



3.4 Class Options

The document class IDSCreport offers two report-related options and several other options for changing the layout of the document. The options are given similar to other document classes in brackets directly after the LaTeX command `\documentclass` and before the class name, which is embraced by curly braces. The different options are separated by commas.

A possible definition of the document class may look like the following example:

```
\documentclass[english,mt,compact,draft]{docstyle/IDSCreport}
```

The document class name contains the relative path to the class file `IDSCreport.cls`, since it is located in the folder `docstyle`.

3.4.1 Report-Related Options

The three report-related options are the definition of the language, the definition of the report type, and the definition of whether the report is part of a teamwork. Currently, the implementation supports German and English, where

- `german` sets the default language to German,
- `english` sets the default language to English.

If neither the option `german` nor the option `english` is stated, the document class assumes that the default language is English.

The report type is either a Studies on Mechatronics, a Bachelor's thesis, a semester project, a Master's thesis, an internship report, or a generic technical report, where

- `sm` sets the document type to *Studies on Mechatronics*,
- `bt` sets the document type to *Bachelor's thesis*,
- `sp` sets the document type to *semester project*,
- `mt` sets the document type to *Master's thesis*, and
- `is` sets the document type to *internship report*.

If none of the above options are given, the document type is automatically set to *technical report*. Note that in this case the layout of the title page is slightly different and the document does not end with an information page.

In case the student project is conducted in a team, the option

- `teamwork` allows to specify two authors, see also section 3.5.

Without this optional argument, the document class accepts only one author for reports that are related to student projects.

3.4.2 Layout-Related Options

By default, a student report created with the document class IDSCreport automatically appends an information page called *infopage* that contains personal information of the student, the document type and identification number, etc. and the declaration of originality.[1] With the following two options this infopage can be adjusted. In particular,

- `noinfopage` prevents the document class to print the infopage, and
- `nodeclaration` prevents the document class to print the declaration part of the infopage.

Typesetting large documents sometimes consumes a lot of time, although only textual changes are made. On that account, the standard classes *article*, *report*, etc. offer the option to typeset the document in *draft mode*. This option is inherited and is also offered by the document class IDSCreport, i.e., the option

[1] If the report is generic, i.e., it's a technical report, no infopage will be produced.

- `draft` causes to speed up the compiling process by using draft mode.

A nicely designed report contains empty pages such that all new chapters, the abstract, the table of contents, etc. are printed on an odd page. For proofreading, a printed version is still better than a digital version, but empty pages are unnecessary. Therefore, the option

- `compact` creates a document that does not contain any empty pages.

Note that a document written with the document class IDSCreport automatically creates empty pages where they are required. For more information consider appendix D.2. As a result, you cannot remove these pages manually. You have to use the *compact* option.

3.5 Main Structure of the Report

In contrast to a regular document, the document class IDSCreport modifies the definition of the so-called preamble, i.e. the code above the command `\begin{document}`. As the target of the class is very specific, it is useful to load many packages in the class file itself. These packages include language and font definitions, convenient commands for graphics, equations, tables, cross-references, links to web pages (URL), etc. A detailed list is provided in appendix D.1.

The preamble of a report is used to define the title, author, supervision, date etc. of the document. The following commands must be used if the report is not generic[2],

- `\title` defines the title of the document,
- `\author` defines the name of the author,
- `\ethid` defines the student's identification number of ETH Zurich,
- `\semester` defines the number of the semester in which the student is registered,
- `\email` defines the e-mail address of the student,
- `\supervision` defines the supervision of the thesis[3],
- `\identification` defines the identification number of the thesis,
- `\date` defines the date of the final submission,
- `\keywords` defines the keywords related to the report[4],
- `\bibliography` is used to define the bibliography file names required for the citations.

If the document has a subtitle, the command `\subtitle` can be used to define the subtitle of the document. Avoid writing the subtitle into the command `\title`.

If the thesis is a teamwork, instead of using `\author`, `\ethid`, `\semester`, and `\email`, the preamble must contain the commands `\authorA` and `\authorB`, `\ethidA` and `\ethidB`, `\semesterA` and `\semesterB`, and `\emailA` and `\emailB`.

The commands `\title` and `\author` offer an optional argument[5], which is explained in detail in appendix D.2.[6]

3.5.1 Front Matter

The front matter contains at least the title page, the abstract, and the table of contents. In addition, it can contain a copyright page, an acknowledgment, a preface, and a nomenclature. These front matter chapters are all created automatically if they are defined. The text has to be written within the preamble of the root file (above `\begin{document}`) using the following environments:

- `copyrightpage` for the copyright page,
- `abstract` for the abstract,

[2] For generic reports only the commands `\title`, `\author`, `\date`, `\keywords`, and `\bibliography` are required.
[3] For multiple supervisors, separate the names by the LaTeX command `\\`.
[4] Different keywords are separated by commas.
[5] All LaTeX commands are of the form `\command[optional arguments]{mandatory arguments}`.
[6] Although not specifically mentioned and explained, the commands `\subtitle`, `\authorA`, and `\authorB` offer this optional argument as well.

- `acknowledgment` for an acknowledgment,
- `preface` to preface the report.

The nomenclature is created automatically if it contains at least one entry. Entries are added as explained in section 4.10.

3.5.2 Main Body of the Report

The body of the document contains all chapters related to the content of the report. In LaTeX code the body starts with `\begin{document}` and ends with `\end{document}`. Within this environment it is possible to directly write text or include other plain text files with the command `\input`, which has exactly the same result as if the included text was written where `\input` is placed. But as separating the source files makes a lot of things much easier, it is usually done for every regular chapter and every Appendix. Appendices are separated from the regular chapters with the command `\appendix`. An example is given in section 3.7.

3.5.3 Back Matter

The back matter contains the bibliography and the information page with the declaration of originality. Both elements are generated automatically by default. For more information consider appendix B and section 4.12 for the bibliography and section 3.4.2 for the information page, respectively.

3.6 Additional Packages

Although the document class already includes a lot of packages, your specific report might require additional packages. The best location to load them is directly after the document class definition. Also, additional commands should be specified at this location.

3.7 Example: Main File of This Document

In order to give an example of how to set up the root file of a report, the following code shows the implementation of the main file of this document, `IDSCreport_HOWTO.tex`. The meaning of the first lines that begin with a percentage symbol is explained in section 3.8.3. Amongst other things, these "magic comments" define the encoding and the build process of the document. Note that this document has no specific type and is thus a generic technical report. Accordingly, not all commands given in section 3.5 are used. This is different for reports of student projects.

```
% !TeX encoding = UTF-8
% !TeX root = IDSCreport_HOWTO.tex
% !TeX TXS-program:compile = txs:///latexmk/[-pdf -f -silent -shell-escape - ↵
latexoption="-synctex=1" -output-directory="build" -r "docstyle/ ↵
nomenclature_latexmkrc"]
% !TeX TXS-program:quick = txs:///compile | txs:///view

\documentclass{docstyle/IDSCreport}

% Use packages and define commands related to this document
\usepackage[update,prepend]{epstopdf} % used for including EPS files
\usepackage{fontawesome} % used for nice symbols in Section 3.1 Structure of ↵
Template
\usepackage{circuitikz} % used for the electric circuit via TikZ
\usepackage{cprotect} % allow verb|...| in macros such as \caption{text}
\usepackage[type={CC},modifier={by-nd},version={4.0}]{doclicense}
```

```
\newcommand{\texcmd}[1]{\texttt{\bs#1}} % used to simplify typesetting of LaTeX ↵
 commands

% Externalize eps graphics containing tags. Argument must be identical to - ↵
output-directory of main tex program. (see above)
\externalizepsfrag{build}

% Externalize the creation of pgfplots such that they are only compiled when  ↵
needed.
\usepgfplotslibrary{external}
\tikzexternalize

\tikzset{
  external/system call={pdflatex \tikzexternalcheckshellescape --halt-on-error  ↵
  --interaction=batchmode --output-directory=./build --jobname "\image" "\ ↵
  texsource"}, /pgf/images/include external/.code={\includegraphics{build/#1}}
}

% Define the properties of this report
\title[How to Use the IDSCreport LaTeX Class]{How to Use the\\IDSCreport \LaTeX ↵
 Class}
\subtitle{Version 1.6.0}
\author[Andreas Ritter]{Andreas Ritter, Dr. Philipp Elbert, Prof. Dr.  ↵
Christopher Onder}
\date{December 2018}
\keywords{LaTeX, Template, Student projects, Institute for Dynamic Systems and  ↵
Control, ETH Zurich}
\bibliography{bibliography,IDSCreport_HOWTO,docstyle/bibtexstyle/IEEEabrv}

\begin{copyrightpage}
\doclicenseThis
\end{copyrightpage}

\begin{abstract}
One important aspect of scientific research is the documentation of its results ↵
. In order to conserve newly found knowledge and ease the access to results,  ↵
every researcher needs to acquire the ability to prepare reports in compliance  ↵
with scientific standards. Only in this way, the new findings can be conserved  ↵
and be communicated to supervisors, successors, and the scientific community.  ↵
However, the task of preparing a high quality scientific report can represent a ↵
 difficulty--especially for undergraduate students conducting their first  ↵
research projects without having any experience to rely on, but also for  ↵
graduate students who are beginning their scientific careers as PhD students.  ↵
In order to facilitate the process of preparing high quality scientific reports ↵
, the Institute for Dynamic Systems and Control (IDSC) provides two items  ↵
aiming to help young students and researchers in the process of learning the  ↵
basics of scientific writing:

The first item is the \emph{IDSCreport \LaTeX Class}, which can serve as a  ↵
template in the preparation of a report. The use of the IDSCreport template is  ↵
mandatory for all student projects conducted under the supervision of Prof. Dr. ↵
 Christopher Onder, such as Studies on Mechatronics, Bachelor's theses,  ↵
semester projects, Master's theses and internships. This standardization aims  ↵
at ensuring that all reports produced at IDSC share identical formatting, as  ↵
```

```
well as that the documented results from different projects are compatible. The ↵
 students benefit from the availability of a starting point when beginning to  ↵
learn the \LaTeX typesetting system.

The second item is this governing document, which is divided in three parts.  ↵
After a general motivation, the first part provides a number of tips to  ↵
consider when writing the text of a report. The second part introduces the  ↵
functionalities of the IDSCreport \LaTeX Class, while the third part covers the ↵
 IDSC guidelines to consider when programming the typesetting of all items that ↵
 are not regular text, such as figures, tables, citations, etc. These  ↵
guidelines define the mandatory standards for all student reports of projects  ↵
conducted under the supervision of Prof. Onder. The students benefit from a  ↵
large knowledge base where they can quickly access a lot of information in case ↵
 of doubt: What are the scientific standards? What is considered the most  ↵
appropriate implementation? And where can further literature be found?

By providing these two items to our students, we hope to guide young  ↵
researchers in taking their first steps of their career, maintain the quality  ↵
of scientific reports written at IDSC at a high level with minimum effort, and  ↵
keep students from struggling with the difficult task of preparing good reports ↵
.
\end{abstract}

\begin{preface}
This template can be seen as the third major revision of the original style  ↵
file \texttt{ethimrt.sty} written by Eric A.~Müller in 2003. In 2009, Soren  ↵
Ebbesen adopted the style file and renamed it \texttt{ethidsc.sty} in  ↵
accordance with the name change of the institute. Soren maintained the style  ↵
file until he left the research group in 2014. When Prof. Dr. Lino Guzzella  ↵
became President of ETH~Zurich on January 1, 2015, Dr. Christopher Onder,  ↵
formerly a senior scientist, was appointed professor and continued the research ↵
 in automotive control and optimization with his own research group.

Having the honor of being one of his first doctoral students, I started  ↵
rethinking the challenging tasks of supervising student projects and how the  ↵
whole process could be standardized and therefore be simplified and streamlined ↵
. With this goal in mind, I started rewriting the \LaTeX template for student  ↵
reports and converted it from a style file to a class definition. In addition,  ↵
I created this how-to guide in order to not only give students more information ↵
 on the \LaTeX-related instructions used by the template, but to provide  ↵
guidelines for writing reports in accordance with scientific standards as well.

Much of my view on supervising student projects was inspired by my former  ↵
supervisor, Dr. Philipp Elbert. On that account, I hereby express my sincere  ↵
gratitude to him for teaching me how to improve my writing style and my  ↵
presentation skills. Philipp was also involved in the development and revision  ↵
of this text. Finally, I want to thank Severin Hänggi and other colleges of  ↵
Prof. Onder's research group for helping me with the implementation of the \ ↵
LaTeX document class. \\[1cm]

Zurich, December 2016 \hfill \textit{Andreas Ritter}
\end{preface}

\begin{document}
```

```
\input{chapters/introduction}
\input{chapters/writingtips}
\input{chapters/howto}
\input{chapters/guidelines}
\appendix
\input{chapters/english}
\input{chapters/literature}
\input{chapters/graphics}
\input{chapters/implementation}
\end{document}
```

3.8 Compiling the Document

To create a document with LaTeX, several typesetting programs have to be executed in a specific order where the main programs `latex` or `pdflatex` have to be executed even multiple times. Furthermore, the author has to decide whether to follow the traditional build chain via DVI and PostScript or the newer build chain that directly produces a PDF document out of the source files. This section depicts the differences between the two recommended build chains and presents the standard execution order of the typesetting programs.

3.8.1 Build Chains

The so-called *DVI-PS-PDF chain* is the traditional way of creating LaTeX documents. The process allows EPS graphics to be included directly and usually achieves a much smaller file size of the PDF generated than the one produced by the PDF chain. As a major downside, images in PNG format or JPEG format cannot be included. Since the conversion of DVI to PostScript and PostScript to PDF are both carried out via programs on the command-line interface as well, this build chain in principle has also a defined execution order of programs, i.e. `latex`, `dvips`, and `ps2pdf`.

The *PDF chain* is the modern approach for typesetting LaTeX documents. It is much faster than the traditional DVI-PS-PDF chain, but it cannot include EPS graphics directly. Instead, they have to be converted to PDF files first either manually or via preceding conversion processes. For example, the latter approach is pursued by the package `epstopdf`. Thereby, the PDF chain offers greater flexibility in including external files. In order to enable the preceding conversion, a special option has to be given to the program `pdflatex`. For more information, please consider the manual [26] of `epstopdf`, which is also included in the distribution of this template.

Other build chains such as the *DVI-PDF chain* are not recommended since the advantage of manipulating the PostScript graphics via the DVI-PS-PDF conversion is lost. Consider the documentation of the LaTeX package `psfrag` [27] for more information on the power of manually adjusting vector graphics.

3.8.2 Build Recipe (Program Execution Order)

Because of the limited memory that the computers had when LaTeX was developed, the building process required to produce a complete document was split up into separate stages. Each stage adds information to the document based on information available from the previous stage. As an example, the cross-references can only be added after the typesetting process knows to which part in the document they should make reference.

The shortest build recipe to get all cross-references right with the main programs `latex` and `pdflatex` is the following:

1. `latex` or `pdflatex`
2. `bibtex`

3. `makeindex`
4. `latex` or `pdflatex`
5. `latex` or `pdflatex`

In the first run, `latex` or `pdflatex` reads the TEX files and creates AUX files with all citation keys and the label information. The program `bibtex` then extracts the citations from these AUX files, identifies them in the BIB file, and writes the formatted references into a BBL file. The second run of `latex` creates the cross-references based on the AUX file, the TEX file, and the BBL file. However, even after this execution, the numbering of the labels is not yet clear. Hence, a fourth step is required to place the correct citation labels in the document [28, 29].

Having said that, when using the program `latex` the result is still only a DVI file, which is finally converted to a PDF document via the programs `dvips` and `ps2pdf`. The program `pdflatex` on the other hand directly produces a PDF document.

Although the above recipe is required to ensure that the document is complete, the cross-references are usually not important during the writing process. Therefore, typically only one typesetting process is executed, i.e. the DVI-PS-PDF chain with `latex`, `dvips`, and `ps2pdf`, or the PDF chain with `pdflatex`.

3.8.3 Recommended TEX Editor

In order to simplify the execution of LATEX-related programs, many different graphical user interfaces have been developed. Practically all of them come with writing editors and are therefore known as TEX editors. This section describes why we have chosen to use the editor TeXstudio, and what adaption are necessary to compile the document in the recommended way.

TeXstudio

TeXstudio [30] is an open-source software application for writing LATEX documents. Compared to the variety of other writing environments, TeXstudio is available for all major operating systems including Windows, MacOS, and Linux. Due to this consistency and the various features of TeXstudio that are really helpful for writing LATEX documents, we highly recommend to use this software.

It is highly recommended to go through the section *Features* on the web page of TeXstudio in order to be able to use the software with its full power, and to consider its manual [31] for further information.

In addition to the features, TeXstudio allows almost every functionality to be configured and customized. These settings can be exported and imported easily via the *Options* menu of TeXstudio. Profiles customized by IDSC are available on the NAS for both Windows and MacOS. Please contact the supervisor of your project.

Magic Comments

TeXstudio allows to use so-called “magic comments”. They allow options for the editor and compilation process to be set on a per-document level. The use of the following commands is recommended for each separate source file of the document:

- `% !TeX spellcheck = en_US`
  defines the language used for spell-checking the document. To see which languages are supported, open the *Configure TeXstudio* window and select the section *Language Checking* on the left. The drop-down menu next to *Default Language* contains all available languages for spell-checking.

- `% !TeX encoding = UTF-8`
  defines the character encoding of a document. In order to ensure the highest level of compatibility, everyone at IDSC should use UTF-8 for all documents.
- `% !TeX root = filename`
  defines the root document for this file. For files that are located in subdirectories with respect to the main document, give the filename as relative path, e.g. `../main.tex`.

The root document contains the definition of the entire document and includes all other files. Therefore, it also defines which LaTeX program is used. Consequently, we recommend to use the following command that relies on the powerful program `latexmk`, which automates the entire compilation process.

- `% !TeX TXS-program:compile = txs:///latexmk/[...]`
  enforces TeXstudio to create the document via the perl script `latexmk`. More information on this procedure and the recommended options (here represented with the three dots) is found in the following section.
- `% !TeX TXS-program:quick = txs:///compile | txs:///view`
  redefines the standard compilation program `quick` of TeXstudio that is associated with the button *Build & View*. It first executes the command defined in the item above and then calls the standard PDF viewer, which is either the internal PDF viewer or any external application. Customize the behavior of the PDF viewer in TeXstudio's preferences under the section *Build* at *Default Viewer*.

Automated Document Compilation

By default, the TeXstudio program `quick` is executed via the button *Build & View* or the shortcut F5. Therefore, by using this unique feature of TeXstudio, the magic comments automatically ensure that the source files are encoded correctly and the document is produced via the designated typesetting programs. Furthermore, in combination with `latexmk`, the build process automatically detects which part of the build recipe of section 3.8.2 has to be executed. For compiling the document directly with the PDF chain the following set of options of `latexmk` is recommended:

```
-pdf -silent -shell-escape -latexoption="-synctex=1" -output-directory="build ↵
" -r "docstyle/nomenclature_latexmkrc"
```

The first two arguments ensure that `pdflatex` is used, without typing a complete diagnosis output. The third option enables LaTeX to call internal compilation processes on its own. The fourth option enables synchronization between the source code and the text of the final PDF. The fifth option enforces LaTeX to put all compilation files such as AUX files, BBL files, etc. into a folder called *build*. Note that the PDF file and the log file is also in this subdirectory. The last option refers to a specific file included in the distribution of this template. It customizes the program `makeindex` in order to typeset the nomenclature automatically.

Instead of using the plain programs `latex` and `pdflatex` in the build recipe, we can now replace the entire build chain with TeXstudio's *Build & View*. The build recipe becomes thus a simple click on the button *Build & View* or pressing F5 on the keyboard.

Add Search Path for PDF and Log File

If LaTeX compiles the document into a specific directory as for example described in the previous section, we must allow TeXstudio to search for the PDF file and the log file in this specific directory. To do so, we add the folder name in TeXstudio's preferences under the section *build*. Show the advanced options by setting a tick to the box on the lower left of the window and type the folder name, e.g. *build*, into the text fields of both, the *Log file* and *PDF file*. Compare figure 3.1 for reference.

Note that this additional search path only works with the internal PDF viewer. In case you want to use an external viewer such as Skim [32] for MacOS or Sumatra PDF [33] for Windows, the search path has to be adapted explicitly in TeXstudio's preferences under the section *commands*.

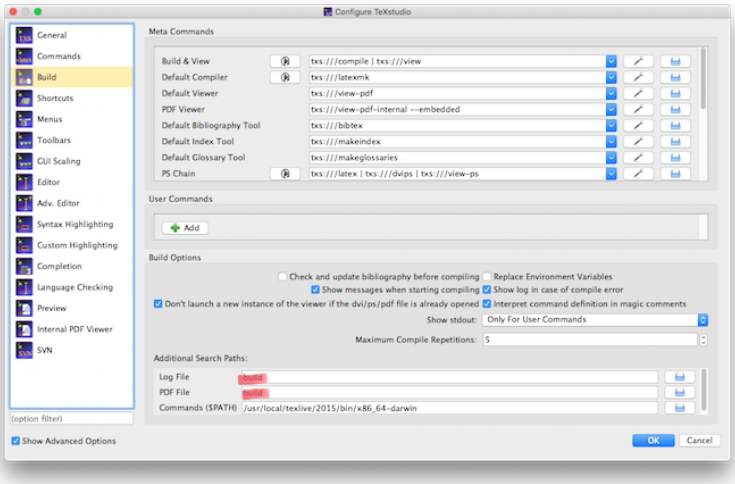


Figure 3.1: TeXstudio's preferences for the additional search path. Write *build* into both text fields for the *Log file* and *PDF file* in order to use the internal PDF viewer and the Log display properly.

