A Guide to Proceedings for RiGs and Beyond

Stefan Waldmann

Institute for Mathematics
Chair of Mathematics X (Mathematical Physics)
Emil-Fischer-Straße 31
97074 Würzburg
Germany

Current Version: 2024-04-09 11:45:19 +0200

Last changes by Stefan@JMU on 2024-04-09
Git revision of proceeding: `9439f53` (HEAD -> master)

Abstract

For a Reseach in Groups (RiG) one usually has to hand in a proceeding which contains details about the talk, additional information, examples, and references which were not mentioned in the talk. Here we elaborate on some general ideas how such a proceeding should look like. Many of our remarks also apply to other longer texts in mathematics like bachelor or master theses as well as to research papers.

Contents

1 Introduction

In a Reseach in Groups (RiGs) at the Institute of Mathematics in Würzburg there are usually several different components: a lecture part by the professor, a seminar part where the participants give talks, and a part where the participants have to hand in written proceedings about the talks. To write such a proceeding is by far not a trivial task. Instead, it can be seen as a core ingredient to scientific research and, as such, it requires particular attention. While most participants of a RiG made already first experiences with writing a scientific text in mathematics (or mathematical physics) in their bachelor thesis, it will nevertheless be important to advance and improve the writing skills. Ultimately, the contributions of all participants will be collected in a proceeding volume and made available to all.

Needless to say, scientific writing will be of crucial importance not only for the final days of the production of a master thesis but for the entire life as a professional mathematician: in research, the main source of communication is still the written article, the scientific publication in journals. With the proceedings for a RiG one obtains a cheap and easy training for writing more serious and important publications. But also outside of university and outside of scientific research, reports have to be written. The details in the presentation may be different but the essentials are still the same.

We will collect now some ideas on how a good proceeding (and similarly, a good thesis, a good article) is structured and written. To some extend they reflect a personal style and should be understood as guidelines instead of strict rules.

There are several aspects to be taken into account: first, in Section 2, we discuss how one has to determine the content of a proceeding and structure it in an appropriate way. This will be essentially entirely independent of the way how the actual proceeding is written. Second, one has to decide on the form of the presentation. Here the specific needs of the community one is writing for enter. Some ideas on this will be presented in Section 3. Third, one has to implement the text by actual typesetting. In earlier days publishers of scientific journals took care of that step almost completely. However, with the advent of more advanced computer technology, this task is left to the authors more and more. Clearly, for a RiG proceeding as well as for a thesis, this will have to be done by the author. Here a clever choice of the appropriate tools enters the game. We will discuss two aspects. First it is clear that in mathematical writing there is no way to avoid the usage of LaTeX: for good reasons this is the very standard of mathematical writing. We will therefore assume that a basic knowledge of LaTeX is already present and indicate how certain pitfalls in its usage can be avoided in Section 4. This section also contains some more specific information on how RiG proceedings are handled at Chair X. Second, there are many tools beyond LaTeX which help to produce a reasonable text. Some of these basic tools are discussed in Section 5. Of course, for a simple RiG proceeding it might not be necessary to use the full machinery of version control, Makefiles, and bibliography handling. However, it is a good start to learn these techniques already for a proceeding in order to have them present when they will actually be needed. Latest for the master thesis it is very advisable to use more elaborate techniques to handle large multiple-file LaTeX-projects.

2 The Content of a RiG Proceeding

While in the talk for the RiG one has a certain time constraint, the proceedings allow for a slightly larger flexibility. Of course, one should not start writing a book and keep a reasonable length. But still, one has the opportunity to write things in more details and explain the content in a way which is not possible in the talk.

It should be clear that a proceeding is not just a collection of keywords but consists of *complete sentences*. This is different from a handout which should be placed on a single page. In a proceeding, one expects explanations and a hence a reasonable amount of text. The language might be either English or German, details will be communicated by the professor. Nevertheless, it is always a good idea to take this opportunity and write in English, in particular as a non-native speaker. Later in your career you will mainly (only) write English texts.

Depending on the professor there might or might not be a *page limit* for the proceedings. In general, it should be detailed enough to be readable without referring too much to external references but not too long to be boring. In some sense, it is always a good guideline to have as much details as the average participant of the RiG would need in order to understand things properly. To have some numbers: everything below 5 pages might be too short and everything above 20 pages will not be read by anyone. But this of course depends on what kind of text one has. Many technical formulas or commutative diagrams which are necessary for the understanding might require some longer explanations. Pictures take a lot of space but are usually very welcome. Long repetitions of well-known definitions are probably not appropriate and can be skipped. Here one has to find a good balance.

Every proceeding starts with a *title page* containing the title of the talk, the name of the RiG, the author, which should coincide with the speaker in the case of a RiG, the professor who offered the RiG, and the date. For the date it is usually enough to give the month and the year or just the term but not the precise day. However, it might be useful for the professor and the other participants to give the date of the talk in full detail. Sometimes it might be possible to have a proceeding written by several participants each of which delivered a separate but related talk. In this case it might be easier to write some larger text together in order to have a coherent presentation without too much overlap and repetition. It will be a first step in actual collaborative work in mathematics and thus an opportunity not to be missed. Of course, such a case should be discussed with the professor first.