Installation of Perl and `latexmk` on Windows

The LaTeX package `latexmk` is dependent on Perl, a family of high-level, general-purpose, interpreted, dynamic programming languages [34]. In contrast to Unix-based operating systems such as Ubuntu and MacOS, Perl is not part of the basic distribution. Therefore, it has to be installed manually. Download ActiveState Perl from `https://www.perl.org`, select the typical installation without changing the default options, and reboot your machine.

Usually, TeX editors such as TeXstudio download required packages automatically if it has been given the permission to do so. However, if a package is not explicitly included via `\usepackage` or `\RequirePackage` in a class definition, the editors may not detect the dependency and fail to compile the document. This issue has been observed on Windows with older versions of TeX Live or MiKTeX, where the package `latexmk` is not installed by default. Accordingly, consider updating TeX Live or MiKTeX as a first attempt.

The easiest way to install additional packages is via designated package managers such as MiKTeX Package Manager [25] on Windows or TeX Live Utility [23] on MacOS. The following steps cover the installation of `latexmk` on Windows using MiKTeX Package Manager.

- Find the MiKTeX Package Manager via the Start button and search. Run the application as administrator.
- Search for `latexmk` and install it as described in [25].

For more information, consider [35, 36]. Especially the file `INSTALL` of the `latexmk.zip` package available from `http://mirrors.ctan.org/support/latexmk.zip` provides detailed instructions for the installation without any package manager.

Some users have also found that the solution provided by [37] is necessary to completely address the issue, i.e.,

- copy the files `latexmk.pl` and `latexmk.bat` from `latexmk.zip` to the directory of the error, which could be `C:\Program Files\MiKTeX 2.9\scripts\latexmk\perl` or the like,

- run the application MiKTeX Update as administrator, and
- reboot the computer.

Chapter 4

IDSC LaTeX Guidelines

This chapter provides guidelines for writing and typesetting high-quality reports. Each section of this chapter addresses a different stylistic element common to technical reports. In doing so, the sections explain both the grammar rules and writing styles that should be considered and how the corresponding elements are produced correctly using LaTeX. If more information on grammar is desired, the authors recommend to consult the The Chicago Manual of Style *[38] and the* Oxford Dictionaries *[39] for American English and British English, respectively. For more information on working with LaTeX, appendix B offers a selection of additional references.*

4.1 Headings

The report can be structured using several different types of headings. It is recommended to use chapters, sections, and subsections, all of which appear in the table of contents. The corresponding commands are `\chapter`, `\section`, and `\subsection`, respectively. Each command offers an optional argument and a required argument. The required argument in curly brackets is the actual heading that is printed where the command is used. The optional argument can be used to modify the corresponding entry in the table of contents. It should only be used if the heading contains special symbols that do not appear as expected in the table of contents.

The present chapter and the heading of this section are produced by

```
\chapter{IDSC \LaTeX Guidelines}
```

and

```
\section{Headings}.
```

Headings should always use *title case* style as described by the capitalization rules of section 2.3.2.

Please note that the document class IDSCreport modifies the command `\chapter` such that new chapters automatically appear on new pages with odd numbers. Doing this manually with `\cleardoublepage` is not required. Consider appendix D.2 for more information.

For further structuring the text, it is suggested to use paragraphs. As opposed to the regular headings, paragraphs start with the given text argument in bold font and continue directly with the remaining text, e.g.

```
\paragraph{Paragraphs} are very useful to distinguish different parts of the ↵
text within sections and subsections. They can be used as some kind of ↵
extended listing.
```

produces the following paragraph.

Paragraphs are very useful to distinguish different parts of the text within sections and subsections. They can be used as some kind of extended listing.

The title in bold font can also be in other fonts. Using for example `\texttt{<title>}` within the argument results in a bold true-type font. The first paragraph of section 4.12.1 shows the effect of using this command.

4.2 Quotation Marks and Italics

Quotation marks and italics are stylistic methods of typography to indicate names and terms or to emphasize certain words or expressions within ordinary text. They are used to highlight names of books, articles, chapters, reports, websites, software applications, etc., but also expressions such as key terms or "scare quotes".

As a guide for which technique should be used on which occasion, *The Chicago Manual of Style* [38] is taken as a reference. Other modifications of font such as boldface, small caps, light face, thin, etc. should be avoided.

4.2.1 Names and Terms

According to Chapter 8 of *The Chicago Manual of Style*, names and terms are always written in title case style. Based on the subject, the capitalized expression can in addition be written in quotation marks or in italics.

In general, italics are preferred for major freestanding works such as books, journals, encyclopedias, and reports, for periodicals such as magazines and newspapers, but also for names of ships and names of species.

Quotation marks are used for shorter works including titles of subsections of larger works such as chapter titles, article titles and web pages[1], and for titles of unpublished works such as theses, dissertations, manuscripts in collections, unpublished transcripts of speeches, etc.

On the other hand, titles of collections such as book series or websites[2] are neither placed in quotations marks nor are they italicized. The same holds for brand names that are trademarks such as companies, software applications, product names, buildings, awards, prizes, etc. Note that if for example an award name consists of a newspaper title, the entire name including the newspaper title is written in Roman letters without quotation marks.

A list of examples that follow the above characterization is given in table A.5 of appendix A.3. For more information on product names consider section 4.3.

4.2.2 Words and Expressions

Chapter 7 of *The Chicago Manual of Style* addresses the motivation for emphasizing words and expressions that are not names or terms. In principle, the author is allowed to emphasize any word with italics. However, it only makes sense if the sentence is not constructed in such a way that the emphasis is clear anyway. Moreover, too many emphasized expressions distract the reader, and thus the intended effect is lost.

Italics are typically used for key terms in a particular context. If they are used often within the text, only their first occurrence is italicized. If a specific term is rarely used, it may be written in italic shape on every occurrence. The same holds for phrases from other languages, such as *a priori* and *vis-à-vis*, and for mathematical short notations, such as the word *iff*, which is an abbreviation of the logical condition "if and only if". If these phrases become familiar through repeated use throughout the text, only the first occurrence should be emphasized. In addition, italics are also

[1] In contrast to a web site, a web page is a single page of the web that is accessible via a specific URL.
[2] A website is a collection of web pages that are grouped together.

used for words or terms that are not used functionally but refer to the words or terms themselves, which is for example the case in appositions.

Quotation marks are used whenever they are more appropriate in a specific context than italics. This can be due to a distinction from words in italics, as for example if words of foreign languages are translated to English, or if they help to express the author's way of thinking and speaking. They are also used for "scare quotes", where the words and expressions that are not used in the regular sense but in a nonstandard, ironic, or otherwise special sense. Similarly, quotation marks can be used for specific words of phases preceded by *so-called*, where the focus might not be entirely clear to the reader. However, if for example only one subject is preceded by *so-called* and thus the reference is clear, neither italics nor quotation marks are needed.

4.2.3 Plural Forms and Punctuation

The plural of an italicized term is usually formed with a Roman letter *s*, unless the term is already in plural form. In case a term is written in quotation marks, the plural is formed in the usual way enclosed by the quotation marks. However, *The Chicago Manual of Style* recommends to avoid plural endings that are followed by closing quotation marks and suggests to rewrite the sentence instead.

Please note that, according to *The Chicago Manual of Style*, periods and commas almost always precede closing quotation marks in American English. In British English it is less clear whether the punctuation marks are enclosed by the quotation marks or not. An explanation of the rules can be found on *Wikipedia* [40].

4.2.4 Typesetting commands

In case the author wants to emphasize a certain word or expression with quotation marks, it is recommended to use the LaTeX commands that are provided by the package `csquotes`. The most basic and therefore probably the most frequently used command is `\enquote`, as in "this" expression written with `\enquote{this}`. It automatically enters the correct quotation marks based on the language and the given settings in the entire document. Furthermore, it is much easier to change an emphasis style afterwards among all documents because it is an easy thing to find and replace all commands at once.

To emphasize text with italics the command `\emph` should be used, as in *this* expression written with `\emph{this}`.[3] Although the command `\textit` also sets the text to italics, the two commands are very different. In contrast to the physical font command `\textit`, `\emph` is a style definition that allows words or expressions to be emphasized in surrounding text that is already emphasized. This nested behavior is not possible with `\textit` since it enforces the italic shape, which in turn is appropriate for specific names or parts of names where the italic shape is required.

4.3 Font Styles for Product Names, Commands, Files, etc.

The use of different font styles in the same text assists the reader in quickly capturing the various pieces of information without effort. However, any extensive and inconsistent use of various styles confuses the reader and thus reverses the effect intended. Therefore, authors should always stick to a given guideline, one of which is described in the following paragraphs.

Trademarks of companies, software applications, product names, etc. are written in Roman font with regular capitalized letters such that they fit the usual representation of the name (see also section 4.2.1). Write for example Google, MathWorks, Microsoft, Apple, Matlab, Simulink,

[3] The default behavior of `\emph` is to italicize the enclosed text. However, this definition can change when certain packages are included, as for example the package `ulem`. Therefore, if this particular package is desired by the author, it is highly recommended to load it with the option `normalem` in order to keep the original behavior of `\emph`.

Inkscape, Mathematica, dSPACE, Toyota Prius, Windows, Macintosh, etc. and avoid using trademark symbols such as © or ™. Especially for LaTeX-related names the use of a specific writing style is common, such as LaTeX and TeX themselves, that are written with `\LaTeX` and `\TeX`, respectively, or BibTeX and Ti*k*Z, which are written with `Bib\TeX` and `Ti\textit{k}Z`, respectively. Please note that the first two commands, `\LaTeX` and `\TeX`, are redefined by the document class. In contrast to the default implementation of LaTeX the commands can be used in text without the need for opening and closing braces. See appendix D.1 for details.

Also note that the translations of ETH Zurich (standing for Eidgenoessische Technische Hochschule Zurich) into English (SFIT - Swiss Federal Institute of Technology), French (EPFZ - Ecole Polytechnique Federale Zurich) and Italian are no longer used. In order to establish a precinct trademark that is easy to memorize, it was decided to use the short version *ETH Zurich* only—without writing the full German expression. Documents written in German, of course, use the German umlaut in the name of Zurich.

Commands of any kind of programming language including LaTeX are written in typewriter font with the command `\texttt` or `\verb`. Write for example `\usepackage{packagename}` for including LaTeX packages, `length` for a Matlab command, `pdflatex` for the typesetting program, etc.

File names and file paths, as well as directories and paths to directories are also written in typewriter font. Write for example `picture.pdf` for a file name that should be stored in the subdirectory `./img/`.

File extensions are written in regular Roman font with capital letters, such as EPS, PDF, JPEG, etc.

4.4 References and Footnotes

Adding any supplementary information to text is very easy in LaTeX. Several commands make it very easy to include references and footnotes wherever the writer wants them to be. The following sections describe the specific methods.

4.4.1 Citations

References to literature are included using the command `\cite{<citekey>}`, where `citekey` is the identifier of the document that is referenced. Multiple references can be given by using a list of cite keys, i.e. `\cite{<citekey1,citekey2,...>}`. The description of the reference must be entered together with the unique cite key in the file `bibliography.bib`.[4] Changes in the bibliography file can be made manually with a regular text editor or by using specialized software or reference managers such as *Mendeley*, *JabRef*, *BibDesk*, *Papers*, *Quiqqa*, etc.[5] Section 4.12 describes how to properly define a bibliography manually.

Note that a number of stylistic choices need to be made when citations are added to a text. Most importantly, citations should be included in the text in a consistent manner. Some citation styles name the first author of the cited publications in parentheses, others add the publication year of the cited document. The most common citation style used in the field of engineering is probably the IEEE citation style [41], where a number in square brackets references a specific item in the bibliography, as used in this document and implemented in the IDSCreport class. For reports

[4] You can use a different name for the bibliography file. All you have to adjust is the argument of the command `\bibliography` in the main `.tex` file. For multiple bibliography files, separate the names of the bibliography files by commas.

[5] When using software to automatically create bibliography files, it is recommended that you have two different bibliography files, where one is for manually adding references and the other one is automatically created/overwritten by the software.

written at IDSC, the use of the IEEE citation style is mandatory. When referencing bibliography items in a text this way, the bracketed number should not be treated as a word or as a part of the sentence. The sentence should be a full and grammatically correct sentence without the citation. This rule implies that sentences cannot start with a citation before the first word. If special credit is to be given to the author(s) of a cited publication, the governing sentence can mention their name(s) in addition to the citation. For more information on the style guide for the bibliography, the reader is referred to section 4.12

4.4.2 Cross-References

Cross-referencing within the text is easily done using `\label{<marker>}` and `\cref{<marker>}`. Note that the latter is a special command provided by the package `cleveref` in order to automatize the creation of references. It detects the type of reference and writes the corresponding type in a predefined format followed by the number of the cross-reference. You can reference almost everything that has an assigned number, e.g. chapters, sections, figures, tables, equations, etc. The only thing that has to be considered is that the label must be placed after the objects to which they reference. This means that for chapters and sections, the command `\label` is used after the commands `\chapter`, `\section`, etc. For floats, such as figures and tables, the command `\label` is used after the command `\caption`, where both commands are within the environment of the float. For equations, the command `\label` should be used just before the environment ends. For equations with multiple numbers, see section 4.8.

The reference to this section 4.4.2 is created by placing `\label{sec:CrossReferences}` right after the title `\subsection{Cross-References}` and writing

```
...The reference to this \cref{sec:CrossReferences} is created...
```

Note that the word *section* is not explicitly written. At the beginning of a sentence, use the command `\Cref` with a capital *C*.

In order to indicate a range of references, `\crefrange` or `\Crefrange` can be used. The result of the command `\crefrange{fig:firstFig}{fig:lastFig}` results in something like “figures 2.3 to 2.7”. A list of labels is referenced with `\cref{marker1,marker2,marker3,...}`. Additional possibilities are provided by the package documentation [42], which is also included in the distribution of this template.

Since the same marker can be used to reference any type of label, it is common practice to include an indicator in the marker as shown in the examples above. For reasons of consistency, you should use the following abbreviations separated from the marker by a colon, i.e.

- `chp` for chapters,
- `sec` for sections and subsections,
- `fig` for figures,
- `tab` for tables,
- `alg` for algorithms, and
- `eq` for equations.

4.4.3 Footnotes

Footnotes can be used to provide additional information to a text that is not strictly necessary to follow the argumentation and to understand the text. The author of a text should keep in mind that most readers develop a strong urge to quickly jump to the bottom of the page to read the footnote, regardless of how (un)important the additional information is. Too many footnotes in a text therefore can hinder the reading flow because the reader has to jump back and forth between the text and the footnotes. Since, per definition, the footnote contains only unnecessary information, the reader will not benefit from reading all footnotes. Therefore, the author of a text should always consider the following questions before adding a footnote to the text: Is the

additional information important enough for the footnote to be converted into regular text? Is the additional information unimportant enough for the footnote to be omitted altogether? Only if both questions can be answered with no, the footnote should be added.

Footnotes are added using the command `\footnote{<footnote text>}`. Use the command directly after short notes about a noun or an expression such that the number is not separated by a blank space. When a footnote adds information to an entire sentence as opposed to a single word or expression, use it after the period. Note that in either situation a footnote is an entire sentence and thus starts with a capital letter and finishes with a period.

4.5 Lists

Two types of list environments are commonly used: `itemize` and `enumerate`. The following example uses `itemize` to create a list without numbering.

- point one, and
- point two,

which is created using the following code.

```
\begin{itemize}
  \item point one, and
  \item point two,
\end{itemize}
```

The following example uses `enumerate` to create a list with numbering.

1. point one, and
2. point two,

which is created with the following code.

```
\begin{enumerate}
  \item point one, and
  \item point two,
\end{enumerate}
```

There exists a third type to create a list with custom text as bullet points. The corresponding list environment is `description`.

Thanks to the package `enumitem`, which is loaded by the document class IDSCreport, the list environments presented can be customized very easily with optional arguments. Among a huge selection of useful options, adjusting the vertical spacing between the items is probably the most common desire. Using the options `topsep`, `partopsep`, `parsep`, `itemsep` combined with units of length, or the option `nosep`, result in nice and robust listings. The options must be given as the following example shows:

```
\begin{itemize}[itemsep=1ex,leftmargin=1cm]
```

For further information on the options of this package, consider the documentation [43], which is also included in the distribution of this template.

4.6 Symbols

LaTeX is probably best known for its ability to create documents that include beautiful representations of complex mathematical formulas. Thanks to the open-source character and its large community of contributors, the vast number of packages that are now available allow practically

any imaginable character or symbol to be printed. The only tricky part is to find the correct commands and the associated packages.

An extensive list with over 14'000 symbols is given by the "The Comprehensive LaTeX Symbol List" [44], also included in the distribution of this template. To facilitate the search for a specific symbol, Daniel Kirsch implemented the convenient website Detexify [45] based on the idea of Philipp Kühl.

Having said this, the document class IDSCreport already loads several symbol-related packages that cover the requirements of almost every report. The author of a text is of course allowed to include additional and alternative packages, but the packages presented in the following sections are considered to be the most qualified ones for the specific purposes.

4.6.1 Mathematical Symbols

There is one standard package for mathematical symbols called `amssymb`, which defines the internationally accepted standard for mathematical expressions. It is provided by the American Mathematical Society (AMS) and contains an extended symbol collection that covers every regular mathematical symbol. The package `amssymb` implicitly loads the package `amsfonts` which extends set of fonts for use in mathematics. For a neatly arranged selection of the most important mathematical symbols, the *Wikipedia* entry "List of mathematical symbols by subject" [46] is very helpful.

The document class IDSCreport currently includes one additional package for mathematical symbols, the package `mathabx` [47], which provides even more mathematical symbols such as special arrows, unusual relation symbols, boxed operations, etc. The documentation is included in the distribution of this template.

Note that there exist several rules for the proper use of mathematical symbols. First, all mathematical symbols are typeset in italics. Second, the mathematical operator "$\cdot$", standing for *multiply*, may be omitted in mathematical expressions, as in $A \cdot b = c = Ab$. Because of this rule, mathematical symbols can consist of one letter at maximum. Variable names like $Cost$ or $Vehicle$ are invalid, since they may be misinterpreted, e.g. $Cost = C \cdot o \cdot s \cdot t$. If a variable needs to be specified further, subscripts and superscripts may be used. In some engineering fields, however, variable names that do not comply with this rule may have been established and used ever since. A prominent example is the variable name SoC referring to the *state of charge* of a battery. A good solution to this problem would be to rename the variable x_{SoC}, where x indicates that the variable is a state variable, while the subscript contains the original name in order to ensure agreement with earlier publications.

Subscripts and superscripts in math mode are created by `_{.}` and `^{.}`, respectively. For subscripts and superscripts that are not variables, the command `\mathrm` should be used in order to distinguish descriptive subscripts from enumerating ones. Note that only variables should appear in italics. Therefore, the expression x_k is correct, since k is itself a variable. However, for the expression t_{f}, referring to the final time of a simulation or optimal control problem, the letter f has to be written with `\mathrm{f}`, since the subscript is descriptive. For subscripts and superscripts before a variable, use the command `\prescript` provided by the package `mathtools` for better alignment.

4.6.2 Mathematical Operators

In order to comply with the standard ISO 80000-2:2009 by the International Organization for Standardization (ISO), variables such as x and y, and functions such as $f(x)$ and $g(x)$ are written in italic type, while mathematical operators such as $\sin(x)$ and $\exp(x)$ are written in Roman type. Furthermore, mathematical constants that do not depend on the context should officially be written in Roman type as well. However, many mathematical journals are not as strict and also

allow constants in italic shapes. Accordingly, the guideline for reports related to IDSC is to ignore the ISO standard and use the italic shape for the Euler's number e, π, and the imaginary unit i, produced with `\epsilon`, `\pi`, and `i`, respectively. Numbers are always in upright shape.

Most mathematical operators are represented by specific commands such as `\sin`, `\exp`, `\min`, etc. Some of these commands even offer additional features to indicate specific properties, as for example the command `\sum` in the following example:

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}, \tag{4.1}$$

which is produced with `\sum_{n=1}^\infty \frac{1}{n^2} = \frac{\pi^2}{6}`. Other mathematical operators that are not available via predefined commands can be added via the command `\DeclareMathOperator`, as for example the rank via `\DeclareMathOperator{\rank}{rank}`.

For derivative operators, the document class IDSCreport includes the package `commath` [48], whose documentation is also included in the distribution of this template. This package is particularly useful for ordinary derivatives with the operator "d", which should not be written in italics. The following ordinary derivatives and integrals,

$$\frac{\mathrm{d}f}{\mathrm{d}x} = f'(x), \qquad \int f'(x)\,\mathrm{d}x = f(x),$$
$$\frac{\mathrm{d}^2 f}{\mathrm{d}x^2} = f''(x), \qquad \iint f''(x)\,\mathrm{d}x\,\mathrm{d}x = f(x),$$

are produced with the code below. Note that the command `\dod` is similar to `\tod` and `\od`, but it ensures that the fraction appears in display style, which by default is not the case in an `align` environment.

```
\begin{align*}
\dod{f}{x} &= f'(x) & \int f'(x) \dif x &= f(x), \\
\dod[2]{f}{x} &= f''(x) & \iint f''(x) \dif x \dif x &= f(x),
\end{align*}
```

For partial derivatives, the package `commath` offers the commands `\pd`, `\tpd`, and `\dpd`.

The transpose of the vector variable $\vec{a}$ is $\vec{a}^{\mathsf{T}}$. In order to get a nice transpose symbol, the font style Sans Serif can be used, e.g. `\vec{a}^\textsf{T}`.

4.6.3 Text Symbols

For text symbols a variety of symbol-related packages are available, the most popular of which is `textcomp`. The symbols provided are covered by "The Comprehensive LaTeX Symbol List" [44]. Table 4.1 lists some of the most frequently used symbols. Although all of them have representative commands for the math mode (within equations or within `$` signs), it is recommended to use the commands for text mode when text is written rather than inserting symbols via math mode, e.g. write `\textpm` instead of `$\pm$` in text mode. Note that `\sfrac` is provided by the package `xfrac` [49], whose documentation is also included in the distribution of this template.

Two additional symbols are mentioned here since they are exclusively provided by the document class IDSCreport:

- `\bs` produces "\" which is equivalent to `\textbackslash`.
- `\texttilde` produces "~".

For subscripts and superscripts in text, use the commands `\textsubscript` and `\textsuperscript`, respectively. For degree symbols, please consider section 4.7.

Table 4.1: Frequently used symbols in text mode and their corresponding commands.

Text mode	Math mode	Text mode	Math mode
< \textless	<	⟨ \textlangle	\langle
> \textgreater	>	⟩ \textrangle	\rangle
½ \textonehalf	\sfrac{1}{2}	± \textpm	\pm
¼ \textonequarter	\sfrac{1}{4}	× \texttimes	\times
¾ \textthreequarters	\sfrac{3}{4}	→ \textrightarrow	\rightarrow

4.6.4 Chemical Symbols

Typesetting chemical molecular formulas often results in inhomogeneous results even within single documents. It is therefore recommended to write every formula and every chemical equation with the commands `\ce` and `\cee` provided by the package `mhchem` [50]. This package is already loaded by the document class IDSCreport, and its documentation is included in the distribution of this template.

Chemical molecular formulas are generated with the command `\ce`, as for example H_2O by `\ce{H2O}`. Amounts are directly given as text to the argument of `\ce`, such as in `\ce{1 1/2 O2}` for $1\frac{1}{2}O_2$. Subscripts and superscrips are usually recognized automatically, such as in `\ce{NO3-}` for NO_3^-. Otherwise, they can also be added manually, as in `\ce{CrO4^2-}` for CrO_4^{2-}.

For reactions, the command `\cee` should be used in order to align multiple reactions. The enclosing environment is a simple `equation` or `align` environment as depicted in section 4.8. The following simple reaction:

$$CO + \frac{1}{2}O_2 \rightleftharpoons CO_2 \tag{4.2}$$

is generated with the code given below.

```
\begin{equation}
\cee{CO + 1/2 O2 <=> CO2}
\end{equation}
```

4.7 Physical Units

The document class IDSCreport automatically loads the packages `units` and `siunitx`, which enable physical units to be printed in a nice format. The first package offers the two commands `\unit` and `\unitfrac` that should always be used whenever a value is attached to a physical quantity. In both commands, the value is given with the optional argument and the physical unit is given with the required arguments, as in the following example:

$$\delta t = 1\,\mathrm{s}, \tag{4.3}$$

$$v = 5\,\mathrm{m/s}, \tag{4.4}$$

which is generated by the following code snippet:

```
\begin{align}
\delta t &= \unit[1]{s}\\
v &= \unitfrac[5]{m}{s}.
\end{align}
```

By using these two commands the horizontal space between the value and the physical unit is perfectly adjusted. Furthermore, LaTeX cannot insert a line break between the value and the unit.

When the physical unit of a variable is indicated, use the brackets for the variable, not for the physical unit, i.e. the force F is given by

$$[F] = \mathrm{N} = {}^{\mathrm{kg\,m}}\!/_{\mathrm{s}^2}. \tag{4.5}$$

Note that adding a unit in square brackets to an axis legend of a graph does not comply with this rule. The strictly correct way of adding the units to an axis of a graph would be to state the physical quantity, followed by the word *in*, followed by the unit. Note that this version is valid in English as well as in German documents. Another, less intuitive way would be to *divide* the physical quantity by its own unit, thus yielding a dimensionless quantity that can be plotted in a graph, i.e., "physical quantity / unit". All three styles are widely used throughout the field of engineering, and thus all three styles are allowed for reports written at IDSC.

Please note also that multiple units are separated by a half space with `\,` as in the example above that is produced by the following code:

```
\begin{equation}
[F] = \unit{N} = \unitfrac{kg \, m}{s^2}.
\end{equation}
```

Since printing the degree symbol is not straightforward, the document class IDSCreport uses the degree symbol from the package `siunitx` and provides the following commands that work in both text mode and math mode. Use

- `\celsius` for degrees Celsius (°C),
- `\fahrenheit` for degrees Fahrenheit (°F), and
- `\degree` for angular degrees (°).

In combination with values, the recommended way to print 23 °C is by writing `\unit[23]{\celsius}`. Accordingly, 5 ° is produced by `\unit[5]{\degree}`.

Physical units that are part of the text, as for example in "seven seconds" or in "7 seconds", are written without the command `\unit`. However, in these cases it is strictly required that the physical unit is written out. Furthermore, if any numbers are used, separate them from the physical unit with a text tilde, as in `7~seconds` that produced the above text element. This symbol between the number and the physical unit keeps LaTeX from inserting a line break and leave the number alone at the end of a line. Finally, number ranges should be specified with n-dashes as in 7–8 produced with `7--8`.

4.8 Equations

LaTeX was developed for the proper typesetting of mathematical and physical-related documents. Therefore, it's hardly surprising that equations are some of the most important elements of LaTeX and that there exists a great number of packages that provide all kinds of different methods for typesetting equations. The document class IDSCreport thus includes only a selection of packages which are considered to be standard packages for mathematics. They are

- `amsmath`, which provides all basic features for writing mathematical formulas,
- `amsthm`, which helps to define theorem-like structures, and
- `mathtools`, which fixes various deficiencies of `amsmath` and standard LaTeX.

The first two packages are provided by the American Mathematical Society (AMS) and define the international standard of writing equations. The third package `mathtools` [51] adjusts some of the AMS styles in order to improve their visual representation. The documentation of this package is included in the distribution of this template.

There are various ways to include equations in LaTeX documents, but only two environments are recommended to be used, i.e.,

- the `equation` environment for equations with only one number, and
- the `align` environment for equations with multiple numbers.

4.8.1 Single Equations with One Number

Equations are always a part of the text and must therefore be treated as such. They can appear as complete sentences, clauses, or as parts of clauses. Accordingly, it is important to respect the correct punctuation. As an example,

$$F = m \cdot a, \tag{4.6}$$

represents Newton's Second Law of Motion under the assumption that the mass of the body does not change with time. Equation (4.6) is a clause and thus is separated by a preceding comma and a succeeding comma written into the equation directly. On the other hand, if the equation is part of a clause, commas are inappropriate. Consider the following example where by inserting the mass $m = 10\,\mathrm{kg}$ and the acceleration constant $a = 5\,\mathrm{m/s^2}$ into (4.6) yields

$$F = m \cdot a = 50\,\mathrm{N}. \tag{4.7}$$

Equation (4.6) is created with the following code:

```
\begin{equation}
F = m \cdot a,
\label{eq:NewtonsSecondLawOfMotion}
\end{equation}
```

It is referenced with `\Cref{eq:NewtonsSecondLawOfMotion}`, where the capital letter in `\Cref` is required because the reference is at the beginning of the sentence.

4.8.2 Multi-Line Equations with Multiple Numbers

Equations with multiple lines and multiple numbers are created with the `align` environment. It can be used either for equations that belong together as in the following example that represents the Maxwell Equations:

$$\vec{\nabla} \cdot \vec{E} = \frac{\rho}{\epsilon_0}, \tag{4.8}$$

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t}, \tag{4.9}$$

$$\vec{\nabla} \cdot \vec{B} = 0, \tag{4.10}$$

$$c^2 \vec{\nabla} \times \vec{B} = \frac{\partial \vec{E}}{\partial t} + \frac{\vec{j}}{\epsilon_0}, \tag{4.11}$$

or the `align` environment can be used for derivations as in the following example:

$$\begin{aligned} (a+b)^3 &= (a+b)^2(a+b), && (4.12)\\ &= (a^2+2ab+b^2)(a+b), && (4.13)\\ &= (a^3+2a^2b+ab^2)+(a^2b+2ab^2+b^3), && (4.14)\\ &= a^3+3a^2b+3ab^2+b^3. && (4.15) \end{aligned}$$

Note that in both sets of equations, (4.8)–(4.11) and (4.12)–(4.15), all lines end with either a comma or a period. This convention is part of a good writing style. Further note that each line is an equation with a unique number. If in exceptional cases these numbers are undesired, their

output can be suppressed, as illustrated in the following example:

$$\begin{aligned}(a+b)^3 &= (a+b)^2(a+b),\\ &= (a^2+2ab+b^2)(a+b),\\ &= (a^3+2a^2b+ab^2)+(a^2b+2ab^2+b^3),\\ &= a^3+3a^2b+3ab^2+b^3. \end{aligned} \tag{4.16}$$

As indicated by its name, the `align` environment causes the equations to be aligned nicely. Most of the time this alignment is desired to be about an equality sign or a comparison sign such as "=" or "⩾". However, the equations can be aligned wherever desired. In terms of typesetting, the position where the different lines should align is defined by the special LaTeX command `&`. Lines are separated by the LaTeX command `\\`.

The code of the above three sets of equations is given by

```
\begin{align}
\vec{\nabla} \cdot \vec{E} &= \frac{\rho}{\epsilon_0}, \label{eq: ↵
MaxwellGausssLaw} \\
\vec{\nabla} \times \vec{E} &= - \frac{\partial \vec{B}}{\partial t}, \label{ ↵
eq:MaxwellFaradaysLawOfInduction} \\
\vec{\nabla} \cdot \vec{B} &= 0, \label{eq:MaxwellGausssLawForMagnetism} \\
c^2\vec{\nabla} \times \vec{B} &= \frac{\partial \vec{E}}{\partial t} + \frac ↵
{\vec{j}}{\epsilon_0}, \label{eq:MaxwellAmperesCircuitalLaw}
\end{align}
```

for (4.8)–(4.11), and

```
\begin{align}
(a+b)^3 &= (a+b)^2(a+b), \label{eq:CubicSumStart} \\
&=(a^2+2ab+b^2)(a+b), \label{eq:CubicSumExansion1} \\
&=(a^3+2a^2b+ab^2)+(a^2b+2ab^2+b^3), \label{eq:CubicSumExpansion2} \\
&=a^3+3a^2b+3ab^2+b^3. \label{eq:CubicSumFinal}
\end{align}
```

for (4.12)–(4.15), and

```
\begin{align}
(a+b)^3 &= (a+b)^2(a+b), \nonumber \\
&=(a^2+2ab+b^2)(a+b), \nonumber \\
&=(a^3+2a^2b+ab^2)+(a^2b+2ab^2+b^3), \nonumber \\
&=a^3+3a^2b+3ab^2+b^3. \label{eq:CubicSumOnlyOneLabel}
\end{align}
```

for the last set with only one number (4.16), respectively.

4.8.3 Multi-Line Equations with Only One Number

If a set of equations should be represented by only one number, the following construct can be used:

```
\begin{equation}
\begin{aligned}
...
\end{aligned}
\end{equation}
```

The `aligned` environment has the same features as the `align` environment, with the only difference that it must be used within an `equation` environment. Compared to (4.8)–(4.11), the Maxwell equations can also be given by

$$\begin{aligned} \vec{\nabla} \cdot \vec{E} &= \frac{\rho}{\epsilon_0}, \\ \vec{\nabla} \times \vec{E} &= -\frac{\partial \vec{B}}{\partial t}, \\ \vec{\nabla} \cdot \vec{B} &= 0, \\ c^2 \vec{\nabla} \times \vec{B} &= \frac{\partial \vec{E}}{\partial t} + \frac{\vec{j}}{\epsilon_0}, \end{aligned} \tag{4.17}$$

which is referenced as (4.17) via the corresponding label directly after the end of the `aligned` environment. The corresponding code reads as follows.

```
\begin{equation}
\begin{aligned}
\vec{\nabla} \cdot \vec{E} &= \frac{\rho}{\epsilon_0}, \\
\vec{\nabla} \times \vec{E} &= - \frac{\partial \vec{B}}{\partial t}, \\
\vec{\nabla} \cdot \vec{B} &= 0, \\
c^2\vec{\nabla} \times \vec{B} &= \frac{\partial \vec{E}}{\partial t} + \frac ↵
{\vec{j}}{\epsilon_0}.
\end{aligned}
\label{eq:MaxwellEquations}
\end{equation}
```

4.8.4 Equations with No Numbers

Numbers of equations are suppressed by adding an asterisk (`*`) to the environment name. The following environments:

```
\begin{equation*}
...
\end{equation*}
```

and

```
\begin{align*}
...
\end{align*}
```

produce equations without numbers. However they can not be referenced, of course.

4.8.5 In-Text Equations

In order to print equations or mathematical expressions in text, the so-called math mode is used. Switching between math mode and text mode is toggled by the special character `$`. Accordingly, the symbol "λ" is printed by writing `$\lambda$`. Math mode allows the same equations to be written as with the environments `equation` and `align`. Hence, these equations are also known as in-line math equations. The only difference to regular equations is the fact that some mathematical operators appear smaller. This style is called the text style, as distinguished from the display style pursued in `equation` and `align` environments.

Note that punctuation marks should never be placed inside an in-line math equation, but rather after switching back to the text mode. Correspondingly, if a list of variables is written, each variable is contained in an separate in-line math equation, i.e., the first three letters of the Greek alphabet, α, β, γ, are written with `$\alpha$, $\beta$, $\gamma$`.

4.8.6 Essential Parts of Equations

There are a few constructs for mathematical equations that are specifically developed for certain purposes. Since it is very likely that LaTeX beginners are not familiar with these constructs and they usually try to come up with their own complicated implementation, the following paragraphs briefly explain some of the most important commands and environment. For further information the reader is referred to [52].

Cases are most easily produced with the environment `cases`, as for example in

$$y(t) = \begin{cases} x_{\min}(t) & \text{if } x(t) < x_{\min}(t), \\ x_{\max}(t) & \text{if } x(t) > x_{\max}(t), \\ x(t) & \text{otherwise}, \end{cases} \tag{4.18}$$

which saturates $x(t)$ such that $y(t) \in [x_{\min}(t), x_{\max}(t)]$. The corresponding code is depicted in the following:

```
\begin{equation}
y(t) = \begin{cases}
x_\mathrm{min}(t) & \text{if}\; x(t) < x_\mathrm{min}(t), \\
x_\mathrm{max}(t) & \text{if}\; x(t) > x_\mathrm{max}(t), \\
x(t) & \text{otherwise},\end{cases}
\label{eq:Saturation}
\end{equation}
```

Vectors and matrices in their normal form with parentheses and brackets are created via the environments `pmatrix` and `bmatrix`, respectively. As an example, the following state-space representation of a system with two state variables and one input:

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 2 \\ 1 \end{pmatrix} \cdot u(t), \tag{4.19}$$

is created with the following code:

```
\begin{equation}
\frac{\mathrm{d}}{\mathrm{d}t} \begin{pmatrix}
x_1(t) \\ x_2(t)
\end{pmatrix} = \begin{bmatrix}
1 & 0 \\ 0 & -1
\end{bmatrix} \cdot \begin{pmatrix}
x_1(t) \\ x_2(t)
\end{pmatrix} + \begin{pmatrix}
2 \\ 1
\end{pmatrix} \cdot u(t),
\label{eq:StateSpaceRepresentation}
\end{equation}
```

Instead of using parentheses for vectors we could have also used brackets. The result would then look as in the following. The decision for which style should be used is made by the supervisor or the author.

$$\frac{\mathrm{d}}{\mathrm{d}t} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{bmatrix} 2 \\ 1 \end{bmatrix} \cdot u(t). \tag{4.20}$$

Brackets can be adjusted to their content automatically. Use the construct `\left(` and `\right)`, where always an opening and a closing command are required. The parentheses `(` and `)` can be replaced by almost any reasonable symbol. The ones that are probably used most frequently are `[` and `]`, `{` and `}`, `\langle` and `\rangle`, and the ones that are identical on both sides, `|`, and `\|`. It is also possible to print only one bracket, left or right, by replacing the bracket symbol of the other side with a period, e.g. `\left.` and `\right\}` prints a right brace that is as large as required by the math expression between `\left.` and `\right\}`.

If in certain circumstances this method is not sufficient, the bracket size can also be defined manually by the commands `\big`, `\Big`, `\bigg`, and `\Bigg`, where each ensuing command creates a larger set of brackets. The commands are followed by the desired bracket, e.g. `\bigg{}`.

Operators such as the partial derivative "∂" or the argument of the minimum "arg min" are printed with special commands provided by the AMS packages. For example "∂" is printed via `\partial` and "arg min" is printed via `\arg\min`. Ensure that all operators are either special symbols or written in Roman letters. Please read section 4.6.2 for more information.

In addition, there are several rules that should be followed in order to profit the most out of the typesetting system.

- Use the command `\mid` in set descriptions, as for example in $\{x \mid x > 0\}$, produced with `$\{x \mid x > 0\}$`. Avoid using the keyboard symbol `|`, which results in $\{x|x > 0\}$.
- Use the command `\colon` for function definitions, as for example in $f(x)\colon x \to x^2$, produced with `$f(x) \colon x \to x^2$`. Avoid using the colon symbol `:` directly.
- Use the command `\dots` whenever three consecutive dots are required. Let LaTeX figure out by itself which type of dots are appropriate. Only if LaTeX fails, use `\ldots`, `\cdots`, etc.

4.9 Floats

Floats are containers for things that are not part of the regular text of a document. They are always printed as a whole entity and are therefore not breakable over several pages. They have a caption with a unique number, which they be referenced anywhere in the text. LaTeX knows by default the floats *figure* and *table*. During the compilation of the source code, LaTeX automatically computes the best location for all floats in a document. To some extent this decision can be influenced using the optional arguments (top, middle, bottom, left, right, page, here). However, in certain situations the only possibility is to move the definition of the float to an earlier or later point in the code.

The problems with floats most commonly arise with large figures or a great number of figures in a row. In both situations it can get difficult to adjust the final layout with only the methods mentioned above. In the first situation, LaTeX sometimes prints the figure together with only a few lines of the regular text stream on the same page. This can cause the reader to miss this text. In the second situation, LaTeX tries to print consecutively defined figures together at the end of the chapter. However, this can be unfavorable when the subsequent text addresses a different topic or the figures are used to clarify and visualize the text of a certain paragraph. The command `\FloatBarrier` from the package `placeins` can be used to force LaTeX to print all floats defined above this command.

In general, the automatic positioning of the floats can be controlled by optional arguments of the environments `figure`, `table`, and `algorithm`, where

- `t` tries to place the float on the top of a page,
- `m` tries to place the float in the middle of a page,
- `l` tries to place the float on the left of a page,
- `r` tries to place the float on the right of a page,
- `p` tries to place the float on a separate page, and

- `h` tries to place the float just where it is defined in the code.

To force LaTeX even further to place the float at the defined location, an exclamation mark can be added to the characters listed above.

4.9.1 Figures

Graphical representations are always helpful to explain the content of the written text. However, in order to get the full capability of a figure, the author has to think carefully about the representation. Often, several iterations are required to yield a final representation that best illustrates the aspects discussed, and actually helps the reader to quickly understand the material presented. The following items give a few hints for producing high-quality figures.

- When data is represented, use meaningful legends and axis labels.
- Always indicate the physical units of the axes.
- Avoid colors if possible.
- Restrict the content of a figure to one message.

Furthermore, once a satisfying representation is found, a proper format has to be chosen. The following list indicates some of the issues that need to be considered in this process.

- Use regular font sizes (usually 12 pts) for labels, legends, titles, etc.
- Use a consistent formatting of physical units for the axis labels, consider section 5.3 of [53].
- Write a caption with full sentences that explain the illustration.
- The graphics and the caption together are self-contained and self-explaining.

Finally, each figure has be mentioned in the text and explicitly referenced. Nonetheless, the text should be self-contained as well. As a result, it is not crucial if the figure is not at the perfect location in the text but is placed somewhere else by the float placement algorithm of LaTeX.