Every proceeding should have a short *abstract*. The abstract should contain a rough overview on what the proceeding is all about. Some 10 lines will typically be enough. The abstract is located directly on the title page.

It might be required and it is always a good idea to have a *table of contents* before the actual text starts. This can be produced automatically, so one only has to place the corresponding LaTeX-command at the right position.

While the above applies to most scientific texts, in a RiG the professor might offer a certain template which takes care of the title page, the abstract, and the table of contents. In particular, it might be that the proceeding is put into a bigger file and, together with many other proceedings, included into a real little booklet. In this case, the table of contents will appear at the beginning of the booklet, not at the beginning of the particular contribution. Also the title page might become the first page of a chapter. Nevertheless, the same information will be required, only the presentation inside the booklet will be adapted. If such a template is offered, you are strongly advised to actually use it. In particular, avoid anything which is in conflict with the structure and settings in the template.

Every proceeding needs an *introduction*. This is not to be confused with the abstract. The introduction is the first section of the proceeding and it is needed for several reasons: first, the introduction gives an overview on the topic as a whole, providing a guide to the standard literature which is used in the field and to the literature which is actually used for the specific proceeding. Here sometimes a short historical overview is very welcome. Who are the main actors in the field, what principal achievements have been done in the last years, what are the main open questions. At the end of this first larger part of the introduction, one should relate the general considerations to the actual task of the proceeding. How does the general theory relate to the question under consideration, why is this question still important, what is the impact of its solution. The next part should be rather precise and also concise. One has to introduce the actual topic of this work. Ideally, the first half of the introduction sets the stage and leads to this part in a very convincing way so that the reader is somehow led to this one particular question: formulate it in one single line as the core of the introduction. After proposing the question and motivating its solution the remaining part of the introduction is used to explain how the problem is formulated and solved. Here one usually can put a sort of commented table of contents, explaining which parts of the proceeding contain what important results. In some sense the introduction is the most important part of the proceeding or any other scientific text. Here all important references are mentioned, the field of interest is presented, the main question and its solution is discussed, and an overview on the structure of the remaining text is outlined. Generally, the introduction should be written as the last part when the remaining pieces of the proceeding are stable enough to be described.

Either at the end of the introduction or as a separate small section at the very end of the text it might be appropriate to place some *acknowledgements*: if the author wants to express gratitude to someone else who helped in one or the other way in the production of the text, this is the right moment to do so. For a thesis but also for journal articles it might be necessary to acknowledge the funding if there was some external funding, a grant etc. involved in the production of the work.

After the introduction the text is structured into *sections* and, maybe, subsections. In general one should structure the text in a sensible way and spend quite some time on organizing the material. Just a wild collection of lemmas and definitions is not at all a good mathematical text. The sections are numbered with Arabic numbers, the subsequent subsections carry the section number in front of their own number. LaTeX usually takes care of this numbering so there is no need to worry about it. Sometimes it might be good to start the numbering with the first section after the introduction, the introduction itself may get no number yet as it stands above (outside, beyond ...) the main text.

Mathematical formulas are already very condensed and contain a very intense amount of information. A good mathematical text therefor gives also *explanations* of the actual mathematical context. Nobody wants to read just a collection of logical implications and mathematical symbols even though this would be completely sufficient to state and prove a mathematical result. Mathematical texts are not computer programs but written for human beings after all. Your explanations should give ideas why a certain proof should work, what the strategy of proving things is, what the motivation behind a definition could be. In particular, it is good practice to start a new section with a short introduction what is going to happen instead of starting it with just a formula or a definition etc. These explanatory parts of a mathematical text are of course considered to be "soft" and subjective. Nevertheless, they might help the reader substantially to understand the "hard" mathematical context.

Inside the sections one should make use of the usual *mathematical structures* of definitions, lemmas, propositions, and theorems. Sometimes, it is not completely obvious what counts as a lemma, what a proposition should be, and when a result deserves the label theorem.

Here everybody can develop an individual style. Nevertheless, it is quite common to use the term *lemma* for some mathematical statement which is not yet interesting per se but needed to prove something bigger. Hence lemmas can be used to split up a large and complicated proof into several single steps. A *proposition* is then a result which is interesting in its own but may not yet deserve to be named theorem. A *theorem* usually should be the core of the proceeding, maybe there is only one theorem in the whole text. This is the main result and should be announced in the introduction. One possibility is even to place the one and only theorem in the introduction and use the rest of the proceeding as an organized way to prove it. Having shown the main theorem of the text it is time for one or the other *corollary*, simple consequences of the main statement which contain nevertheless interesting particular cases or special applications. A good mathematical text contains *examples*: in fact, most of the time in mathematics it is important to understand a sufficiently non-trivial example well enough before one can even formulate the general theorem. Ideally, a well-chosen example motivates the whole text and gives already the hint of how the main statement can be obtained. One should never underestimate the force and power of examples. A common mistake is that inside *definitions* one already hides conclusions, statements about the new structure which was defined. This is very bad mathematical style. Statements, as simple and obvious as they might be, should never be part of a definition. Introducing new notation or recalling well-known definitions might or might not be done in a separate definition. Alternatively, this can be done in the main text. In this case it is good style to emphasize the newly defined word in the main text appropriately by choosing e.g. a different font. This has of course to be done in a systematic way. Sometimes other mathematical structures like a *remark*, a *claim*, a *conjecture*, or a *question* might be needed.