The float object of figures is produced by the `figure` environment. In order to center the graphic horizontally, either the command `\centering` or the `center` environment can be used. Regular images are included with the command `\includegraphics`, that is provided by the package `graphicx`. Captions of figures are always below the graphics. Therefore, the command `\caption` is used after the graphic is included. The label is set after the caption such that it refers to the correct figure. An example is given by figure 4.1, which shows the logo of IDSC. The corresponding code is given in the following:

```
\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{img/idsc.pdf}
\caption{Logo of the Institute for Dynamic Systems and Control (IDSC)  ↵
directly inserted as a PDF document.}
\label{fig:IDSClogoPDF}
\end{figure}
```

Figure 4.1: Logo of the Institute for Dynamic Systems and Control (IDSC) directly inserted as PDF document.

The included graphic `idsc.pdf` is located in the folder `img` which is on the same path as the root file of this document. The graphic is a PDF document, because the document is compiled with the TeX-program `pdflatex`. If instead the program `latex` and corresponding the DVI-PS-PDF chain

were used, the document would have to be given as an encapsulated PostScript (EPS). By using the package `epstopdf` it is also possible to include EPS graphics with the PDF chain, as shown in figure 4.2 which is produced by the following code. The package is not included in the document class IDSCreport but is loaded in the main file of this document, as can be seen in section 3.7.

```
\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{img/idsc.eps}
\caption{Logo of the Institute for Dynamic Systems and Control (IDSC) given ↵
as encapsulated PostScript document and automatically converted to a PDF ↵
document via the package \texttt{epstopdf}.}
\label{fig:IDSClogoEPS}
\end{figure}
```

Figure 4.2: Logo of the Institute for Dynamic Systems and Control (IDSC) given as encapsulated PostScript document and automatically converted to a PDF document via the package `epstopdf`.

For graphics other than just simple images the LaTeX community offers a variety of procedures to prepare graphics of third-party applications in order to include them easily in LaTeX documents. Appendix C provides suggestions for a selection of applications that are very often used in engineering.

4.9.2 Tables

Tables are an essential part of technical reports. They often present characteristic values of similar items or parameters of technical components to provide the reader with a quick overview on the topic discussed. Furthermore, by separating data from the text, the reading flow is not disturbed. The dense form of the representation of a table allows more data to be printed than would be possible in textual form.

The document class IDSCreport includes three packages to facilitate nice tables. Apart from a great number of other packages available, the following packages extend the basic commands of LaTeX with the most popular features of table construction:

- `array` provides essential options for column styles etc.,
- `booktabs` enhances the visual quality of tables with additional commands,
- `multirow` allows cells to be created that span more than one row.

A floating table always consists of two environments. The `table` environment produces the float object which encloses the caption, the label, and the actual table. The latter requires the `tabular` environment. In contrast to figures, the caption is placed above the actual table. This convention prevails because tables contain written text and are therefore recognized differently by our brains than images. In fact, the reader intuitively scans text from left to right and from top to bottom. Consequently, the caption should represent the content below in form of the title of the table. It is written in title case style as described by the capitalization rules of section 2.3.2.

As an example, table 4.2 shows the key characteristics of the Worldwide light-duty test cycles (WLTC) that have been developed within the Worldwide harmonized Light vehicles Test Procedure (WLTP) [54]. The test procedure features three different driving cycles to account for different vehicle types. In particular, based on the vehicle's rated power to unladen mass ratio, P_{mr}, the following classes are defined:

- Class 1 vehicles have a power to unladen mass ratio of $P_{\mathrm{mr}} \leqslant 22\,\mathrm{W/kg}$.
- Class 2 vehicles have a power to unladen mass ratio of $22\,\mathrm{W/kg} < P_{\mathrm{mr}} \leqslant 34\,\mathrm{W/kg}$.
- Class 3 vehicles have a power to unladen mass ratio of $P_{\mathrm{mr}} > 34\,\mathrm{W/kg}$.

Furthermore, there are two versions of the driving cycle for class 3 vehicles. Based on the maximum velocity, $v_{\max}$, the two subclasses are the following:

- Class 3a vehicles have a maximum speed of $v_{\max} < 120\,\mathrm{km/h}$.
- Class 3b vehicles have a maximum speed of $v_{\max} \geqslant 120\,\mathrm{km/h}$.

Table 4.2: Worldwide light-duty test cycles (WLTC)

Vehicle Class	Phase	Duration	Distance	Percentage of stop time	Maximum speed	Average speed with stops	Average speed without stops
		in s	in m	in %	in km/h	in km/h	in km/h
1	Low_1	589	3330	25.8	49.1	20.3	27.4
	Medium_1	433	4767	10.9	64.4	39.6	44.5
	WLTC	1022	8098	19.5	64.4	28.5	35.4
2	Low_2	589	3101	26.3	51.4	18.9	25.7
	Medium_2	433	4737	11.1	74.7	39.4	44.3
	High_2	455	6792	6.6	85.2	53.7	57.5
	Extra High_2	323	8019	2.2	123.1	89.4	91.4
	WLTC	1800	22649	13.3	123.1	45.3	52.2
3a	Low_3	589	3095	25.5	56.5	18.9	25.3
	$\mathrm{Medium}_{3\text{-}1}$	433	4721	11.1	76.6	39.3	44.1
	$\mathrm{High}_{3\text{-}1}$	455	7124	6.6	97.4	56.4	60.3
	Extra High_3	323	8254	2.2	131.3	92.0	94.0
	WLTC	1800	23194	13.1	131.3	46.4	53.3
3b	Low_3	589	3095	25.5	56.5	18.9	25.3
	$\mathrm{Medium}_{3\text{-}2}$	433	4756	11.1	76.6	39.5	44.5
	$\mathrm{High}_{3\text{-}2}$	455	7162	6.6	97.4	56.7	60.7
	Extra High_3	323	8254	2.2	131.3	92.0	94.0
	WLTC	1800	23266	13.1	131.3	46.5	53.5

These four cycles are each split up into separate speed phases which are called *low phase*, *medium phase*, *high phase*, and *extra high phase*. The WLTC for class 1 vehicles consists of phase Low_1 and phase Medium_1. The WLTC for class 2 vehicles consists of the phases Low_2, Medium_2, High_2, and Extra High_2. The WLTC for class 3a vehicles consists of the phases Low_3, $\mathrm{Medium}_{3\text{-}1}$, $\mathrm{High}_{3\text{-}1}$, and Extra High_3. The WLTC for class 3b vehicles consists of the phases Low_3, $\mathrm{Medium}_{3\text{-}2}$, $\mathrm{High}_{3\text{-}2}$, and Extra High_3. All low-speed phases last 589 seconds. All medium-speed phases last 433 seconds. All high-speed phases last 455 seconds. All extra-high-speed phases last 323 seconds.

Table 4.2 is produced by the code below, on which a few remarks shall be given here. The package `booktabs` provides commands for producing helpful horizontal lines. The upper line is produced with `\toprule`, the middle lines are produced with `\midrule`, and the lower line is produced with `\bottomrule`. The extra space between the individual vehicle classes is produced with `\addlinespace`, which is also part of the `booktabs` package. Note that the table doesn't have any vertical lines. As a result, the design looks very clean and the content is captured quickly.

The columns are special in the way that some of them have a fixed width. In doing so, the text of the first row is split up into several lines automatically. The first column has a fixed width of 1 cm and is left-aligned. The second column is a regular left-aligned column such that the text is not

broken over several lines. However, in order to save horizontal space, the inter-column space on the right of this column is suppressed with `@{}`. The remaining columns are all fixed in width but are aligned horizontally. A simple way to do this is by using the command `\newcolumntype` from the package `array` in combination with the construct `<{\centering}`. The corresponding line in the code below defines a new column type `C` that has a fixed given width and is centered. If on the other hand not all cells of a certain column should show this behavior, the command `\multirow` of the package with the same name is helpful.

For more information please consider the documentations of the mentioned packages [55, 56, 57], all of which are also included in the distribution of this template. A lot of additional information on commands for creating tables is given by *Wikipedia* [58]

```
\begin{table}[ht]
\centering
\caption{Worldwide light-duty test cycles (WLTC)}
\label{tab:WLTC}
\newcolumntype{C}[1]{p{#1}<{\centering}}
\begin{tabular}{p{1cm} l@{} C{1.2cm} C{1.2cm} C{1.2cm} C{1.5cm} C{1.5cm} C ↵
{1.5cm}}
\toprule
Vehicle Class & Phase & Duration & Distance & Percen\-tage of stop time &  ↵
Maximum speed & Average speed with stops & Average speed without stops \\
\midrule
& & in s & in m & in \% & in km/h & in km/h & in km/h \\
\midrule
1 & Low\textsubscript{1} & 589 & 3330 & 25.8 & 49.1 & 20.3 & 27.4 \\
& Medium\textsubscript{1} & 433 & 4767 & 10.9 & 64.4 & 39.6 & 44.5 \\
& WLTC & 1022 & 8098 & 19.5 & 64.4 & 28.5 & 35.4 \\
\addlinespace[1ex]
2 & Low\textsubscript{2} & 589 & 3101 & 26.3 & 51.4 & 18.9 & 25.7 \\
& Medium\textsubscript{2} & 433 & 4737 & 11.1 & 74.7 & 39.4 & 44.3 \\
& High\textsubscript{2} & 455 & 6792 & 6.6 & 85.2 & 53.7 & 57.5 \\
& Extra High\textsubscript{2} & 323 & 8019 & 2.2 & 123.1 & 89.4 & 91.4 \\
& WLTC & 1800 & 22649 & 13.3 & 123.1 & 45.3 & 52.2 \\
\addlinespace[1ex]
3a & Low\textsubscript{3} & 589 & 3095 & 25.5 & 56.5 & 18.9 & 25.3 \\
& Medium\textsubscript{3-1} & 433 & 4721 & 11.1 & 76.6 & 39.3 & 44.1 \\
& High\textsubscript{3-1} & 455 & 7124 & 6.6 & 97.4 & 56.4 & 60.3 \\
& Extra High\textsubscript{3} & 323 & 8254 & 2.2 & 131.3 & 92.0 & 94.0 \\
& WLTC & 1800 & 23194 & 13.1 & 131.3 & 46.4 & 53.3 \\
\addlinespace[1ex]
3b & Low\textsubscript{3} & 589 & 3095 & 25.5 & 56.5 & 18.9 & 25.3 \\
& Medium\textsubscript{3-2} & 433 & 4756 & 11.1 & 76.6 & 39.5 & 44.5 \\
& High\textsubscript{3-2} & 455 & 7162 & 6.6 & 97.4 & 56.7 & 60.7 \\
& Extra High\textsubscript{3} & 323 & 8254 & 2.2 & 131.3 & 92.0 & 94.0 \\
& WLTC & 1800 & 23266 & 13.1 & 131.3 & 46.5 & 53.5 \\
\bottomrule
\end{tabular}
\end{table}
```

Writing tables with a lot of entries can somehow be really painful. Therefore, several tools have been developed, two of which are mentioned here. The Excel Plugin Excel2LaTeX [59] allows a selected range of cells to be exported from Excel to LaTeX code. Another interesting tool is the website Tables Generator [60] that lets you create tables graphically.

4.9.3 Algorithms

Similarly to tables, algorithms require two kinds of environments. The first environment defines the floating object, and the second environment allows the algorithm to be defined. A float for an algorithm is produced by the environment `algorithm` that is provided by the package with the same name. For typesetting the actual algorithm several packages exist, of which the package `algorithmicx` [61] provides the best implementation and is thus the one that is included and recommended by this document class. On the other hand, its predecessor, the package `algorithmic`, features very convenient commands to produce the algorithm. Therefore, a third package `algpseudocode` is included that enables the use of the syntax of `algorithmic` in combination with the superior features `algorithmicx`.

As a result, the syntax required for writing algorithms is defined by the `algorithmic` package. This package is part of the `algorithms` bundle [62], which is included in the distribution of this template. The documentation of the `algorithmicx` package is included as well. However, the syntax is given by the `algorithmic` package, as mentioned above. The benefit of using this indirect way is that the `algorithmicx` package enables the full control over the definitions of the commands.

A typical algorithm is produced by an `algorithm` environment that encloses the caption, the label, and the `algorithmic` environment. Consider the example given in algorithm 4.1 that depicts Dijkstra's algorithm for finding the shortest paths from source node s to any other node of a directed graph G. The interested reader is referred to [63] for more information.

Algorithm 4.1 Dijkstra's Shortest Path Algorithm

```
function DIJKSTRA(G, s)
    for each node v in Graph G do                          ▷ Initialization
        dist[v] ← ∞                                        ▷ distance from source s to node v
        prev[v] ← undefined                                ▷ Previous node in optimal path from s
        add v to node set Q
    end for
    dist[s] ← 0
    while Q is not empty do
        u ← node v in Q with min. dist[v]                  ▷ new source node, will be s in 1st iteration
        remove u from Q
        for each neighbor v ∈ Q of u do
            d ← dist[u] + length[u, v]
            if d < dist[v] then                            ▷ a shorter path to v has been found
                dist[v] ← d
                prev[v] ← u
            end if
        end for
    end while
    return dist, prev                                      ▷ distance and path to any node in G from s
end function
```

4.10 Nomenclature

The nomenclature is created automatically once you have assigned symbols or subscripts. The process of creating a sorted list of abbreviations and mathematical symbols works analogously to the bibliography in that in both situations an additional compilation step is required. Whereas for the bibliography the corresponding LaTeX program `bibtex` is needed, the program for the nomenclature is called `makeindex`. See section 3.8.2 for more information. It is also possible to define these commands in other LaTeX editors, sometimes with much more effort, though. Please contact your supervisor if you need any assistance.

The predefined nomenclature offers five different lists, i.e.,

- a list for acronyms and abbreviations,
- a list for mathematical symbols,
- a list for subscripts,
- a list for superscripts, and
- a glossary.

In principle, entities are added to the nomenclature wherever they are used in the text. Thereby, the description of the abbreviation or symbol is added directly and thus is connected to the text. The current implementation offers for each category two possibilities to add entries, which are described in the following paragraphs.

Acronyms and Abbreviations are added via the commands `\newabbr` and `\typeabbr`. The first command is used if a abbreviation is added to the nomenclature but is not directly written in the text. Accordingly, the command can basically be used anywhere in the text. The second command prints the abbreviation directly where the command is used. Both commands require two input arguments, where the first argument is the abbreviation and the second argument is its description. In the following example, the abbreviation abbr. is added to the nomenclature via

```
\typeabbr{abbr.}{Abbreviation}
```

Mathematical Symbols are added via the commands `\newsymb` and `\typesymb`. Similarly to the acronyms and abbreviations, the first command adds entries without printing them into the text directly. But, in contrast to the latter category, both commands require a third input argument that describes the physical unit of the variable. Therefore, to add the distance variable s to the nomenclature the following example command

```
\typesymb{$s$}{Distance variable}{\unit{m}}
```

should be used. Note that the physical units have to be given via the command `\unit` or `\unitfrac` in order to appear properly. More information on this command is found in section 4.7.

Subscripts are added via the commands `\newsub` and `\typesub`. Similarly to acronyms and abbreviations, `\typesub` prints the subscript and adds it to the nomenclature at the same time. The code which adds the subscript k for an iteration index to the nomenclature is given by

```
\typesub{$k$}{Iteration index}
```

Superscripts are added via the commands `\newsuper` and `\typesuper`. Similarly to acronyms and abbreviations, `\typesuper` prints the superscript and adds it to the nomenclature at the same time. The code which adds the superscript $\star$ for optimal solution to the nomenclature is given by

```
\typesuper{$\star$}{Optimal solution}
```

Glossary content is added via the commands `\newgloss` and `\typegloss`. Similarly to acronyms and abbreviations, `\typegloss` prints the glossary reference and adds it to the nomenclature at the same time. The code which adds the glossary entry "glossary" to the nomenclature is given by

```
\typegloss{glossary}{A collection of textual glosses or of specialized terms ↵
with their meanings, \cite{merriam-webster:glossary}.}
```

4.11 Including Code in Your Document

Code is most conveniently included via the `lstlistings` environment provided by the `listings` package [64], whose manual is also included in the distribution of this template. This environment allows the given code to be formatted in practically every programming language. Furthermore, it provides a great number of commands to define new styles and languages. The document class IDSCreport currently provides two proprietary styles.

- The style `plaincode` inserts the code as raw text in typewriter font.
- The style `matlabcode` formats the keywords as nicely as in the Matlab editor.

Both styles enable automatic line breaks such that the code does not exceed the width of the text. Overflows are indicated by the symbol "↵", as shown clearly in section 3.7, for example.

The style of the code is declared via the optional argument `style`. For plain code, as all LaTeX commands presented in this template, the `lstlisting` environment is used as follows:

```
\begin{lstlisting}[style=plaincode,xleftmargin=1em]
.
\end{lstlisting}
```

For Matlab code, the environment is used as follows:

```
\begin{lstlisting}[style=matlabcode,xleftmargin=1em]
.
\end{lstlisting}
```

Note that `xleftmargin=1em` creates a nice horizontal space on the left side of the code. In this way, the code block is easier and faster to grasp by the reader. A code example may look like this:

```
% Evaluate y = 2x
for i = 1:length(x)
   y(i) = 2*x(i);
end
```

The example is created using the following Matlab code:

```
\begin{lstlisting}[style=matlabcode,xleftmargin=1em]
% Evaluate y = 2x
for i = 1:length(x)
  y(i) = 2*x(i);
end
\end{lstlisting}
```

If the entire content of a file should be included, as shown for example in section 3.7, the following command from the `listings` package is your friend:

```
\lstinputlisting[style=matlabcode]{mycode.m}
```

Here, the horizontal space could also be invoked via `xleftmargin=1em`. Including files directly ensures that the code is always up-to-date. However, it is usually not helpful to include entire files within the regular content of the report. If necessary, entire files should rather be included in the Appendix and thus are only referenced in the text. Anyway, most of the time it is much more useful to represent the code as pseudo-code instead. Consider section 4.9.3 for this purpose.

4.12 Bibliography

It is highly recommended to use the IEEEtran BibTeX citation style defined by the Institute of Electrical and Electronics Engineers (IEEE). The document class IDSCreport is implemented

accordingly and thus expects this citation style by default. The bibliography style defines new entry types compared to the basic styles, e.g., plain, unsrt, ieeetr, etc. It also extends the available fields that can be used to specify the documents cited. However, it is recommended to use only the following entry types and complete all given fields. If any additional entry types are required[6], the corresponding documentation [41] should be consulted. For convenience, it is also included in the distribution of this template, named `IEEEtran_bst_HOWTO.pdf`.

The cite key should consist of the last name of the main author (first letter capitalized), and the year of publication, separated by a colon. To avoid any ambiguities when the same author has published multiple documents in the same year, two random letters are appended to the cite key. For student reports the appendices `sm`, `bt`, `sp`, `mt`, and `diss` can be used to indicate Studies on Mechatronics, Bachelor's thesis, semester project, Master's thesis, and dissertation, respectively. As an example, `Ritter:2014mt` is an appropriate cite key for the Master's thesis of Ritter, published in 2014.

4.12.1 Journals and Magazines

`@article` is used to reference journal articles. The required fields are author, title, journal, and year. Additionally supported fields are language, volume, number, pages, month, note, and url. We recommend that you use and complete the following fields:

```
@article{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
journal = <use an abbreviation as defined in IEEEabrv.bib>,
volume = {},
number = {},
pages = {},
year = {}
}
```

A proper citation of an article is given by [65].

4.12.2 Books

`@book` is used to reference entire books that are formally published. The required fields are author and/or editor, title, publisher, and year. Additionally supported fields are language, edition, series, address, month, volume, number, note, and url. We recommend that you use and complete the following fields:

```
@book{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
editor = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
edition = {},
address = {},
publisher = {},
year = {}
}
```

Books appear in the bibliography as shown by the citation [66].

[6] Additional entry types of the IEEEtran BibTeX style that are not covered by this section are `@periodical`, `@inbook`, `@conference`, `@proceedings`, `@online`, `@internet`, `@www`, `@patent`, `@standard`, and `@misc`.

@booklet is used to reference complete books that are not formally published (by a publisher). Mostly books available only on the Internet fall into this category. The only required field is title. Additionally supported fields are author, language, howpublished, organization, address, month, year, note, url. We recommend that you use and complete the following fields:

```
@booklet{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>}
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
howpublished = {<similar to publisher, e.g. university press>}
year = {}
}
```

Booklets appear as shown by [67].

@incollection is used to reference a chapter of a book that has its own title (and its own author). The required fields are author, title, booktitle, year. Additionally supported fields are language, edition, series, editor, address, publisher, month, volume, number, chapter, type, pages, note, and url. We recommend that you use and complete the following fields:

```
@incollection{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
editor = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
booktitle = {<use {XX} to preserve capitalization of acronyms or german nouns ↵
>},
language = {<english/german/...>},
address = {},
publisher = {},
year = {}
}
```

The properly cited in-collection [68] is given in the bibliography.

4.12.3 Conference Proceedings

@inproceedings is used to reference papers in conference proceedings. The required fields are author, title, booktitle, and year. Additionally supported fields are intype, language, series, editor, volume, number, organization, address, publisher, month, paper, type, pages, note, and url. We recommend that you use and complete the following fields:

```
@inproceedings{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
booktitle = {<use {XX} to preserve capitalization of acronyms or german nouns ↵
>},
language = {<english/german/...>},
publisher = {},
pages = {},
year = {}
}
```

In-proceedings are shown as the citation [69].