One main part of the work in mathematics is done in the *proofs* (the other main work is finding the good definitions). In a proceeding one has now the time and space to give the proofs in self-contained details, thereby clarifying those aspects for which there was no time in the talk. It should always be clear when a proof starts and when it ends. Typically, one uses particular symbols like $\square$ or a QED to indicate the end of the proof. Using the appropriate LaTeX-environment this is done automatically at the end of the proof.

In other scientific communities it is common to place a *summary* at the end of the text. In mathematical writing however, this is typically neither required nor considered to be good style: why should one repeat the things already said in the main text again? Mathematicians tend to be intelligent enough to grasp the core of the text without reading a summary at the end. In fact, in mathematics it is the introduction which contains the important meta-information about the main text, the guideline how to read it, and the overview on the relations to previous works. For a RiG proceeding, one should not put a summary at all. Sometimes it might be an option to place a last section containing an outlook, a collection of open problems related to the main text, or a list of possible future developments and projects.

It might be a good idea to collect some well-known definitions and statements needed in the main text in an *appendix*. The appendix appears after all the main sections of the proceeding but before the bibliography. It can contain sections and subsections as well, but for a RiG proceeding it should not be too long. In fact, it is always suspicious if the appendix exceeds the size of the main text.

At the very end of the proceeding one has to place the *bibliography*. This should be as detailed and complete as possible: all works used in the text and also in the talk should appear here. If one used several books for the same topic, one can either place them all in the

Title page
Abstract
Table of Contents (optional)
Introduction (with optional acknowledgements)
Various sections, possibly with subsections
Appendix, several sections (optional)
Bibliography

Table 1: The structure of a RiG proceeding

bibliography or those which are most easily available. Of course, we all use Wikipedia as a quick reference in our daily life. However, this does not qualify as a scientific reference. Always find some textbook or journal article where the things can be found instead. As a general rule one should make references to all statements, definitions, examples, and conclusions which are not genuinely original to oneself. In a proceeding this is typically everything!

We have schematically summarized the rough structure of a RiG proceeding in Table 1. Note that this is to be understood as a guideline and not as a mandatory form. However, if deviating from it, one should have good reasons to do so.

3 Typesetting Mathematics

Typesetting is of course an *art* in itself and deserves certainly much more explanation as it can be done here. Even if one is mainly interested in writing the rather formalized mathematical texts needed for a RiG proceeding or a thesis, it is good to be familiar at least with some of the basic principles of good typesetting. There are several classics one can take a look at [1, 4, 5, 7]. Going through all this is certainly a rather demanding task but ultimately much more rewarding than a naive typing of mathematical texts can every be. Since a proceeding and similarly every other possible text in mathematics has, by design, the purpose of being read by others, one has to take into account their needs and demands in order to make the reading easier and hopefully even pleasant. As mathematical texts tend to be difficult to read anyway, at least the optical appearance should be helping and supportive instead of confusing and distracting.

On top of general principles of good writing, mathematical texts follow in large parts their own specific rules: it will make a serious difference if a symbol a is set in italic or like $\boldsymbol{a}$ in boldface. Of course, very good mathematical texts are robust enough and use notation which is not too sensitive to such typesetting issues. However, most of the time this is unavoidable. It will also be of great importance that certain mathematical structures will be set in a uniform way. This increases the readability in a significant way.

Mathematical *symbols* have to be explained. There are so many different communities in mathematics and everyone uses its own terminology and notation that it is absolutely crucial for a good mathematical text to explain the meaning of all the symbols occurring in the text. This becomes even more important if one is using non-standard notations and own ideas. A general guideline is that too baroquely decorated symbols (with tildes and hats on top, two subscripts left and right, typeset in a rather obscure and fancy font, and with slashes, circles, and dots around) will not increase the readability of the text. Sometimes it is important to emphasize a functorial dependence of one symbol on others but this should not be exaggerated.

If possible, stick to standard notation used in the lecture and in textbooks. Sometimes it is also helpful to explicitly tell that you follow the notations and conventions form a certain, widely used textbook which, however, are different from another widely used textbook. Maybe not yet in a proceeding but in longer texts like a thesis it is advantageous and supportive for the reader to provide a *list of symbols* containing the most important symbols specific for the text with short explanations. This can be put as a separate section either between the introduction and the following main text or just before the bibliography.