4.12.4 Student Reports

@mastersthesis is used to reference all kinds of student reports except for dissertations, i.e. Studies on Mechatronics, Bachelor's theses, semester projects, and Master's theses. The required fields are author, title, school, and year. Additionally supported fields are language, type, address, month, note, url. We recommend that you use and complete the following fields:

```
@mastersthesis{nethz:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
type = {<Studies on mechatronics/Bachelor's thesis/Semester project/Master's ↵
thesis>},
school = {ETH Zurich, Institute for Dynamic Systems and Control},
address = {Z\"{u}rich, Switzerland},
month = <jan/feb/mar/apr/may/jun/jul/aug/sep/oct/nov/dec>,
year = {YEAR},
note = {{IDSC-??-??-??}}
}
```

Citations of a Studies on Mechatronics, a Bachelor's thesis, a semester project, and a Master's thesis are given by [70, 71, 72, 73].

@phdthesis is used to reference dissertations. The required fields are author, title, school, and year. Additionally supported fields are language, type, address, month, note, url. We recommend that you use and complete the following fields:

```
@phdthesis{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
school = {ETH Zurich, Institute for Dynamic Systems and Control},
address = {},
month = <jan/feb/mar/apr/may/jun/jul/aug/sep/oct/nov/dec>,
year = {},
note = {{Diss. ETH No. ?????}}
}
```

A dissertation should appear in the bibliography similarly to the one that is given in [74].

4.12.5 Technical Reports

@manual is used to reference technical documentations such as those furnished with software programs for example. The only required field is title. Additionally supported fields are author, title, language, edition, howpublished, organization, address, month, year, note, and url. We recommend that you use and complete the following fields:

```
@manual{Name:YEARxx,
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
howpublished = {<Version of software or the like>},
organization = {},
year = {}
}
```

Manuals are technical documentations for software programs such as Matlab [75].

@techreport is used to reference all kinds of technical reports that do not fit any of the other entry styles. The required fields are author, title, institution, and year. Additionally supported fields are language, howpublished, address, number, type, month, note, and url. We recommend to use and complete the following fields:

```
@techreport{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
institution = {},
month = <jan/feb/mar/apr/may/jun/jul/aug/sep/oct/nov/dec>,
year = {}
}
```

Technical reports as [76] are documents from governmental institutions, e.g. the Federal Statistical Office.

4.12.6 Web Pages

@webpage is used to reference all kinds of content that is only available on the Internet. There are no required fields. However, supported fields are author, month, year, title, language, howpublished, organization, address, note, and url. We recommend to use and complete the following fields:

```
@webpage{Name:YEARxx,
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
language = {<english/german/...>},
organization = {},
month = <jan/feb/mar/apr/may/jun/jul/aug/sep/oct/nov/dec>,
year = {},
url = {}
}
```

Web pages appear in the bibliography with their uniform resource locator (URL), as for example shown in this very important web page [77].

4.12.7 Software

Software and software packages should be referenced via their manuals as described in section 4.12.5. However, for software that is primarily distributed online, [41] mentions that the following non-standard IEEEtran entry type may be used.

@electronic may be used to reference software and software packages that are primarily distributed online and that do not have a manual that is suitable for a corresponding reference. There are no required fields. The supported fields are author, title, language, howpublished, organization, address, month, year, note, and url. We recommend to use and complete the following fields:

```
@electronic{Name:YEARxx,
author = {<[first name] [last name] and [first name] [last name] and ...>},
title = {<use {XX} to preserve capitalization of acronyms or german nouns>},
note = {<version number>},
organization = {},
month = <jan/feb/mar/apr/may/jun/jul/aug/sep/oct/nov/dec>,
year = {},
url = {}
}
```

Appendix A

English Grammar

English is the universal language of science. Almost every relevant publication is written in English or at least translated into English. Especially since the Internet serves as the one major communication channel and information provider, English is the most dominant language in science and industry. However, the proper use of grammar and writing style is often difficult, particularly for nonnative speakers. Therefore, this supplementary chapter aims at providing additional information on English grammar and primarily extends sections 2.3.2 and 4.2.

A.1 Compound Words

Compound words are a combination of multiple words that result in a single meaning. Although there are only a few important rules to follow it is not always clear whether the single words are written together or connected by a hyphen, or even written as separate words. Table A.1 lists the most common compound words in the correct (or recommended) way. For other words please consult *Merriam-Webster's Collegiate Dictionary* [78], also available at `http://www.merriam-webster.com`.

Table A.1: Common compound words.

accident-free	object-oriented	self-sustaining
air-to-fuel ratio	online	so-called
dynamometer	on-screen	test bench
e-book	nonlinear	three-way catalyst
e-mail	northeast	web page
everlasting	off-line	web-related
ever-recurring	one-half	website
full-length	self-learning	well-known

A.2 Differences Between British and American English

The following sections and paragraphs try to clarify the small and not so small variations. The structure and the explanations are based on the detailed web page "American and British English Spelling Differences" [79] of *Wikipedia*.

A.2.1 Spelling

Word Endings

-our*, *-or differences derive from the adoption of Latin words to Old French after the Norman conquest of England. As a result, many endings with *-or* from Latin became *-our* in British English, e.g., *colour*, *flavour*, *behaviour*, *neighbour*, etc. American English retained the Latin endings, which causes most word endings in *unstressed -our* to end in *-or*, e.g. *color*, *flavor*, *behavior*, *neighbor*, etc. However, the spelling is the same whenever the vowel is unreduced in pronunciation, as for example in *contour*. A more extensive list of words with different spelling is given in [80].

-re*, *-er differences address words from French, Latin or Greek that end with a consonant followed by an unstressed non-rhotic accent. In British English, these words are written with a *-re* ending, e.g., centre, litre, metre, manoeuvre, spectre, calibre, etc. In American English, almost all of these endings have become *-er*, e.g. center, liter, meter, maneuver, specter, caliber, etc. Note that there are a few exceptional words that use the *-re* ending in both British English and American English.

-ce*, *-se differences appear not only between British English and American English, but also as the noun–verb distinction, where the pronunciation is a soft *s* for nouns and a sharp *s* for verbs. American English dropped this distinction for certain words, hence *license* and *practice* are for example spelled identically for both nouns and verbs. On the other hand, some nouns are spelled with *-se* in American English, e.g., *defense* and *offense*. British English kept the original spelling distinction.

***-ise*, *-ize* (*-isation*, *-ization*)** are both used in British English, where various style guides follow different rules. For these words, American English always uses the endings *-ize* and *-ization*. There are a few exceptions, though, where *-ise* is used worldwide, i.e., *advertise*, *advise*, *arise*, *circumcise*, *comprise*, *compromise*, *demise*, *despise*, *devise*, *disguise*, *excise*, *exercise*, *franchise*, *guise*, *improvise*, *incise*, *revise*, *rise*, *supervise*, *surmise*, *surprise*, *televise*, and *wise*.

-yse*, *-yze follow a clear distinction, where the ending *-yse* is British and the ending *-yze* is American.

-ogue*, *-og are both used in American English, although besides certain exceptions the *-ogue* endings prevail. The exceptions are the noun *catalog* and the adjective *analog*. Both forms, analog and analogue, are used for the noun. Another critical word is *dialogue*, which is still preferred over dialog.

ae and oe appear only in British English, e.g. manoeuvre. The American counterparts are written only with a single *e*.

Doubled consonants

The final consonants of words are sometimes doubled in both American and British English when a suffix is added. This is usually the case when the final syllable is stressed and the word ends with a lone vowel followed by a lone consonant, e.g., *compelled*, *excelling*, *propelled*, *rebelling*. On the other hand, the second *l* is usually dropped when prefixes or suffixes are used, e.g., *useful* derives from *full*, *always* and *altogether* derive from *all*, *welfare* and *welcome* derive from *well*, etc.

In British English, double consonants are often used even though the final syllable is unstressed. This is the case for all inflections (*-ed*, *-ing*, *-er*, *-est*) and for the noun suffixes *-er* and *-or*, but not for the endings *-ize*/*-ise*, *-ism*, *-ist*, *-ish*. A few examples are *cancelled*, *counsellor*, *cruellest*, *labelled*, *modelling*, *quarrelled*, *signalling*, *traveller*, and *travelling*. In American English these words are written with a single *l* only.

In American English, double consonants appear in certain words such as *will*, *skill*, *fill*, *roll*, *stall*, *still*, etc. As a result, also compound words contain double consonants, e.g. *willful*, *skillful*, *fulfill*, *fulfillment*, *enroll*. In contrast, British spelling uses only one consonant.

Dropped *e*

In contrast to British English, American English drops the silent *e* whenever it is not needed. Thus, the British words *ageing*, *routeing*, *likeable*, *liveable*, *rateable*, *saleable*, and *sizeable* become *aging*, *routing*, *likable*, *livable*, *ratable*, *salable*, and *sizable*. However, both spellings use *breathable*, *curable*, *lovable*, *movable*, *notable*, *provable*, *quotable*, *scalable*, *solvable*, *usable*. Moreover, the silent *e* is kept when it is needed to preserve a soft *c*, *ch*, or *g*, e.g., *traceable*, *cacheable*, *changeable*, etc., and it is kept after *-dge*, as in *knowledgeable*.

A.2.2 Vocabulary

British English and American English not only differ in spelling but often use different vocabulary for the same things. Classic examples are anticlockwise vs. counterclockwise, postbox vs. mailbox, full stop vs. period, etc.

Table A.2: Different automotive words in British English and American English.

Br. English	Am. English	Br. English	Am. English
boot (of a car)	trunk	milometer	odometer
breakdown van	tow truck	number plate	license plate
car park	parking lot	petrol	gas *or* gasoline
drink-driving	drunk driving	racing car	race car
driving licence	driver's license	registration plate	license plate
dual carriageway	divided highway	roundabout (in road)	traffic circle
estate car	station wagon	tram	streetcar *or* cable car
gearbox	transmission	tyre	tire
gear lever	gearshift	verge (of a road)	shoulder
lorry	truck	windscreen	windshield
manoeuvre	maneuver	wing (of a car)	fender

Interestingly, there are quite a few different words related to automotive applications, e.g. petrol vs. gasoline or gas, gearbox vs. transmission, etc. Table A.2 list a selection of words that are related to automotive applications. Table A.3 consists of nonautomotive words, mostly taken from [81]. Table A.4 consists of expressions that are spelled or formulated differently.

Table A.3: Non-automotive words in British English and American English.

Br. English	Am. English	Br. English	Am. English
anticlockwise	counterclockwise	postbox	mailbox
earth (electrical)	ground	postcode	zip code
first floor	second floor	programme	program
full stop (punct.)	period	public transport	public transportation
grey	gray	queue	line
ground floor	first floor	railway	railroad
lift	elevator	storey	story
maths	math	tick	check mark
mobile phone	cell phone	tonne	ton
pedestrian crossing	crosswalk	underground	subway

Moreover, British English and American English differ in the writing style of expressions, e.g. orientated vs. oriented, to write to sb. vs. to write sb, etc. More expressions are listed in table A.4.

Table A.4: Different expressions of British English and American English.

Br. English	Am. English
dependant (noun)	dependent (noun)
orientated	oriented
to write to sb.	to write sb.
unalike	unlike

A.3 Emphasis Style for Names and Terms

Section 4.2.1 addresses the emphasis style of names and terms in a slightly compressed way. For this reason, this section provides a clearer representation of the given rules in form of table A.5. In case another subject has to be written, the author might consult chapter 8 of *The Chicago Manual of Style* [38] or just take a listed entry that best approaches the desired subject.

Table A.5: Emphasis style based on *The Chicago Manual of Style*.

Subject	Emphasis
Articles	Quotes
Blog entries	Quotes
Blogs	Italics
Books	Italics
Book series	Neither
Book editions	Neither
Chapters	Quotes
Computer software	Neither
Conferences	Neither
Drawings	Italics
Encyclopedias	Italics
Essays	Quotes
Handbooks	Quotes
Journals	Italics
Lecture series	Neither

Subject	Emphasis
Lectures (individual)	Quotes
Magazines	Italics
Meetings	Neither
Movies	Italics
Newspapers	Italics
Periodicals	Italics
Photographs	Italics
Podcast episodes	Quotes
Podcasts	Italics
Reports	Italics
Unpublished works	Quotes
Video blogs	Italics
Video-blog episodes	Quotes
Web pages and sections	Quotes
Websites	Neither

Appendix B

Further Literature for Learning LaTeX

Working with LaTeX can sometimes be tedious and cumbersome. Especially if the author wants to tweak the default definition of the document structure or its layout, a deeper knowledge of the implementation is required. On the other hand, if the purpose of the document is a technical report or article where the content is far more important than the layout, LaTeX is probably the most suitable typesetting system available. Still, for some specific elements of scientific reports it is occasionally desired by the authors to have certain design flexibilities. Therefore, this chapter provides additional references to give back the freedom of manipulating the workings of LaTeX, which definitely extends beyond the scope of the present document.

B.1 General Knowledge

One of the best manuals for authors is the *The Not So Short Introduction to LaTeX 2ε* [82] and its German translation *LaTeX 2ε-Kurzbeschreibung* [83]. Both documents are included in the distribution of this template.

Another valuable source of information is the book *More Math Into LaTeX* by George Grätzer [84]. Especially the appendices A and B are useful for short lists of math symbols and text symbols, respectively. The corresponding abstract of the book is included in the distribution of this template as `SymbolTables.pdf`. The current edition of the book is freely available for all ETH affiliates via the *SpringerLink* program of the publisher *Springer International Publishing AG*. It is also found via the library web page of ETH Zurich (`http://www.library.ethz.ch`). For those interested without access, the previous edition (fourth edition) is also available for free via the Comprehensive TeX Archive Network (CTAN) on `http://tug.ctan.org/info/Math_into_LaTeX-4/Short_Course.pdf`.

B.2 Symbols

An extended list of LaTeX symbols is given in *The Comprehensive LaTeX Symbol List* by Scott Pakin [44]. As this document is available for free and it comprises probably every symbol that is ever required to write any document, it is very often cited in LaTeX related forums and very easy to find on the Internet. Anyhow, the document is also included in the distribution of this template.

B.3 Bibliography

For more information on the bibliography style recommended by the IDSC guidelines, consider the manual *How to Use the IEEEtran BIBTEX Style* by Michael Shell [41]. It describes how the BibTEX style for IEEE transactions journals and conferences should be used in order to produce bibliographies that conform with these standards. In addition to the recommended entry types of section 4.12, the document covers a number of other entry types, which in some rare cases might be useful and more appropriate than the entry types presented earlier. The manual named `IEEEtran_bst_HOWTO.pdf` is also included in the distribution of this template.

Appendix C

Creating Attractive Graphics

Graphics are essential parts of technical reports. Therefore, it is only reasonable to invest an appropriate amount of the writing time into their development and polishing. The idea and motivation unfortunately is not sufficient for the ideal result. Often the knowledge of the different approaches and the experience of which solution qualifies best for the specific goal is missing. To overcome these problems, the following sections provide an overview on the various tools that can be used to create attractive and meaningful graphics and include them in the LaTeX document. Each section presents a different approach and discusses its advantages and drawbacks.

C.1 Plots with Matlab

Matlab is a very convenient software application for the illustration of data. It offers a variety of plotting functions for many different graph-based diagrams such as function graphs, scatter plots, bar graphs, pie charts, histograms, radial graphs, contour plots, and many more. Most of the so-called plot functions are available as two-dimensional plotting functions and three-dimensional plotting functions, whereas Matlab easily creates diagrams that contain data plots of any combination of the available plot functions. Furthermore, almost any Matlab toolbox offers additional plot functions that are specifically developed for the dedicated application field.

There are several ways to export Matlab plots to LaTeX. First of all, it is always possible to make a screen shot of the Matlab figure or export the figure as an image. Although this may seem to be the easiest way, it is also the one that is least recommended. Only in certain circumstances, an image is appropriate. This could for example be if the author wants to emphasize the appearance of the figure of the graphical user interface (GUI). Far better results are achieved by exporting the illustration as vector-based data. The formats EPS and PDF are compatible with the LaTeX typesetting engine. Finally, there is another less known approach that consists of exporting the data and the definition of the plot separately.

The following sections cover the two last-mentioned approaches, i.e., exporting the Matlab figures to vector-based formats (EPS or PDF), and exporting the data and the definition separately to independent file formats. Since the different build chains accept different graphic formats, as explained in section 3.8.1, the exporting approach has to be chosen accordingly. In case the DVI-PS-PDF chain is used, the Matlab plots have to be exported to EPS. In case the PDF chain is used, the plots can be exported to EPS or PDF. However, in both cases it is possible to export figures to *editable formats*, where the graphics are recreated by the typesetting system of LaTeX during the compilation process.

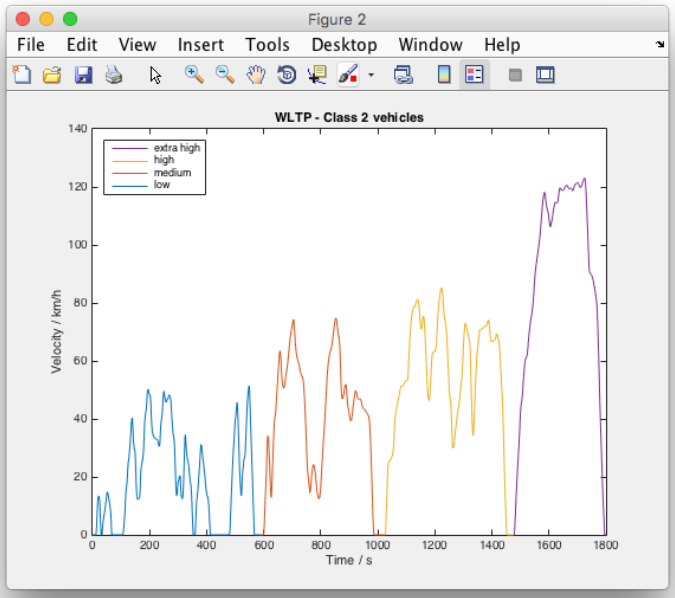


Figure C.1: Screen shot of the Matlab figure that shows the worldwide light-duty test cycle (WLTC) for class 2 vehicles. This representation method is only appropriate if the focus of the text is on the appearance of the figure in Matlab, which applies to the present case.

C.1.1 Manual Figure Export to EPS

The easiest way to export a Matlab figure to EPS is undoubtedly by just saving it as an EPS file via the menu tool bar, which yields a result as shown in figure C.2. Clearly, the annotations do not match the font of the text of the document, which does not look very professional. A better result is achieved by adjusting the font of the annotations in Matlab. The following code sets the font of all text elements to *Times New Roman* and the font size to 14 points. The last line saves the file as `filename.eps` with format `epsc` that stands for *Encapsulated PostScript (EPS) Level 3 color*.

```
allTextHandles = findall(gca, 'Type', 'text');
set([gca; allTextHandles], 'FontName', 'Times New Roman', 'FontSize', 14);
saveas(gcf, 'filename.eps', 'epsc');
```

Similarly to the EPS file exported via the menu tool bar this resulting EPS file is self-contained, and thus is included with the regular command `\includegraphics`, as shown in section 4.9.1. The font size is set to 14 points instead of the 12 points of this text because the figure is not included in its full size but is downscaled such that the width is equal to 70 % of the text width. The result is shown in figure C.3. Clearly, the font size of the annotations still does not match the font size of the regular text. Finding a better value is only possible with tedious trial and error. Far better results are obtained if the graphical elements and the annotations are exported separately and are recombined with designated commands of LaTeX. Corresponding approaches are discussed in the following sections.

Although Matlab offers the option to export the figure as a PDF document, which might at first sight be favorable for the PDF build chain, the figure should still be exported to EPS as described in the code above. The reason is that exporting it to PDF would produce a full-size A4 or US letter page with the graphic centered in the middle. Therefore, the better approach is to produce an EPS file and convert it automatically with the LaTeX package `epstopdf`.

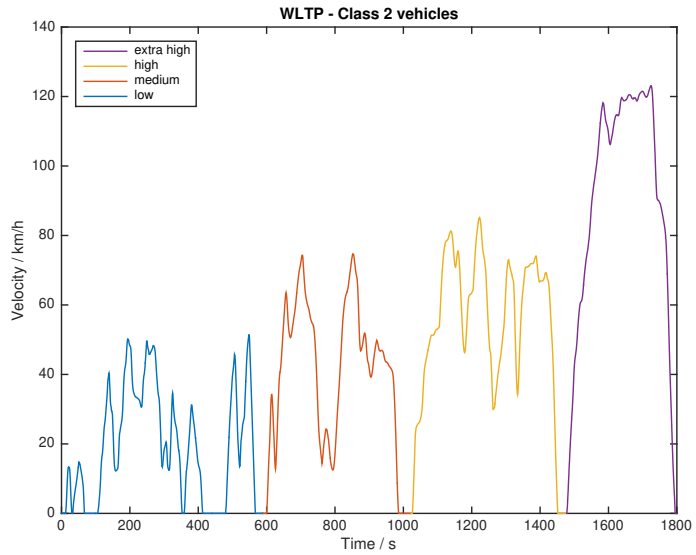


Figure C.2: The same Matlab figure as shown in figure C.1 simply exported to EPS via the menu tool bar. As expected, the annotations of the plot, such as the labels, the title, and the legend, do not exhibit the beautiful text font of LaTeX. Better results are achieved with more sophisticated approaches as shown in figures C.3 and C.4.

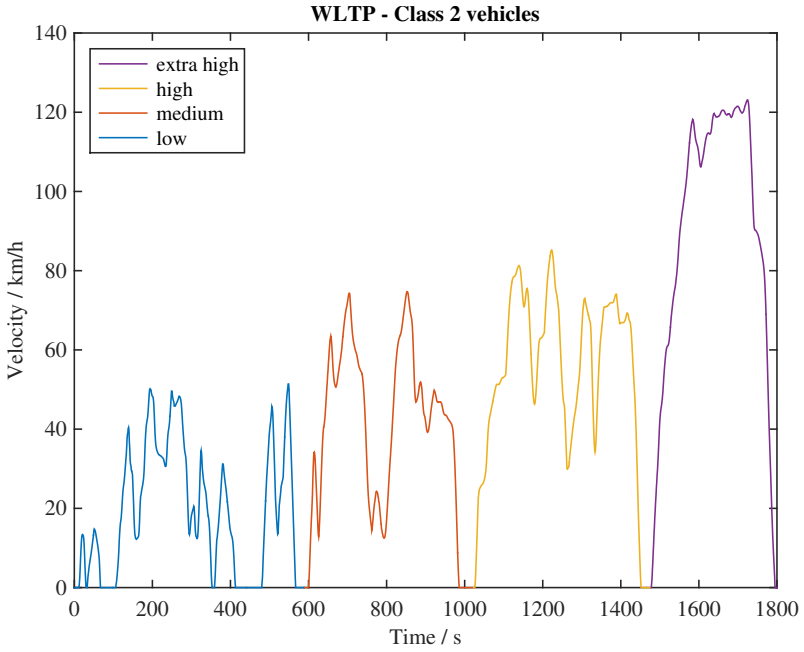


Figure C.3: The same Matlab figure as shown in figure C.1 but with the annotation and label font set to Times New Roman. Since the figure is not included in its full size, the text is set to have a font size of 14 points. Although still not ideal, this figure shows a remarkable improvement over figure C.2. However, even better results are achieved if LaTeX is left to take over the typesetting of the annotations.

C.1.2 Advanced Figure Export for DVI-PS-PDF Chain

More sophisticated tools for exporting Matlab figures separate the annotations from the graph, export each part individually, and then leave it to LaTeX to typeset the annotations of the figure back to the figure with the given font of the document.

There is a great number of tools and practices that allow Matlab figures to be *easily* converted to EPS, and allow the text to be separated from the graph. The following paragraphs provide an overview on the approaches that are most popular.

`laprint` is a Matlab function by the Control and Automation Department of Electrical and Computer Engineering of the University of Kassel [85]. The code is hosted by Matlab on its website File Exchange [86]. The TEX file created by `laprint.m` contains all textual elements of the figure such as titles, labels, legend entries, etc. For typesetting, the packages `graphicx`, `psfrag` and, for coloring, possibly `color` are required. Furthermore, it assumes that a PostScript driver such as `dvips` is used. Accordingly, other build chains apart from the DVI-PS-PDF chain will not yield proper results. The latest version is 3.16, released on September 2004. Since the graphics engine of Matlab was changed with the release 8.4 (R2014b) on October 3, 2014, `laprint` only works with earlier Matlab versions.

`matlabfrag` is similar to `laprint`, but it provides improved functionalities in various aspects. It is developed by Zebb Prime and comes with a user manual [87]. Both the code and the documentation are hosted on GitHub [88] and Matlab's website File Exchange [89]. The latest version is v0.6.16, released on April 2010. The Matlab function `matlabfrag.m` exports the graphic elements of the figure to an EPS file and the text elements to a TEX file. Accordingly, the DVI-PS-PDF chain should be the build chain of your choice. In addition, since the position of the annotations is determined via the font size of these annotations in Matlab, changing the figure size in LaTeX does not yield good results. Accordingly, the figure has to be adjusted in Matlab before it is exported to EPS. This can require a few iterative steps until the result is satisfactory.

C.1.3 Figure Export for PDF Chain

`mlf2pdf` is based on the command `matlabfrag` and calls the LaTeX program `pdflatex` internally via the command-line interface of the operating system. The Matlab function `mlf2pdf.m` is accessible through Matlab's website File Exchange [90]. and requires the Matlab function `matlabfrag.m`. Since the graphics engine of Matlab is changed by the release 8.4 (R2014b) of October 3, 2014, `mlf2pdf` can no longer export the legends properly. Moreover, the command-line interface command used by `mlf2pdf` does not work with MacOS version El Capitan since the programs stored in `usr/texbin` are no longer accessible for non-Apple software.

C.1.4 Exporting Figures to Editable Formats

`matlab2tikz` is the most promising candidate under the functions that convert Matlab figures into typesetting commands of LaTeX. The code is hosted by GitHub [91] and is regularly updated by several different contributors. It is also available via Matlab's website File Exchange [92]. The result of the Matlab function `matlab2tikz.m` is a TEX file that defines the complete plot including the data. The commands used for the plot creation are part of the LaTeX package `pgfplots` that is based on the graphics language Ti*k*Z. The resulting plot exported with `pgfplots` is shown in figure C.4. However, due to the enormous flexibility of the LaTeX package `pgfplots` the representation can be improved even further. Particularly advantageous is the fact that this fine-tuning is conducted in LaTeX without distorting the figure in Matlab. A possible result may look like figure C.5. For more information on the implementation please consult appendix C.2.

`plot2svg` converts 3D and 2D Matlab plots to the scalable vector format (SVG), which is then manually modifiable with designated software such as Inkscape (see appendix C.4). The Mat-

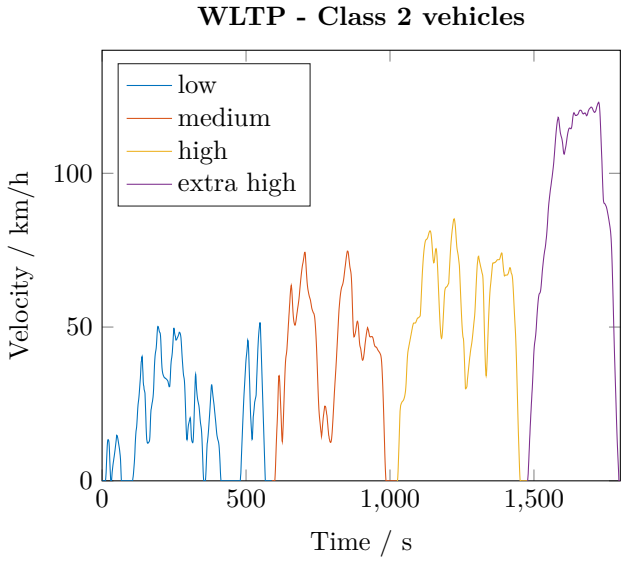


Figure C.4: The definition of this plot is produced with the Matlab function `matlab2tikz` with option `noSize` set to `true`. The plot is then typeset via the TEX file that this Matlab function created. Further adjustments of the layout are easily possible via the designated LaTeX commands of the package `pgfplots`.

lab function `plot2svg.m` and the corresponding files are available on Matlab's website File Exchange [93] or on GitHub [94]. It was written by Jürg Schwizer and was under continuous development until a few years ago. Unfortunately, since Matlab changed its graphics engine with the release 8.4 (R2014b) on October 3, 2014, the function no longer works. Although a small hack[1] recovers most of its functionality, it is for example not possible anymore to export the legends. As an alternative, Matlab offers the option to save figures as SVG files directly via the menu tool bar or via the function `saveas`. To get an idea of the appearance of such a customized solution, see appendix C.4.

C.2 Function Plots with PGFPlots

The LaTeX package `pgfplots` most probably is the best implementation to produce high-quality function plots in LaTeX. It offers a wide range of different plot commands to produce almost any kind of function plots including line plots, scatter plots, piecewise constant plots, bar plots, area plots, mesh plots, surface plots, contour plots, histogram plots, polar axes, etc. The implementation is based on the graphics language Ti*k*Z and its underlying PGF system by Till Tantau [95]. Based on this solid groundwork, Christian Feuersänger developed the `pgfplots` package [96] to generate plots with simple, user-friendly commands. The very well written documentation is also included in the distribution of this template.

To illustrate the power of PGFPlots, the same figure C.1 is visually improved by adjusting the plot definition that is exported to the TEX file via the Matlab function `matlab2tikz.m`. First, in order to get a clearer TEX file than with the default settings of `matlab2tikz.m`, the following additional options are given:

- The option `externalData` set to `true` determines that the data is separated from the plot definition. As a result, one main TEX file and several other TSV data files are generated.

[1] To get `plot2svg` running simply comment out the following lines, 2421, 2428–2430, 2446, 2453–2455, 2471, and 2478–2480.

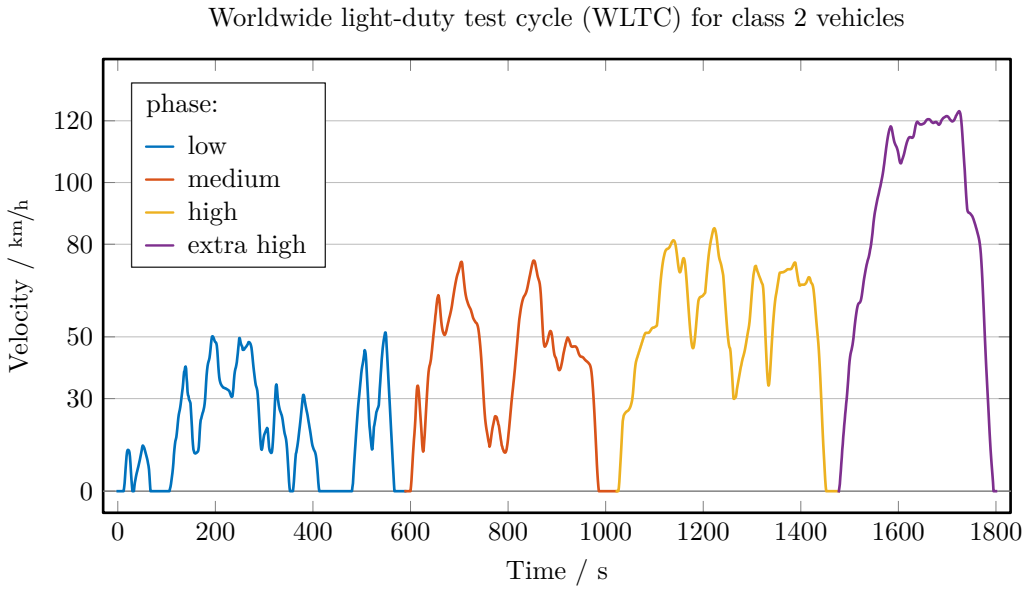


Figure C.5: This plot produced with PGFPlots is probably the best representation compared to the other export techniques shown in figures C.2 and C.3. The data and basic figure definition are exported from Matlab via the function `pgfplots.m`. The plot appearance is then manually improved with various commands of the LaTeX package `pgfplots`.

- The option `relativeDataPath` ensures that the data files are included correctly if they are stored in a subdirectory relative to the main path of the LaTeX document.

The Matlab code that was used to export the graph shown in figure C.1 is as follows:

```
open('WLTC.fig');
matlab2tikz('WLTC_pgfplots.tex', 'noSize', true, 'externalData', true, ...
   'relativeDataPath', 'img/wltc/');
```

When calling the function `matlab2tikz` Matlab will store a `.fig` and a `.tex`, as well as several data files in the the relative subdirectory `img/wltc`:

```
wltc
    WLTC.fig
    WLTC_matlab2tikz.tex
    WLTC_matlab2tikz-1.tsv
    WLTC_matlab2tikz-2.tsv
    WLTC_matlab2tikz-3.tsv
    WLTC_matlab2tikz-4.tsv
```

The representation is then manually adjusted with the help of the extensive manual of PGFPlots and the various examples given by [97]. The complete code that produces figure C.5 is contained in the file `WLTC_matlab2tikz.tex`. Its content is given in the following:

```
% This file was created by matlab2tikz.
%
%The latest updates can be retrieved from
% http://www.mathworks.com/matlabcentral/fileexchange/22022-matlab2tikz- ↵
matlab2tikz
%where you can also make suggestions and rate matlab2tikz.
%
```

```
\definecolor{mycolor1}{rgb}{0.00000,0.44700,0.74100}%
\definecolor{mycolor2}{rgb}{0.85000,0.32500,0.09800}%
\definecolor{mycolor3}{rgb}{0.92900,0.69400,0.12500}%
\definecolor{mycolor4}{rgb}{0.49400,0.18400,0.55600}%
%
\begin{tikzpicture}

\begin{axis}[%
width=0.8\textwidth,
height=0.4\textwidth,
at={(1.011in,0.642in)},
scale only axis,
/pgf/number format/1000 sep={},
xmin=-30,
xmax=1830,
xlabel={Time / s},
ymin=-7,
ymax=140,
ytick={0,30,50,80,100,120},
ymajorgrids=true,
ylabel={Velocity / \unitfrac{km}{h}},
title={Worldwide light-duty test cycle (WLTC) for class 2 vehicles},
line width=1.0pt,
line cap=rect,
line join=round,
legend style={at={(0.03,0.95)},anchor=north west,legend cell align=left,align ↵
=left,draw=white!15!black,legend image post style={xscale=0.6},inner xsep=0.5 ↵
em,column sep=3pt,line width=0.5pt,cells={line width=0.9pt},line cap=round}
]
\addlegendimage{empty legend}
\addlegendentry{\hspace{-1.9em} phase:\\[-2ex]}

\draw[color=gray,solid,line width=0.5pt] (current axis.left of origin) -- ( ↵
current axis.right of origin);

\addplot [color=mycolor1,solid] table[]{img/wltc/WLTC_pgfplots-1.tsv};
\addlegendentry{ low};

\addplot [color=mycolor2,solid] table[]{img/wltc/WLTC_pgfplots-2.tsv};
\addlegendentry{ medium};

\addplot [color=mycolor3,solid] table[]{img/wltc/WLTC_pgfplots-3.tsv};
\addlegendentry{ high};

\addplot [color=mycolor4,solid] table[]{img/wltc/WLTC_pgfplots-4.tsv};
\addlegendentry{ extra high};

\end{axis}
\end{tikzpicture}%
```

The plot definition file `WLTC_matlab2tikz.tex` is then included in the main document via the LaTeX command `\input`, as shown in the following code that produces figure C.5:

```
\begin{figure}[h!]
\centering
```

```
\input{img/wltc/WLTC_pgfplots.tex}
\caption{Plot of the worldwide light-duty test cycle (WLTC) for class~2 ↵
vehicles, produced with PGFPlots. The data and basic figure definition are ↵
exported from Matlab via the function \texttt{matlab2tikz.m}. The plot ↵
appearance is then manually improved with various commands of the \LaTeX ↵
package \texttt{pgfplots}.}
\label{fig:WLTC_pgfplots}
\end{figure}
```

In order to speed up the typesetting process, the compilation of Ti*k*Z-related figures can be externalized with the following two commands. They need to be called in the preamble as described in section 3.7. The first command loads the corresponding library and the second command actually tells LaTeX to precompile the Ti*k*Z figures.

```
\usepgfplotslibrary{external}
\tikzexternalize
```

Given the option `-output-directory="build"` of `latekmk` (see section 3.8.3), appropriate behavior is when all precompiled graphic files are stored in the subdirectory *build* too. Unfortunately, the option `prefix="build"` of `\tikzexternalize` is not satisfactory, since LaTeX assumes that there is a subdirectory *build* within the directory *build*. And because Ti*k*Z is not able to create directories on its own, using the mentioned command can cause compilation errors.

A better approach is provided by [98]. Using `-output-directory="build"` and `\tikzexternalize` without options as proposed in the code snippet above, the precompiled graphic files are stored in the main output directory *build*. The following command then redefines the search location and thus solves the problem with the sub-subdirectory. The command can be added to the preamble of the main document.

```
\tikzset{
  external/system call={pdflatex \tikzexternalcheckshellescape --halt-on- ↵
  error --interaction=batchmode --output-directory=./build --jobname "\image" ↵
   "\texsource"}, /pgf/images/include external/.code={\includegraphics{build ↵
  /#1}}
}
```

C.3 Chart-Like Diagrams with Ti*k*Z

The drawing language Ti*k*Z is not only useful for `pgfplots` to create nice function graphs as shown in appendix C.2, but it also allows beautiful chart-like graphs to be created on their own. An example of such a graph is given in figure C.6, where the build recipe discussed in section 3.8.2 is illustrated in the form of a flowchart. The required code for this graph is given in the following. For detailed information on this implementation please consider Till Tantau's extensive manual [99], which is also included in the distribution of this template.

```
\begin{figure}[h!]
\centering
\usetikzlibrary{shapes,arrows,positioning,calc}
\tikzstyle{decin} = [diamond, draw, fill=black!22, text width=2cm, text badly ↵
 centered]
\tikzstyle{block} = [rectangle, draw, fill=black!13, text width=5em, text ↵
centered, minimum height=3em]
\tikzstyle{cloud} = [ellipse, draw, dashed, fill=black!5, text width=4em, ↵
text centered, minimum height=3em]
```

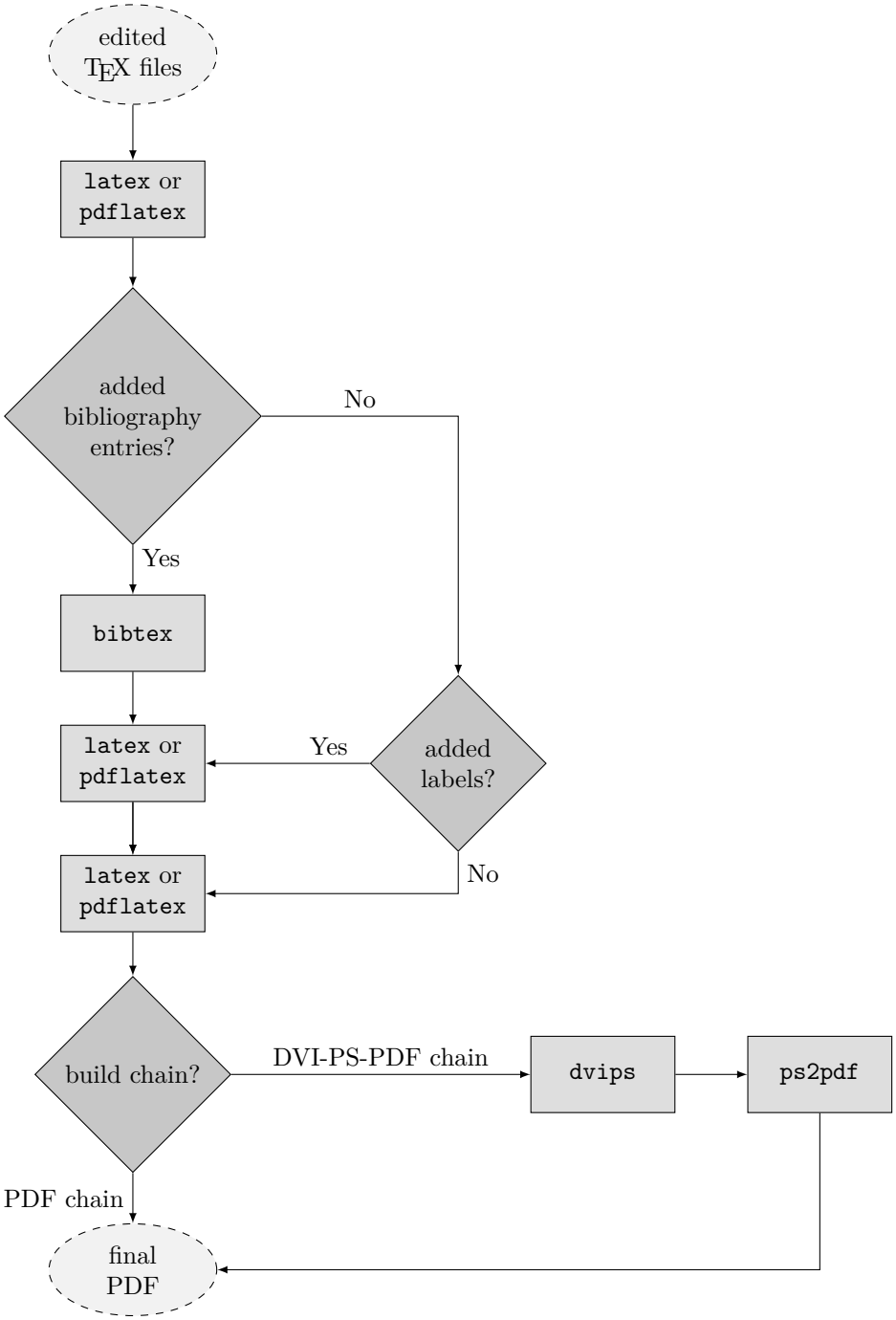


Figure C.6: Flowchart of the build recipe for LaTeX documents produced using the drawing language Ti*k*Z. The corresponding code is listed in appendix C.3.

```
\tikzstyle{arrow} = [draw, color=black, -latex]
\begin{tikzpicture}[scale=2, node distance = 2cm, auto]
% Place nodes
\node[cloud] (edit) {edited \TeX files};
\node[block, below of=edit, node distance=2cm] (tex1) {\texttt{latex} or \ ↵
texttt{pdflatex}};
\node[decin, below of=tex1, node distance=3cm] (dec1) {added bibliography ↵
entries?};
\node[block, below of=dec1, node distance=3cm] (bibtex) {\texttt{bibtex}};
\node[block, below of=bibtex, node distance=1.8cm] (tex2) {\texttt{latex} or ↵
\texttt{pdflatex}};
\node[block, below of=tex2, node distance=1.8cm] (tex3) {\texttt{latex} or \ ↵
texttt{pdflatex}};
\node[decin, right of=tex2, node distance=4.5cm, text width=1.3cm] (dec2) { ↵
added labels?};
\node[decin, below of=tex3, node distance=2.5cm] (dec3) {build chain?};
\node[block, right of=dec3, node distance=6.5cm] (dvips) {\texttt{dvips}};
\node[block, right of=dvips, node distance=3cm] (ps2pdf) {\texttt{ps2pdf}};
\node[cloud, below of=dec3, node distance=2.7cm] (pdf) {final PDF};
% Draw arrows
\path[arrow] (edit) -- (tex1);
\path[arrow] (tex1) -- (dec1);
\path[arrow] (dec1) -- node[near start, right] {Yes} (bibtex);
\path[arrow] (dec1) -| node[near start, above] {No} (dec2);
\path[arrow] (bibtex) -- (tex2);
\path[arrow] (tex2) -- (tex3);
\path[arrow] (tex2) -- (tex3);
\path[arrow] (tex3) -- (dec3);
\path[arrow] (dec2) -- node[near start, above] {Yes} (tex2);
\path[arrow] (dec2) |- node[near start, right] {No} (tex3);
\path[arrow] (dec3) -- node[left] {PDF chain} (pdf);
\path[arrow] (dec3) -- node[above] {DVI-PS-PDF chain} (dvips);
\path[arrow] (dvips) -- (ps2pdf);
\path[arrow] (ps2pdf) |- (pdf);
\end{tikzpicture}
\caption{Flowchart of the build recipe for \LaTeX documents produced using ↵
the drawing language Ti\textit{k}Z. The corresponding code is listed in \cref ↵
{sec:tikz}.}
\label{fig:tikzFlowchart}
\end{figure}
```