Citations to the *references* should be made as precise as possible. A reference to a textbook with 600 pages is completely useless if one is not referring to a particular section, a particular theorem, or a particular example. Even when referring to a short article it is good style to make the reference to a specific part of the article as precise as possible. Some references can be made globally, say for a whole section, if the reference applies to everything done in this section. In this case one gives the reference at the beginning of the section and writes that the content of this particular section can be found in the reference. This is a acceptable strategy for short sections containing a single topic. However, if the section becomes longer, it might be a better strategy to make references more specifically: it is this definition which was taken from there and that theorem which was taken from somewhere else. If in doubt whether to make a reference or not, make a reference! Giving proper references is not just a nice service to the possible reader but simply the basic core of good scientific practice: achievements of other persons have to be acknowledged in a sensible and fair way. In fact, the last years showed that political careers tend to end very early if these rules of good scientific practise are mistreated.

Longer mathematical computations are not placed in the main text but in separate *equations* or even whole arrays of equations. Now the importance of equations may be very different: either they contain the main statement, the $E = mc^2$ of the whole proceeding. In this case they deserve a separate equation number which typically should include the section number as well. Or the equation is just one of many lines of computation hidden in some proof and simply too long to be stated in the main text. In this case, an equation number looks superfluous. As a general rule it might be good to place numbers for all those equations which are not inside a proof while those inside a proof do not get numbers. Sometimes it is then of course necessary to refer to an equation of a proof from within the same proof. In this case one can place a $(*)$ at the equation and use this as a local number. In the next proof, $(*)$ is then used for a new, different equation. Instead, the true equation numbers should have a global meaning throughout the proceeding. If you need to refer to an equation hidden in a previous proof then it might be a better idea to phrase that equation as a separate lemma, thus making it visible to the main text. Finally, putting boxes around equations or decorating them in some other way looks childish and should be avoided. During a lecture this might be an option to draw the attention to certain parts of the blackboard, but in a mathematical text it is the explanation of the equations which stresses its importance.

The definitions, lemmas etc. should get *numbers* as well. Typically one includes the number of the section and then uses the same counter for all the mathematical environments together: nothing is more disturbing than Lemma 4 following Definition 15. Only the proofs will get no numbers, they are attached immediately to the lemma, proposition, or theorem they belong to. There might be exceptions to this rule, e.g. if the main theorem of the proceeding is stated already in the introduction. As such it might not even get a number at all but is denoted as **Main Theorem**. A single exception from the rule typically stresses the importance even better than just calling it **Theorem 1.1**. Then at some point in the last section one is in the position to actually prove the main theorem. In this case, one should of course indicate that now the proof of the main theorem really starts.

Beside the numbers it is a good service to the reader to give *titles* for the more important mathematical structures. If in a definition the notion of a manifold is defined, then the definition should be set as **Definition 2.1 (Manifold)** or similar. Also the important mathematical statements like the theorems should get a name like this. For theorems it is sometimes common to call them by the person who actually found them first, like **Theorem 3.2 (Stokes)**. Lemmas and propositions usually do not get such names, neither do corollaries or remarks. The title of a definition or a theorem may also contain a reference to the place in the literature where this particular statement occurred first. Alternatively, one can put the reference right before in the main text.

The numbering of equations, definitions and theorems etc. can then be used to give *cross-references* also within the proceeding. This is typically a good service to the reader in complicated proofs. Note, however, that a descriptive cross-reference like "We use now the uniform continuity of f ..." is typically more informative than a reference just quoting the corresponding lemma as "We use now Lemma 2.1 ...". Ideally, one can combine both aspects as "We use now the uniform continuity of f as obtained in Lemma 2.1 ...", which has the advantage that the reader knows which result is used *and* where it was obtained.

The *bibliography* should be ordered in a reasonable way. Most common in mathematical texts is the alphabetical ordering by the names of the authors. Moreover, each entry should get either a number or a short label under which it is referred to in the main text. Both ways are common and have their benefits and difficulties. It is important to stick to one convention and to format the bibliography in a consistent way. In fact, this is by far not trivial. One has to decide on many little details like whether one wants a comma after the volume of the journal, whether the first names of the authors should be abbreviated, whether years are in brackets, or whether page numbers are at the end of the entry etc. None of these decisions is really important, but if one formats the bibliography by hand this usually gets mixed up completely. Then the result looks very disturbing and unprofessional. Fortunately, LaTeX provides tools to handle this automatically and uniformly.

It is an ongoing debate whether formulas should be treated as syntactical entities of a sentence or not: is e.g. the sign "=" in an equation to be interpreted as a verb "is equal to" or should the sentence still have a verb beyond that? There seems to be no real consensus on this question, but it has quite an impact on how the typesetting is done. Among the consequences is the question whether and how one should put *punctuation* inside the formulas. Should they end with a "." when they occur at the end of a sentence or does that disturb the mathematical meaning? Personally, I prefer to have correct punctuation inside formulas treating the parts of the formula grammatically as if read aloud. But there are arguments against this. In any case, one should stick to a uniform way to handle this instead of making random decisions at every new equation.