Another example showing an electric circuit is shown in figure C.7. To produce this kind of schematics, Massimo A. Redaelli has implemented the package `circuitikz` [100], which is loaded only by this template as the code of the main/root file in section 3.7 shows. The code for figure C.7 is listed below. Additional examples of other chart-like diagrams can be accessed via [97] and specifically on `http://www.texample.net/tikz/examples/tag/diagrams/`.

```
\begin{figure}[h!]
\centering
\begin{circuitikz}
\draw (0,0) to [american voltage source, l_=$U_\mathrm{oc}$] (0,3);
\draw (0,3) to [resistor, l^=$R_\mathrm{batt}$, -o] (4,3);
\draw (0,0) to [short,i<=$I_\mathrm{batt}$,-o] (4,0);
\draw (4,3) to[open, v^>=\mbox{ }$U_\mathrm{batt}$] (4,0);
\end{circuitikz}
```

```
\caption{Equivalent circuit of a battery produced using the \LaTeX package \ ↵
texttt{circuitikz} that is using the drawing language Ti\textit{k}Z. The  ↵
corresponding code is listed in \cref{sec:tikz}.}
\label{fig:tikzCircuit}
\end{figure}
```

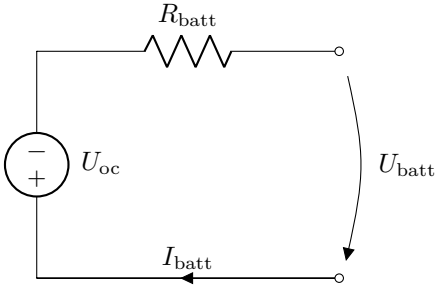


Figure C.7: Equivalent circuit of a battery produced using the LaTeX package `circuitikz` that is using the drawing language Ti*k*Z. The corresponding code is listed in appendix C.3.

C.4 Vector Graphics with Inkscape

Inkscape [101] is a free and open-source software application that is available for Windows, MacOS, and Linux. It is a vector graphics editor similar to Adobe Illustrator or Corel Draw. Although it might be a bit time-consuming to learn how to work with this amazing software in the beginning, good-quality vector graphics are already achieved with little effort. In addition, the website of Inkscape provides an online manual, written tutorials, and even video tutorials. Furthermore, the online community is large and active, such that almost every problem can be solved with the help of designated forums. If the help accessible online is still not sufficient, it is always possible to address your supervisor and probably start your own design with an already existing graphic.

C.4.1 Recommended Formats for LaTeX

Experience with various color printers has shown that it is better to export the graphics of Inkscape to EPS instead of PDF. Although there is no difference noticeable in the resulting digital document, the Inkscape graphics on hard copies appear blurred when they have been exported to PDF instead of EPS. The reason for this inconsistency might be due to Inkscape's internal EPS-to-PDF conversion. We therefore recommend to export Inkscape graphics to EPS and use the LaTeX package `epstopdf` to automatically convert the graphics if `pdflatex` is used.

In order to export an SVG graphic to EPS, select File and Save a Copy in Inkscape. The advantage of saving a copy over saving the file as SVG is given by the fact that Inkscape can also edit PDF and EPS files. Therefore, if the current SVG file is saved as EPS, the current file will be this EPS file and Inkscape continues the editing process on this EPS file. The option Save a Copy causes Inkscape to keep on editing the SVG file.

Out of the three export options to EPS, we recommend the following two, *Embed fonts* and *Omit text in PDF and create LaTeX file*, which are described in the two following subsections. Note that the latter option description states "PDF" even if the output file will be EPS. This is just a minor inconsistency of Inkscape.

Define Labels in Your LaTeX Document

LaTeX offers the possibility to manipulate input EPS files on the fly before inserting them into the document. This allows for example to adjust labels of graphics and replace so-called flags with

proper LaTeX-formatted text such as mathematical expressions. The core package that enables this procedure is `psfrag`, which however only works with the DVI-PS-PDF chain with `latex`, `dvips`, and `ps2pdf`. For the PDF chain the package `pstool` can be used. More information on build chains is given in section 3.8.1. As we recommend using the PDF chain, the remainder of this section focuses on the corresponding approach.

In order to export a regular EPS file from Inkscape, the export options as shown in figure C.8 should be used. As a result, the text elements are included in the EPS file as readable text, which is where `\psfrag` can do its magic during the typesetting process. However, the text is only readable if the SVG source file does not contain any graphical objects that have opacity values different than 100 % or alpha channel values different than 255. For verification purposes, you can always open the EPS document with a text editor and search for the text contents. If you cannot find these text elements, `\psfrag` will not be able to replace them.

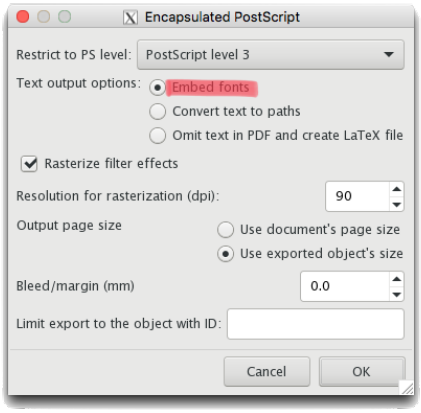


Figure C.8: Export setup for regular export of the SVG graphic to EPS or PDF. Choose this export process when the text elements of your graphic are already formatted, or if the text is replaced via `psfrag` commands in your LaTeX document. Use the LaTeX command `\includelabeledeps` provided by the IDSCreport document class.

The IDSCreport document class offers the command `\includelabeledeps` to conveniently include EPS graphics that contain labels. The following code creates figure C.9.

```
\begin{figure}[h!]
\centering
\includegraphics[width=0.45\linewidth]{img/hysteresis/hystCtrl_raw.eps}
\includelabeledeps[width=0.45\linewidth]{img/hysteresis/hystCtrl_raw.eps}{
\psfrag{0}[c][c]{0}
\psfrag{1}[c][c]{1}
\psfrag{T}[c][c]{$T$}
\psfrag{zR}[c][c]{$z_\mathrm{R}$}
\psfrag{TRmin}[c][c]{$T_{\mathrm{R,min}}$}
\psfrag{TRmax}[c][c]{$T_{\mathrm{R,max}}$}
}
\caption{Example of a figure included with the command \texcmd{ ↵
includelabeledeps}. On the left, the original EPS graphic is shown with its  ↵
labels. On the right, the result is shown when the labels are replaced in \ ↵
LaTeX using the command \texcmd{psfrag}. The corresponding export option of  ↵
Inkscape is shown in \cref{fig:inkscape-export-eps}. However, note that the  ↵
```

```
command \texcmd{includelabeledeps} could be used with any labeled EPS graphic ↵
.}
\label{fig:hystCtrl_epstex_raw}
\end{figure}
```

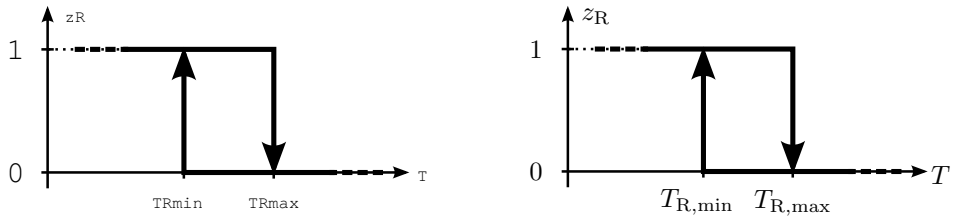


Figure C.9: Example of a figure included with the command `\includelabeledeps`. On the left, the original EPS graphic is shown with its labels. On the right, the result is shown when the labels are replaced in LaTeX using the command `\psfrag`. The corresponding export option of Inkscape is shown in figure C.8. However, note that the command `\includelabeledeps` could be used with any labeled EPS graphic.

If the output directory of the main TeX program is different than the current directory (compare with section 3.8.3), the following command is necessary to specify this directory properly. See also the code of this document in section 3.7 for a reference.

```
\externalizepsfrag{build}
```

Define Labels in Inkscape

The second approach separates the text elements from the graphical elements, as figure C.10 indicates. Inkscape produces two files, a EPS file and a plain-text file with the extension `eps_tex`. The latter file contains LaTeX code to typeset the text elements and to include the graphics contained in the EPS file.

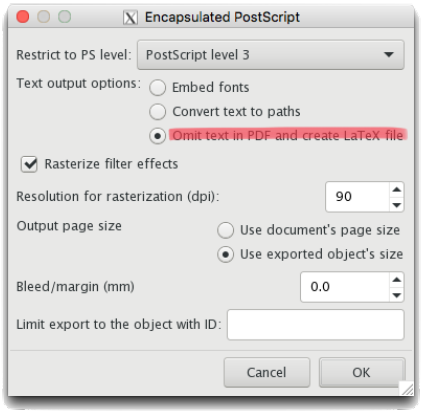


Figure C.10: Export setup for creating an additional TEX file that contains the text labels. For import the LaTeX command `\includeinkscape` may be used.

In order to simplify the process of including Inkscape graphics exported via text separation, the document class IDSCreport offers the command `\includeinkscape`, which emulates the behavior of `\includegraphics`. The code for including figure C.11 is given in the following. Please note that the file extensions PDF or EPS have to be given explicitly.

```
\begin{figure}[h]
\centering
\includegraphics[width=0.45\textwidth]{img/hysteresis/hystCtrl.eps}
\includeinkscape[width=0.45\textwidth]{img/hysteresis/hystCtrl.eps}
\cprotect\caption{Example of a figure included with the command \texcmd{ ↵
includeinkscape}. On the left, the original EPS graphic without text elements ↵
 is shown. On the right, the result is shown when the labels are put on top  ↵
of the graphic via the additional \verb|eps_tex| file.}
\label{fig:hystCtrl_epstex}
\end{figure}
```

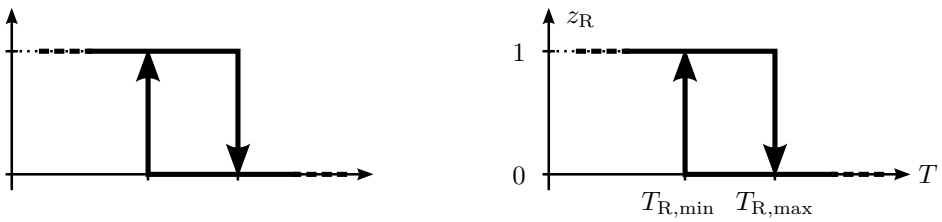


Figure C.11: Example of a figure included with the command `\includeinkscape`. On the left, the original EPS graphic without text elements is shown. On the right, the result is shown when the labels are put on top of the graphic via the additional `eps_tex` file.

One important thing has to be mentioned concerning SVG files that are exported from Inkscape to EPS. For some reason, the text elements are shifted in the resulting figure if the position of any text element is further on the left, right, top, or bottom than any graphical element. Although this seems to be a bug, there is no fix available that is known to the authors. However, a simple workaround can be used to avoid the defect described. Best practice is to create an additional layer at the lowest level that contains a rectangle with exactly the same size as the page size. Set the face color to none and the border line to white and make it very thin. Finally, lock the layer by clicking the padlock symbol, and select the other layer again to continue working on the graph.

C.4.2 Further Inkscape Examples

The illustrative example of the previous section shows a hysteresis controller for a radiator. The control variable z_R is equal to 1 (turn heater on) if the temperature T falls below $T_\mathrm{R,min}$ and 0 (turn heater off) if the temperature exceeds $T_\mathrm{R,max}$.

Addressing the example shown in figure C.1 showing the worldwide light-duty test cycle, Inkscape offers a very convenient way to customize the representation with a graphical editor. A possible result is shown in figure C.12, which is exported via the menu tool bar of the figure in Matlab before it is adjusted in Inkscape. Since showing the source code of this figure is not possible, the corresponding graphical representation in Inkscape is shown in figure C.13.

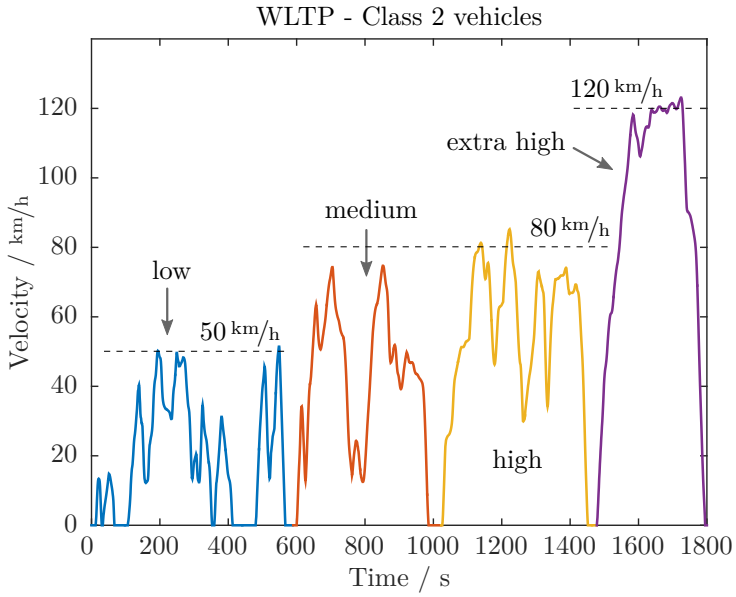


Figure C.12: This figure represents the same Matlab figure as shown in figures C.1 to C.5, but is exported to SVG via the menu tool bar of the figure window in Matlab and adjusted with Inkscape. The corresponding representation in Inkscape is shown in figure C.13.

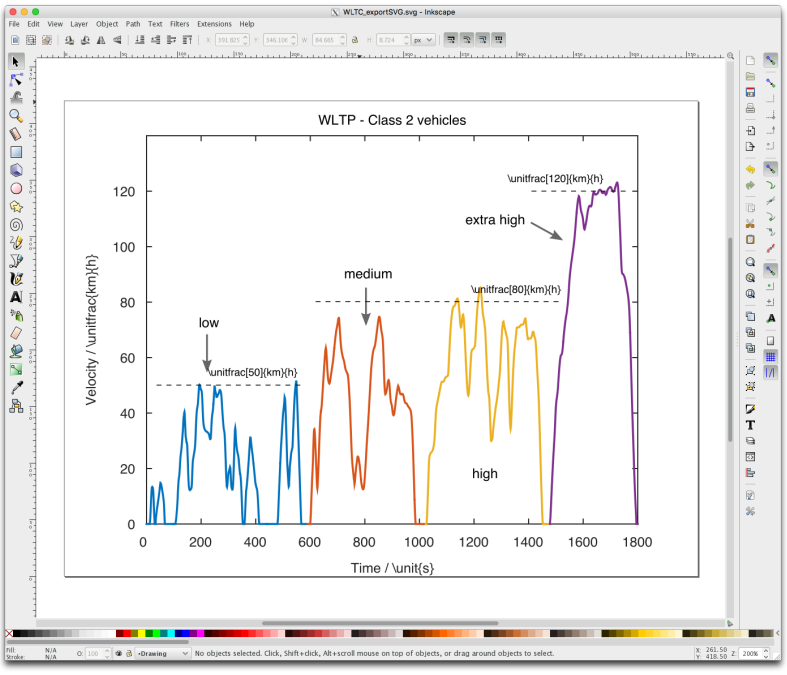


Figure C.13: Screen shot of the vector graphics editor Inkscape while editing the worldwide light-duty test cycle. The original graphic is the exported SVG-version of figure C.1. Note that when Inkscape saves this SVG file to EPS with the option "EPS+LaTeX: Omit text in EPS, and create LaTeX file", it removes all text elements from the EPS and puts them into a separate plain-text file. LaTeX then reads this source file and interprets every text element as regular LaTeX source code. On that account, the physical units of figure C.12 are written with LaTeX commands as explained in section 4.7.

Appendix D

Notes on the Implementation of `IDSCreport.cls`

Everything that a user of this document class IDSCreport has to know is described in the main chapters 3 and 4. The purpose of this appendix chapter is to give an overview on the packages that are included by the document class and the commands that are manipulated. If more information is needed the user is advised to study the implementation of `IDSCreport.cls` *and consult the references mentioned in appendix B.*

D.1 Loaded Packages

The document class IDSCreport loads a bunch of packages that are partly required for the implementation itself and partly recommended regarding the style guides given in chapter 4. The following sections list and describe all packages that are loaded by the document class. If any additional packages are required for a certain report, they should be included by the root file of the document, as explained in section 3.6.

D.1.1 Class Implementation

Special packages required for the implementation of this class:

- `environ` simplifies the definition of new environments such as `abstract`, `preface`, etc.
- `ifpdf` provides the command `\ifpdf` to determine whether the typesetting program is `pdflatex` or `latex`.
- `silence` allows warning messages to be suppressed. This package is used to ignore warnings of LaTeX for packages in subdirectories and warnings of the package `hyperref` when PDF properties are set.
- `draftwatermark` is only loaded if the corresponding class option is set. If loaded, LaTeX automatically prints water marks on each page.

D.1.2 Language

- `babel` with option `ngerman` is only loaded if the report is written in German.
- `fontenc` with option `T1` ensures that only type 1 fonts are used, which are fully vector-based.
- `inputenc` with option `utf8` enables umlauts in German to be written directly (without `\"`).

D.1.3 General

- `fancyhdr` gives full flexibility for creating customized headers and footers.
- `geometry` with options `textwidth=150mm`, `textheight=229mm`, `hcentering`, and `tmargin=4cm`, and depending on the typesetting program with the additional option `pdftex` or `dvipdfm` for `pdflatex` or `latex`, respectively.
- `tocbibind` with option `nottoc` to include the bibliography in the table of contents
- `parskip` manages the two lengths `\parskip` and `\parindent` such that new paragraphs are separated from the previous ones by a small vertical space and the first line of the new paragraph is not indented.
- `placeins` provides the command `\FloatBarrier` that can be used to prevent floats from floating past this barrier.
- `caption` provides many ways to customize the captions in floating environments. In this template it is used to increase the vertical space between the captions and the content of tables to 10 pts.
- `nowidow` with option `all` removes orphans and widows by adjusting the corresponding penalties of LaTeX.

D.1.4 Font

- `lmodern` improves fonts compared to the standard LaTeX fonts.
- `anyfontsize` allows every font size to be set in points.
- `xcolor` provides the command `\color{<color>}{<text>}` for colored text.
- `xspace` provides the command `\xspace` that is used in other macros such that the subsequent spaces are not to be eaten up by the LaTeX typesetting system.

D.1.5 Graphics

- `graphicx` provides a key-value interface for optional arguments to the command `\includegraphics` for including graphics.
- `float` improves the interface for defining floating objects such as figures and tables.
- `pgfplots` allows high–quality function plots to be drawn with a user-friendly interface directly in LaTeX.

D.1.6 Math Symbols

- `amsmath` provides most of the mathematical features of $\mathcal{AMS}$-LaTeX.
- `amssymb` provides an extended collection of symbols and loads the package `amsfonts`.
- `amsthm` helps to define theorem-like structures.
- `mathtools` fixes various deficiencies of `amsmath` and standard LaTeX.
- `mathabx` provides even more mathematical symbols.