4 The Struggle with LaTeX

Every mathematician has to get some reasonably good knowledge and practice with LaTeX. There is simply no alternative producing even a vaguely comparable quality in the typesetting of mathematics. There are many textbooks on LaTeX, like [3, 6], where one can get from first steps and basic introductions everything to the more fancy details of typesetting, package programming, and beyond. Luckily, it is rarely necessary to get such a deep understanding of the LaTeX-system. In this section we will thus not give a general introduction to LaTeX as this can be found elsewhere. We just list a fairly random collection of common mistakes, problems, and solutions which occur frequently in RiG proceedings (and all sorts of theses). Of course, the following comments reflect a personal style, which is, nevertheless, based on a long experience.

Many mathematical structures are needed for every generic text in mathematics. For this reasons Chair X has issued a small style file called `nchairx` containing some of the most important mathematical structures together with a collection of useful mathematical macros. This style file can be obtained at

`https://www.ctan.org/pkg/nchairx`

and should be part of any current LaTeX-distribution. One can include it at the beginning of the LaTeX-file as usual. It requires several other packages to be loaded and will do that implicitly and partially with specific options. Even if these macros are not used, for a RiG at Chair X your own macros *must not conflict* with `nchairx` since this style file will be used when all the RiG proceedings will be put together into a big file to produce the actual proceeding volume of the RiG. For a collection of other useful packages see Table 2.

Whether or not you make use of the package `nchairx` the following hints and suggestions can make your life easier:

- Use `utf8` and `T1`.

  This allows to place funny German umlauts and other obscure accents directly in the source code and increases its readability tremendously. Since this is by now available on almost all reasonable platforms, the inter-operability is still guaranteed. For font encoding, `T1` is the reasonable choice.

- Use `nag`.

  Before you put the actual `\documentclass` it is quite useful to load the `nag` package by

  ```
  \RequirePackage[l2tabu, orthodox]{nag}
  ```

  in order to get warned if old and silly LaTeX-commands are used. Take a look at the log-file for the results of `nag`. After completing the proceeding, this line can be removed.

- Use short hard-coded lines.

  Even though modern editors can handle arbitrarily long lines and LaTeX has no difficulty with this either, one should stick to some reasonable limit of say *80 characters per line*. Good editors will insert line breaks automatically. One reason for this is that version control systems like `git` are typically line-based. They work most efficiently if you have short lines.

- Use comments at the beginning of your file explaining its purpose.

  A couple of lines of comments at the very beginning of your source code will help you and perhaps others a lot: it is the proceeding for the RiG XY with title, author and date.

- Use comments also at other places in your source code.

  It will help you to organize your code a lot.

- Never use \\ to start a new line.

  There is almost *no* reason to use \\ in LaTeX. Never, never do that unless you really have a good reason (**never!**, really, little dolphin babies will die!). To start a new paragraph, one uses an empty line. Even better, place two empty lines in the source code so that the new paragraph is found easier: this way they can be distinguished easier from accidental empty lines. Paragraphs consisting of a single and short sentence are typically never a good idea. The usage of \\ is reserved for special LaTeX-environments like tables and certain math environments.

- Do not use super-exotic symbols.

  In particular, never use funny symbols for mathematical entities which otherwise have some rather common standard notation. Of course, notation is always debatable but the more baroque the notation becomes, the harder the text is to read. If there is a commonly accepted notation then it is almost always a good advice to use it.

- Use `\colon` for maps.

  For specifying a map use

  $$f\colon A \longrightarrow B \quad \text{instead of} \quad f : A \longrightarrow B.$$

  The spacing becomes much better this way.

- Organize your LaTeX-source code.

  If some mathematical structure occurs often in your text, the usage of macros will help you to typeset it in all instances in the same way. Nothing is more distracting than the same mathematical expression set in slightly different ways, simply because one forgot to do it right.

- Use macros to organize you source code, not to abbreviate it.

  There is no point in calling the real numbers `\R` since every reasonable editor will allow you to get a `\mathbb{R}` with a single keystroke. The gain with typing less in having short macros is clearly out-weighted by the loss of having an unreadable source code. Using such unreadable short macros in your text you will never be able to use the text again after a couple of years. If, however, you use a macro like `\ring{}` to typeset a ring in a particular font, then the source code becomes even more readable since you know that this `\ring{R}` (producing R) is now a ring and not just some random symbol R (generated by `\mathsf{R}`). This way, your source code will probably become longer but the ultimate benefit is drastic. The code becomes maintainable also by other persons. This aspect might not yet be an issue for a RiG proceeding, but at some point you may need (want!) to collaborate with other persons and edit LaTeX-files together. Needless to say, `nchairx` provides already many such macros for the daily use.

- Use macros to organize your code (we had that already) by using reasonable names for the macros.

  Five days after you edited your RiG proceeding you will have forgotten what all these macros with the funny names `\sAxss`, `\Sfg`, `\cs_Arw` etc. were good for. However, macros like `\algebra{A}`, `\Schouten{X, Y}`, etc. will be easy to remember. This becomes an important issue as soon as you want to collaborate with others. The `nchairx`-package will provide already many of these macros, just use them.

- Use comments to explain your macros.