D.1.7 Text Symbols

- `textcomp` provides an extensive list of text symbols.
- `xfrac` provides the command `\sfrac` to print nice $^{d}/_{n}$ fractions.

D.1.8 Special Symbols

- mhchem with option version=3 allows the typesetting of chemical molecular formulas and chemical equations.

D.1.9 Physical Units

- units allows the typesetting of physical units and nice fractions.
- siunitx is used for nice degree Celsius and degree Fahrenheit symbols.

D.1.10 Tables

- multirow enables the construction of table cells that span more than one row.
- array extends options of the tabular environment.
- booktabs enhances the quality of tables and provides extra commands.

D.1.11 Environments

- algorithm with option chapter provides the floating algorithm environment.
- algpseudocode provides the environment for typesetting pseudocode.
- fancyvrb provides a better verbatim environment which can be indented.
- enumitem modifies the environment itemize, enumerate, and description, such that the appearance can be customized with optional arguments.
- ragged2e allows setting ragged text which is easy to configure.
- csquotes provides advanced facilities for in-line and display quotations.
- listings provides advanced facilities for the typesetting of source code.

D.1.12 Internet addresses and interactive links

- url with option hyphens provides the command \url for typesetting internet addresses.
- hyperref enables interactive cross-references within the document. If the typesetting program pdflatex is used, the package is loaded with the options pdftex, pdfpagelabels, pdfencoding=auto, and psdextra. No option is given if the typesetting program latex is used.
- breakurl with option hyphenbreaks is only loaded if the typesetting program latex is used.

D.1.13 Special packages that are not part of the TEX distribution

- sourcecode provides style definitions and additional environments for the package listings. It loads the following packages internally:
  - mathabx includes an extended list of math symbols.
  - listings is the basis for all kinds of source code typesetting.
  - matlab-prettifier extends the listings for pretty matlab code.
- specialgraphics provides the commands \includeinkscape and \includelabeledeps which feature a key-value interface for optional arguments that is similar to the command \includegraphics. It loads the following packages internally:
  - import is required to include Inkscape graphics that are stored in subdirectories.
  - xstring allows to manipulate string macros.

The command `\includelabeledeps` creates the images by calling additional LaTeX processes, which require the packages `preview` and `psfrag`.

- `autonomenclature` with option `intoc` provides commands for adding items to the nomenclature and defines the style of the appearance of the nomenclature. Based on the language of the document, the additional option `german` or `english` for German or English, respectively, is given. It loads the following packages internally:
  - `nomencl` allows to create automated nomenclatures. This is the basis of this package.
  - `tabto` allows tabbing to fixed positions in a paragraph with `\tab` or `\tabto{<length>}`.
- `cleverefequations` loads and modifies the definitions of the package `cleveref` such that references to equations appear in parentheses as they would with the command `\eqref`. It loads the following packages internally:
  - `cleveref` simplifies cross-referencing and is the basis of this package.

D.2 Important Redefinitions of Standard Commands

The document class IDSCreport contains a few redefinitions of commands that are common in LaTeX. In order to not confuse the users of this document class, the commands do not imply a drastic change in behavior. However, the modifications are listed here for the sake of completeness.

`\author` and `title` are both tweaked in such a way that they provide optional arguments[1] to specify the author and title attribute of the PDF document information, respectively. If this additional argument is not given, the document class IDSCreport automatically sets the main command input argument to the PDF attributes.

`\chapter` is modified such that new chapters always start on a new odd page. The use of `\cleardoublepage` or the like is therefore not required when the document class `IDSCreport` is used.

`\LaTeX` and `\TeX` have been modified such that they do not eat up the subsequent space in the source code. As an example, this clause containing the work LaTeX is produced by the following code:

```
... this clause containing the work \LaTeX is produced ...
```

The default implementation of `\LaTeX` would require the same text clipping to be written with the following code:

```
... this clause containing the work \LaTeX{} is produced ...
```

[1] A LaTeX command usually has the following structure: `\command[optional arguments]{mandatory arguments}`.

Bibliography

[1] glossary. Merriam-Webster. Accessed: 2017-07-24. [Online]. Available: https://www.merriam-webster.com/dictionary/glossary

[2] Creative commons. Accessed: 2016-12-23. [Online]. Available: https://creativecommons.org/licenses/by-nd/4.0/

[3] GNU general public license. Accessed: 2016-12-23. [Online]. Available: https://www.gnu.org/licenses/licenses.html#GPL

[4] The LaTeX Public Protected License 1.3. Accessed: 2016-12-23. [Online]. Available: http://www.ctan.org/license/lppl1.3

[5] A. Ritter, P. Elbert, and C. Onder, *How to Use the IDSCreport LaTeX Class*, Version 1.6.0, Institute for Dynamic Systems and Control (IDSC), ETH Zürich, Switzerland, Dec. 2018.

[6] A. Foca, *ETH Zurich English Style Guide*, Corporate Communications, Dec. 2015. [Online]. Available: https://www.ethz.ch/services/de/service/kommunikation/corporate-design/language-style-guide.html

[7] L. Guzzella, *Analysis and Synthesis of Single-Input/Single-Output Control Systems*, 3rd ed., ser. vdf Vorlesungsskripte. Zürich: vdf Hochschulverlag AG, 2011.

[8] Robert Bosch GmbH, "Fachwörterbuch Kraftfahrzeugtechnik," in *Fachwörterbuch Kraftfahrzeugtechnik*. Wiesbaden: Vieweg+Teubner Verlag, 2002, pp. 6–513.

[9] (2015, Jun.) 'that' or 'which'? Oxford University Press. Accessed: 2015-06-30. [Online]. Available: http://www.oxforddictionaries.com/words/that-or-which

[10] (2015, Jun.) List of transition words. LoveToKnow, Corp. Accessed: 2015-06-30. [Online]. Available: http://grammar.yourdictionary.com/style-and-usage/list-transition-words.html

[11] (2015, Jun.) Compound words: When to hyphenate. Get It Write Online. Accessed: 2015-06-30. [Online]. Available: http://www.getitwriteonline.com/archive/042703compwdshyph.htm

[12] Compound (linguistics). Wikipedia. Accessed: 2015-06-30. [Online]. Available: https://en.wikipedia.org/wiki/Compound_(linguistics)

[13] (2015, Jun.) Allow (to) + infinitive, substantive, verb+ -ing. FumbleFingers. Accessed: 2015-07-23. [Online]. Available: http://ell.stackexchange.com/questions/11193/allow-to-infinitive-substantive-verb-ing

[14] Serial comma. Wikipedia. Accessed: 2015-07-23. [Online]. Available: https://en.wikipedia.org/wiki/Serial_comma

[15] (2015, Jun.) Commas. GrammarBook.com. Accessed: 2015-06-30. [Online]. Available: http://www.grammarbook.com/punctuation/commas.asp

[16] Comma - separation of clauses. Wikipedia. Accessed: 2015-07-23. [Online]. Available: https://en.wikipedia.org/wiki/Comma#Separation_of_clauses

[17] (2015, Jun.) Should i always use a comma after “e.g.” or “i.e.”? StackExchange: English Language & Usage. Accessed: 2015-07-02. [Online]. Available: http://english.stackexchange.com/questions/16172/should-i-always-use-a-comma-after-e-g-or-i-e

[18] Adverbs and adverb phrases: position. Cambridge Dictionaries Online. Accessed: 2015-07-23. [Online]. Available: http://dictionary.cambridge.org/grammar/british-grammar/adverbs-and-adverb-phrases-position

[19] (2015, Jun.) Adverb placement. Mary Nell Sorensen. Accessed: 2015-07-23. [Online]. Available: http://staff.washington.edu/marynell/grammar/AdverbPl.html

[20] (2015, Jun.) Rules for capitalization in titles of articles. LoveToKnow, Corp. Accessed: 2015-07-08. [Online]. Available: http://grammar.yourdictionary.com/capitalization/rules-for-capitalization-in-titles.html

[21] TeX Live. Wikipedia. Accessed: 2017-04-10. [Online]. Available: https://de.wikipedia.org/wiki/TeX_Live

[22] MacTeX. Wikipedia. Accessed: 2017-04-10. [Online]. Available: https://en.wikipedia.org/wiki/MacTeX

[23] TeX Live Utility. GitHub. Accessed: 2017-04-10. [Online]. Available: http://amaxwell.github.io/tlutility/

[24] MiKTeX. Wikipedia. Accessed: 2017-04-10. [Online]. Available: https://en.wikipedia.org/wiki/MiKTeX

[25] MikTeX. Accessed: 2017-04-10. [Online]. Available: http://docs.miktex.org/manual/pkgmgt.html

[26] S. Rahtz, T. Esser, G. Wierda, H. Oberdiek, K. Berry, and J. V. Zandt, *EPSTOPDF General Commands Manual*, Version 2.23, Jan. 2014.

[27] M. C. Grant and D. Carlisle, *The PSfrag system, version 3*, Apr. 1998.

[28] (2015, Jun.) How does the LaTeX compile chain work exactly? StackExchange: TeX. Accessed: 2015-07-03. [Online]. Available: http://tex.stackexchange.com/questions/121383/how-does-the-latex-compile-chain-work-exactly

[29] (2015, Jun.) Why does LaTeX/bibTeX need three passes to clear up all warnings? StackExchange: TeX. Accessed: 2015-07-08. [Online]. Available: http://tex.stackexchange.com/questions/53235/why-does-latex-bibtex-need-three-passes-to-clear-up-all-warnings

[30] (2016, May) TeXstudio, LaTeX made comfortable. TeXstudio. [Online]. Available: http://texstudio.sourceforge.net

[31] (2016, May) TeXstudio : user manual. TeXstudio. [Online]. Available: http://texstudio.sourceforge.net/manual/current/usermanual_en.html

[32] Skim. Accessed: 2017-04-10. [Online]. Available: http://skim-app.sourceforge.net

[33] Sumatra PDF. Accessed: 2017-04-10. [Online]. Available: https://www.sumatrapdfreader.org/free-pdf-reader.html

[34] Perl. Wikipedia. Accessed: 2017-04-10. [Online]. Available: https://en.wikipedia.org/wiki/Perl

[35] LaTeX/Installing extra packages. Wikipedia. Accessed: 2017-04-10. [Online]. Available: https://en.wikibooks.org/wiki/LaTeX/Installing_Extra_Packages

[36] latexmk. CTAN. Accessed: 2017-04-10. [Online]. Available: https://www.ctan.org/pkg/latexmk/

[37] Latexmk script could not be found. StackExchange. Accessed: 2017-04-10. [Online]. Available: http://tex.stackexchange.com/questions/336771/latexmk-script-could-not-be-found

[38] The chicago manual of style online. Accessed: 2016-05-30. [Online]. Available: http://www.chicagomanualofstyle.org

[39] Oxford dictionaries. Accessed: 2016-05-30. [Online]. Available: http://www.oxforddictionaries.com

[40] Quotation marks in english. Wikipedia. Accessed: 2016-05-30. [Online]. Available: https://en.wikipedia.org/wiki/Quotation_marks_in_English#Order_of_punctuation

[41] M. Shell, *How to Use the IEEEtran BIBTEX Style*, Version 1.8b, Sep. 2008.

[42] T. Cubitt, *The cleveref package*, corresponds to cleveref 0.19, dated 2013/12/28, Dec. 2013.

[43] J. Bezos, *Customizing lists with the enumitem package*, Version 3.5.2, 2011-09-28, Sep. 2011.

[44] S. Pakin, "The Comprehensive LaTeX Symbol List," http://tug.ctan.org/info/symbols/comprehensive/symbols-a4.pdf, Nov. 2015.

[45] (2016, May) Detexify. kirelabs.org. [Online]. Available: http://detexify.kirelabs.org/classify.html

[46] List of mathematical symbols by subject. Wikipedia. Accessed: 2016-05-13. [Online]. Available: https://en.wikipedia.org/wiki/List_of_mathematical_symbols_by_subject

[47] A. Phan, *Mathabx series, Informations and tests*, Latest version: May 18, 2005, May 2016.

[48] W. Putschögl, *The commath LATEXpackage v0.3*, July 18, 2006, Jun. 2006.

[49] M. Høgholm, L. Madsen, W. Robertson, and J. Wright, *The xfrac package Split-level fractions*, Version v6476, last revised 2016/04/20, Apr. 2016.

[50] M. Hensel, *The mhchem Bundle*, Version 3.07, 2007/05/19, May 2007.

[51] M. HØgholm and L. Madsen, *The mathtools package*, Version v1.18, last revised 2015/11/12, Nov. 2015.

[52] (2016, May) LaTeX/Advanced Mathematics. Wikibooks. Accessed: 2016-05-13. [Online]. Available: https://en.wikibooks.org/wiki/LaTeX/Mathematics

[53] Bureau International des Poids et Mesures (BIPM), *Le Sysème International d'Unités (SI), (The International System of Units (SI))*, 8th ed. Sevres, France: Organisation Intergouvernementale de la Convention du Mètre, 2006. [Online]. Available: http://www.bipm.org/en/publications/si-brochure/

[54] UNECE, "Addendum 15: Global technical regulation No. 15 (Worldwide harmonized Light vehicles Test Procedure)," United Nations Economic Commission for Europe, Tech. Rep., May 2014, last Accessed: 20 May 2016. [Online]. Available: http://www.unece.org/fileadmin/DAM/trans/main/wp29/wp29r-1998agr-rules/ECE-TRANS-180a15e.pdf

[55] S. Fear, *Publication quality tables in LATEX*, Version v1.618033, last revised 2016/04/27, Apr. 2016.

[56] F. Mittelbach and D. Carlisle, *A new implementation of LaTeX's tabular and array environment*, Version v2.4c, last revised 2014/10/28, May 2016.

[57] P. van Oostrum, O. Bache, and J. Leichter, *The multirow, bigstrut and bigdelim packages*, Version 1.6, Feb. 2007.

[58] (2016, May) Latex/tables. Wikibooks. Accessed: 2016-05-20. [Online]. Available: https://en.wikibooks.org/wiki/LaTeX/Tables

[59] K. Müller, A. Hawryluk, and J. Marder. (2016, May) Excel2latex. Accessed: 2016-05-20. [Online]. Available: http://www.ctan.org/tex-archive/support/excel2latex/

[60] (2016, May) Tables generator. Accessed: 2016-05-20. [Online]. Available: http://www.tablesgenerator.com

[61] S. János, *The algorithmicx package*, Version 1.2, 2005/04/27, Apr. 2005.

[62] R. Brito, *The algorithms bundle*, Version v0.1, dated 2009/08/24, Aug. 2004.

[63] Dijkstra's algorithm. Wikipedia. Accessed: 2016-05-16. [Online]. Available: https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm

[64] C. Heinz, B. Moses, J. Hoffmann, and J. Hoffmann, *The Listings Package*, Version 1.6, 2015/06/04, May 2016.

[65] M. Haklay and P. Weber, "OpenStreetMap: User-generated street maps," vol. 7, no. 4, pp. 12–18, 2008.

[66] L. Guzzella and C. Onder, *Introduction to Modeling and Control of Internal Combustion Engine Systems*, 2nd ed. Berlin, Heidelberg: Springer Science & Business Media, 2010.

[67] S. Chacon, "Pro Git," Apress, 2009.

[68] M. Andersen, J. Dahl, Z. Liu, and L. Vandenberghe, "Interior-point methods for large-scale cone programming," in *Optimization For Machine Learning*, S. Sra, S. Nowozin, and S. J. Wright, Eds. Cambridge: MIT Press, 2012.

[69] J. F. Sturm, "Using sedumi 1.02, a Matlab toolbox for optimization over symmetric cones," in *Optimization Methods and Software*. Taylor & Francis, 1999, pp. 625–653.

[70] A. Ritter, "Thermodynamisches Modell eines Busses und Vorschläge für alternative HVAC Systeme," Studies on mechatronics, ETH Zurich, Institute for Dynamic Systems and Control, Zurich, Mar. 2011, IDSC-LG-PE-08.

[71] S. Losa, "Model of a three-way catalytic converter in simulink," Bachelor's thesis, ETH Zurich, Institute for Dynamic Systems and Control, Zurich, Jun. 2015, IDSC-CO-AR-02.

[72] A. Ritter, "Optimal energy management for hybrid electric buses via iterative convex optimization," Semester project, ETH Zurich, Institute for Dynamic Systems and Control, Zurich, Dec. 2012, IDSC-LG-PE-11.

[73] D. Flubacher, "lighTram Hybrid: Datenbearbeitung und geschwindigkeitsinduzierte Optimierung," Master's thesis, ETH Zurich, Institute for Dynamic Systems and Control, Zurich, Bellach, Dec. 2010, IDSC-LG-PE-01.

[74] P. Elbert, "Noncausal and causal optimization strategies for hybrid electric vehicles," Ph.D. dissertation, ETH Zurich, Institute for Dynamic Systems and Control, Zurich, Oct. 2014, Diss. ETH No. 21522.

[75] *MATLAB Object-Oriented Programming*, R2013, MathWorks, Inc., 2013.

[76] Bundesamt für Statistik (BFS), "Mobilität in der Schweiz," Bundesamt für Statistik, Bundesamt für Raumentwicklung, Tech. Rep., Jan. 2012.

[77] (2015, Oct.) Institute for Dynamic Systems and Control. ETH Zurich. [Online]. Available: http://www.idsc.ethz.ch

[78] Merriam-Webster, *Merriam-Webster's Collegiate Dictionary*, 11th ed. Springfield: Merriam-Webster, Incorporated, 2003.

[79] American and british english spelling differences. Wikipedia. Accessed: 2015-07-09. [Online]. Available: https://en.wikipedia.org/wiki/American_and_British_English_spelling_differences

[80] (2015, Jun.) Differences between british english and american english spelling. Spellzone Limited 2013. Accessed: 2015-07-09. [Online]. Available: http://www.spellzone.com/pages/british-american.cfm

[81] (2015, Jun.) British and american terms. Oxford University Press. Accessed: 2015-06-29. [Online]. Available: http://www.oxforddictionaries.com/words/british-and-american-terms

[82] T. Oetiker, H. Partl, I. Hyna, and E. Schlegl, *The Not So Short Introduction to LaTeX 2ε*, Version 5.05, Jul. 2015.

[83] M. Daniel, P. Gundlach, W. Schmidt, J. Knappen, H. Partl, and I. Hyna, *LaTeX 2ε-Kurzbeschreibung*, Version 3.0a, Jun. 2015.

[84] G. Grätzer, *More Math Into LaTeX*, 5th ed. Toronto, ON, Canada: Springer International Publishing, 2016.

[85] (2004, Sep.) LaPrint (LaTeX Print). Control and Automation Department of Electrical and Computer Engineering, University of Kassel. Accessed: 2016-05-25. [Online]. Available: http://www.uni-kassel.de/fb16/rat/matlab/laprint/

[86] A. Linnemann. (2009, Sep.) Laprint. MathWorks. Accessed: 2016-05-25. [Online]. Available: http://de.mathworks.com/matlabcentral/fileexchange/4638-laprint

[87] Z. Prime, *matlabfrag user guide*, May 2010.

[88] ——. (2014, Feb.) matlabfrag. GitHub. Accessed: 2016-05-25. [Online]. Available: https://github.com/zprime/matlabfrag

[89] ——. (2010, May) matlabfrag. MathWorks. Accessed: 2016-05-25. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/21286-matlabfrag

[90] Martin. (2010, Sep.) matlabfrag to pdf. MathWorks. Accessed: 2016-06-02. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/28545-matlabfrag-to-pdf

[91] matlab2tikz. GitHub. Accessed: 2016-05-25. [Online]. Available: https://github.com/matlab2tikz/matlab2tikz

[92] N. Schlömer. (2016, May) matlab2tikz. MathWorks. Accessed: 2016-05-25. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/22022-matlab2tikz

[93] J. Schwizer. (2012, Sep.) Scalable vector graphics (SVG) export of figures. MathWorks. Accessed: 2016-06-01. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/7401-scalable-vector-graphics--svg--export-of-figures

[94] ——. (2015, May) Save matlab plots as svg files. GitHub. Accessed: 2016-06-01. [Online]. Available: https://github.com/jschwizer99/plot2svg

[95] T. Tantau and C. Feuersänger. Pgf and tikz – graphic systems for tex. Sourceforge. Accessed: 2016-06-02. [Online]. Available: https://sourceforge.net/projects/pgf/

[96] D. C. Feuersänger, *Manual for Package PGFPlots*, Oct. 2012. [Online]. Available: https://sourceforge.net/projects/pgfplots/

[97] S. Kottwitz. TeXample.net. Accessed: 2016-05-26. [Online]. Available: http://www.texample.net/tikz/examples/

[98] How to prevent tikzexternalize to generate files in current directory? Accessed: 2016-12-15. [Online]. Available: http://tex.stackexchange.com/questions/333960/how-to-prevent-tikzexternalize-to-generate-files-in-current-directory

[99] T. Tantau, *The TikZ and PGF Packages, Manual for Version 3.0.1a*, Aug. 2015. [Online]. Available: http://www.pirbot.com/mirrors/ctan/graphics/pgf/base/doc/pgfmanual.pdf

[100] M. A. Redaelli, *CircuiTikZ*, Version 0.3.0, Dec. 2012.

[101] Inkscape. [Online]. Available: www.inkscape.org