  If someone else has to handle your source file it will be absolutely crucial to understand your macros. Here a short comment on what the macro is supposed to do is really helpful. It will even help yourself to understand your own files after some years. And yes, it might be interesting and necessary to get back and use such files again, even after a couple of years. And yes, you will be very annoyed by the person who did not explain the macros a couple of years before.

- Use the system of symbolic labels and references to place references to every structure in your text.

  Even the smallest subitem of a list in a list in a proposition should get its own label by which it is referred to. *Never* put hard-coded references in a LaTeX-file as the numbering might change in unpredictable ways in the end.

- Use reasonable names for your labels.

  Something like `\label{1}` is of course completely pointless. Better include the type of mathematical structure into the label and give it a long and unique name like `\label{definition:Manifo` or `\label{theorem:Stokes}`. This will help you to organize the source code considerably. Remember also that in the end all contributions to the RiG will be put together and this might cause collisions of label naming. Hence the more detailed the names are the less probable are conflicts. One advantage is that reasonable editors will help you a lot with the labels and references while typing. Having labels of a specific type for every specific mathematical structure gives additional benefits: with a good editor, you will never have to type the long label again, it can be typically just selected from a list, maybe even from a list of only lemma-labels etc.

- Avoid blanks in your labels.

  Some editors get confused with labels containing too many white space like `\label{my very long label}` and produce line breaks in between. This might then confuse LaTeX yielding hard to find missing references.

- Avoid any funny symbols in labels.

  Even though LaTeX is by now fairly robust your file might be used on a different computer using a different encoding system. This will then produce missing references of very obscure kind. Just stay with the very standard alphabet and perhaps very few interpunctations.

- Capitalize references to mathematical structures.

  When referring to a lemma with its number, capitalize it as “Lemma 2.1”. It is also very convenient and improves the readability a lot if you place a non-breakable space between the name of the mathematical structure and the actual reference, i.e. `Theorem~\ref{theorem:stokes}` instead of `Theorem \ref{theorem:stokes}`. When referring in a more colloquial style like e.g. “the following lemma”, the keyword lemma is not capitalized. The same applies of course to sections, theorems, definitions, etc.

- Use `bibtex` to manage your bibliography.

  This is absolutely crucial since otherwise it will become a pain to get a reasonably looking consistent bibliography. Never mind about the bibtex-style you should use. In the end, there might be some particular style file used for the final version of the RiG proceeding, after all have been put together. For your own editing, use some standard style file allowing for references by numbers.

- Use precise references.

  When referring to other texts, put the details of a reference into the corresponding `\cite` command like `\cite[Thm.~2.3]{kontsevich:2003a}`. This will produce nice looking references in a consistent way.

- Use reasonable names for the labels in your bibliography.

  Bibitems with labels like `\bibitem{a}` are of course again completely ridiculous. A good way is e.g. to use the last names of the authors, separated by `.` followed by `:` and the year, then `a`, `b`, etc. depending on the number of the publication by these authors in this year. Typically, there will be not much more than very few publications by the same authors in the same year, so this will produce unique labels. Again, a good editor will take care that you do not have to type the labels by hand. Instead, the editor will automatically read the `bibtex` files and offer you the relevant labels based on a search.

- Use pictures as floats.

  We all like pictures. If you have pictures in your text, never try to place them by hand. This will almost surely end in a typesetting catastrophe. The page-break algorithm of LaTeX is reasonably good and certainly much better than anything done by hand from a non-professional typesetter, i.e. you. Simply use the pictures as floats and do not try to think too much about where they will end up. This will change during the preparation of the text anyway. Label your figures and make references to them. The same applies of course to (larger) tables. Place a reasonable description with `\caption` below the picture to quickly explain its relevance.

- Use TikZ pictures.

  If the pictures should have reasonable high quality, they should be vector graphics. One extremely powerful way to produce beautiful pictures is by means of TikZ which is a collection of LaTeX-macros designed for the purpose to produce high-quality pdf-based pictures. Since in the beginning, TikZ is quite demanding to learn, it is a good idea to program the code necessary for a TikZ picture in a separate file. This can be done e.g. using the `standalone` package. Then the main file includes the picture as a pdf-file using `\includegraphics{filename}`. For this purpose the picture file has to be in the search path. Beside the system search path one can set an additional search path using the command `\graphicspath{}`.

- Use pictures with a reasonable size.

  Concerning the size of a picture, it is better for the optical appearance to have pictures with approximately the full width of the text, say 80% – 90% of `\textwidth`. Otherwise there will be large white parts of the page which disturb the grey-value enormously. While for equations this is sometimes not avoidable, a clever design of the picture can typically achieve that feature. Sometimes it is also an option to put two pictures into one `figure`-environment side by side to get a more homogeneous grey-value of the page. In general, colors are nice to be used.

- Draw commutative diagrams with TikZ.

  The TikZ package also offers very powerful tools to design commutative diagrams. In fact, they should always be done with this package, no other has similar flexibility and quality. The basic TikZ command one should take a look at is `\matrix`. Also matrices with more elaborate structure inside like dots, blocks, etc. can be set most easily with the TikZ command `\matrix`. Some higher level handling of commutative diagrams can also be obtained with the `tikz-cd` package.

- Never use hard-coded spaces like `\quad`, `\;`, etc.

  Unless you *really* know what you are doing, doctoring around in equations with such extra spaces will typically not improve the appearance. Instead, you will forget to do it homogeneously and this way produce a messed-up text. In the end, it will be extremely difficult to find them and to get rid of such extra spaces again. If you really need to put extra spaces then use a macro to adjust things once and for all in a uniform way.

- Never try to make line breaks or even page breaks by hand (again, the dolphin babies. . .).

  This is almost never a good idea and the algorithm of LaTeX works way better than every other possibility. If you have overfull lines try to make a hyphenation explicit. It sometimes confuses LaTeX if words and mathematics are connected like in A-spaceoid. Also long formulas in the text might cause problems. Here one can either split it into separate parts like `$a$, \ldots, $b$` instead of `$a, \ldots, b$` or use separate equations instead. In the end, your file will be included into some bigger context causing all line breaks and page breaks to change anyway. Then hard-coded breaks are very hard to find and certainly will mess up things a lot.

- Do not use bold face excessively.

  This disturbs the grey-value of the page quite a bit and leads to very uneasy and bumpy appearance. Of course, sometimes it is reasonable to denote certain mathematical quantities in bold face. Just avoid having too many of them. If text needs to be emphasized, use `\emph{Important}` instead.

- Do not use footnotes.

  If something *is* important then state it in the main text. If something *is not* important, leave it out. Footnotes are used in the humanities, not in mathematics.

- Never start a sentence with a mathematical symbol.

  This will produce typically very disturbing effects, in particular if the symbol is in lowercase. Rephrase the sentence appropriately to avoid this, e.g. write "The function f maps . . . " instead of "f maps . . . ".

- Use ℓ instead of l.

  For summation indices and beyond, it is often a good idea to use ℓ instead of l. This makes things like $\sum_{\ell=1}$ much more readable than $\sum_{l=1}$.

- Use `geometry` to adjust the page size.

  Usually, one should not change the page dimensions much. If needed, the `geometry` package provides the necessary tools for this. It has several standard sizes adapted e.g. to the European DIN A4 page size. While editing, it is usually a good idea to have rather large pages (safe a tree!) for proof reading. In the end for the final copies smaller pages will typically increase the readability (kill the rain forest!). Finding a good balance is not trivial at all. A good discussion on the page design can be found e.g. in [1, 7].

5 Useful Tools

There are many little helpers which will ease the pain of producing LaTeX-files. The most important choice is certainly the editor. But even beyond a good editor one can make life easier by using a version control system, tools to handle the bibliography, `Makefiles` for automatic generation of all related files and much more.

Asking for help is often much better than not asking for help[1]. On the internet there are several Question&Answer places specifically for LaTeX and friends like e.g.

`http://tex.stackexchange.com/`

or the general LaTeX-sites

`http://www.dante.de/` and `http://www.ctan.org/`

for further style files, documentation, and other resources related to LaTeX.

Concerning the *editor*, there are several editors available which are designed specifically to support LaTeX (like Kile, Texmaker, and many more). Other more generic editors support not only LaTeX by special commands but also other particular types of files (like Emacs or Vi). Wikipedia lists a large number of possibilities

`http://en.wikipedia.org/wiki/Comparison_of_TeX_editors`

with some comparison among them. Important aspects are syntax highlighting, multi-file handling when it comes to larger projects like a thesis, a good label and reference system including bibliography and index handling, keyboard shortcuts for particular commands like math symbols or special fonts, good support for references from the bibliography, handling of various `bib`-files, and jumping back and forth between the editor and the pdf-file for debugging. Of course, an editor is only as good as its configuration. It might take some time to figure out a useful configuration suitable for the personal purposes.

As a personal suggestion `Emacs` in combination with `AucTeX` and `RefTeX` is very suitable for all types of documents. The learning curve is rather step but the overall pay-off is tremendous. It is perhaps the editor with most flexible configuration properties.

The *document viewer* should be capable to read the meta-information from `synctex` which allows you to jump back to the source code in the editor. This is supported by several combinations of editor and viewer like `Emacs` and `Okular`. For correcting typos this is an enormous help.

[1]Surprise, surprise! But remember: never use footnotes.

```
\usepackage[english,                % default language
            strings]{babel}         % with string handling
\usepackage{makeidx}                % index generation (optional)
\usepackage{amsmath}                % ams mathematical stuff*
\usepackage{amssymb}                % ams mathematical stuff*
\usepackage[utf8]{inputenc}         % smart input of funny chars
\usepackage[T1]{fontenc}            % also for the font encoding
\usepackage{longtable}              % tables longer than one page
\usepackage{exscale}                % large summation signs in 11pt
\usepackage[final]{graphicx}        % to include pdf pictures*
\usepackage[sort]{cite}             % nicer citations
\usepackage{array}                  % nicer tables
\usepackage{suffix}                 % macro handling*
\usepackage{mathtools}              % a life saver for math macros*
\usepackage[amsmath,                % compatibility with amsmath
            thmmarks,               % marks and
            hyperref]{ntheorem}     % hyperrefs for your theorems*
\usepackage{enumitem}               % handling of lists*
\usepackage{tensor}                 % tensor and indicies*
\usepackage{fancyhdr}               % nicer headers
\usepackage[a4paper]{geometry}      % geometry of page layout
\usepackage{gitinfo2}               % git info
\usepackage[multiuser]{fixme}       % correction notes, warnings etc.
\usepackage{xspace}                 % better spacing after macros
\usepackage{tikz}                   % pictures (optional)
\usepackage{ifdraft}                % determine whether draft mode
\usepackage{nchairx}                % the new Chair X style
\usepackage[expansion=false         % no font expansion
           ]{microtype}             % but protrusion
\usepackage[nottoc]{tocbibind}      % refs and index in the toc
\usepackage[backref=page,           % backrefs in the bibliography
            final=true,             % always treat as final
            pdfpagelabels,          % use pdf page labels
            hypertexnames=true,     % needs both to get index and
            naturalnames=true       % crossrefs correct
           ]{hyperref}              % hyperrefs, always at the end
```

Table 2: Useful LaTeX-packages. Some should be deactivated in the final version. The packages marked with "*" are loaded automatically by `nchairx` and can be thus omitted when `nchairx` is used. For `TikZ` you might want to include additional packages and libraries.

Automatizing becomes a real issue as soon as the projects become larger. This might not yet be the case for a RiG proceeding but here one can try first steps and learn the basic concepts already. If one has several pictures based on TikZ it is clever to place them in a separate directory, compile them there, and include the final pdf-files of the pictures into the main LaTeX-file by `\includegraphics`. Of course, one wants to run LaTeX on all picture files from time to time, e.g. if the macro collection has changed. In order to make this automatic for many (say hundreds) of pictures it is almost unavoidable to use a systematic approach. Here the usage of the GNU `make` program proved to be the right choice. This can handle multiple dependencies, works over different directories and is highly configurable. Of course, the possibilities of `make` go far beyond the usage for LaTeX-projects and one certainly will not need the full functionality. Nevertheless, it is a highly developed tool for which a lot of useful documentation and examples can be found on the internet. The `make` program becomes even more powerful in combination with e.g. `latexmk` or a similar suite designed specifically for the needs of LaTeX.

When producing pictures it might be necessary to use a *function-plotter* at some point. Of course there are many very powerful commercial products accomplishing this. Nevertheless, the quite old but still powerful program `gnuplot` available at e.g.

`https://gnuplot.sourceforge.net/`

should be taken into account since it allows a wonderful integration into the TikZ environment working on all relevant platforms. Moreover, it is capable of scripting and hence allows for automatizing the production of the pictures very well. For function plots it is always a good idea to make them by means of a script: only this way the precise settings can be specified in a way which makes the procedure reproducible. If one wants to change some parameters of the plot later it is extremely handy to have everything documented in form of a script. At least, one should document the precise settings of the function plotting program for every plot.

Already for a proceeding but certainly for larger projects one should consider to use a *version control system*. If working on different computers a version control system can also help in keeping the various copies synchronized. Here the `git` program proved to be very efficient and useful. In particular, for synchronization between different computers and different authors, a decentralized version control system like `git` is the right choice. Originally, `git` was developed to manage the Linux kernel, but its usage is not limited to this at all. Every line-based text file can be put under version control in a very efficient way. Beside many textbooks like [2], one finds extensive documentation on the internet e.g. at

`http://git-scm.com/`.

There are specific LaTeX-packages like `gitinfo2` which make the `git`-information available in the LaTeX-file directly. Some examples can be seen on the title page and in the footer. This is very useful for proof-reading. Do not use `svn` but watch the video of Linus Torvalds explaining why.

If using tools like `make` and `git`, it might be a good choice to use the full power of a unixoid operating system as well. These tools are available for other platforms, too, but they will develop their full strength only in combination with the standard *GNU tool chain* including things like `bash` shell scripting, `grep`, `sed`, and `awk` to name just a few.

References

[1] Bringhurst, R.: *The Elements of Typographic Style.* Hartley & Marks, Vancouver, version 3.2. edition, 2008. 6, 14

[2] Loeliger, J.: *Version Control with Git.* O'Reilly, Sebastopol, 2009. 16

[3] Mittelbach, F., Gossens, M.: *Der LaTeX-Begleiter.* Pearson Studium, München, 2. edition, 2005. 8

[4] Oxford University Press, Oxford. *New Oxford Style Manual*, 2012. 6

[5] Strunk, W., White, E. B.: *The Elements of Style.* Pearson Education, Harlow, 4. edition, 2009. Fifthies Anniversary Edition. 6

[6] Voss, H.: *Mathematiksatz mit LaTeX.* Lehmanns Media, Berlin, 2. edition, 2012. 8

[7] Willberg, H. P., Forssmann, F.: *Lesetypographie.* Hermann-Schmidt Verlag, Mainz, 2010. 6, 14