I went to the woods because I wanted to live deliberately. I wanted to live deep and suck out all the marrow of life. To put to rout all that was not life; and not, when I had come to die, discover that I had not lived.

Neil Perry - *Dead Poet Society*

CHAPTER 1 Introduction

Contents

This thesis addresses the issue of translating mathematical expressions from LaTeX to the syntax of Computer Algebra Systems (CAS), which is typically a time-consuming and error-prone task in the modern life of many researchers. A reliable and comprehensive translation approach requires analyzing the textual context of mathematical formulae. In turn, research advances in translating LaTeX contribute directly towards related tasks in the Mathematical Information Retrieval (MathIR) arena. In this chapter, I provide an introduction to the topic. Section 1.1 introduces my motivation and provides an overview of the problem. Section 1.2 summarizes the research gap. In Section 1.3, I define the research objective and research tasks of this thesis. Section 1.4 concludes with an outline of the thesis including an overview of the publications that contributed to the goals of this thesis and the research path that led to these publications.

1.1 Motivation & Problem

Consider a researcher is working on Jacobi polynomials and examines the existing English Wikipedia article about the topic[1]. While she might be familiar with the Digital Library of Mathematical Functions (DLMF) [98], a standard resource for Orthogonal Polynomials and Special Functions (OPSF), the equation 1.1 from the article might be new to her

$$P_n^{(\alpha,\beta)}(x) = \frac{\Gamma(\alpha+n+1)}{n!\,\Gamma(\alpha+\beta+n+1)} \sum_{m=0}^{n} \binom{n}{m} \frac{\Gamma(\alpha+\beta+n+m+1)}{\Gamma(\alpha+m+1)} \left(\frac{z-1}{2}\right)^m . \tag{1.1}$$

In order to analyze this new equation, e.g., to validate it, she wants to use CAS. CAS are powerful mathematical software tools with numerous applications [207]. Today's most widely

[1]https://en.wikipedia.org/wiki/Jacobi_polynomials [accessed 2021-10-01].
Hereafter, dates follow the ISO 8601 standard. i.e., YYYY-MM-DD.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-658-40473-4_1.

A. Greiner-Petter, *Making Presentation Math Computable*,
https://doi.org/10.1007/978-3-658-40473-4_1

Table 1.1: Different representations of a Jacobi polynomial.

System	Representation
Rendered Version	$P_n^{(\alpha,\beta)}(\cos(a\Theta))$
Generic LaTeX	`P_n^{(\alpha, \beta)}(\cos(a\Theta))`
Semantic LaTeX	`\JacobipolyP{\alpha}{\beta}{n}@{\cos@{a\Theta}}`
Maple [36]	`JacobiP(n, alpha, beta, cos(a*Theta))`
Mathematica [393]	`JacobiP[n, \[Alpha], \[Beta], Cos[a \[CapitalTheta]]]`
SymPy [252]	`jacobi(n,Symbol('alpha'),Symbol('beta'),cos(a*Symbol('Theta')))`

used CAS include Maple [36], Mathematica [393], and MATLAB [246]. Scientists use CAS[2] to simplify, manipulate, evaluate, compute, or even visualize mathematical expressions. Thus, CAS play a crucial role in the modern era for pure and applied mathematics [8, 184, 207, 262] and even found their way into classrooms [237, 363, 365, 389, 390]. In turn, CAS are the perfect tool for the researcher in our example to examine the formula further. In order to use a CAS, she needs to translate the expression into the correct CAS syntax.

Table 1.1 illustrates the differences between computable and presentational encodings for a Jacobi polynomial. While the rendered version and the LaTeX [220] encoding only provide visual information, semantic LaTeX [403] and the CAS encodings explicitly encode the meaning, i.e., the semantics, of the formula. On the one hand, LaTeX[3] has become the de-facto standard to typeset mathematics in scientific publications [129, 248, 402], especially in the domain of Science, Technology, Engineering, and Mathematics (STEM). On the other hand, computational advances make CAS an essential asset in the modern workflow of experimenting and publishing in the Sciences. Translating expressions between LaTeX and CAS syntaxes is, therefore, a typical task in the everyday life of our hypothetical researcher. Despite this common need, no reliable translation from a presentational format, such as LaTeX, to a computable format, such as Mathematica, is available to date. The only option our hypothetical researcher has is to manually translate the expression in the specific syntax of a CAS. This process is time-consuming and often error-prone.

Problem: No reliable translation from a presentational mathematical format to a computable mathematical format exists to date.

If a translation between LaTeX and CAS is so essential, why are there no translation tools available? As is often the case in research, the reasons for this are diversified. First, there are translation approaches available. Some CAS, such as Mathematica and SymPy, allow to import LaTeX expressions. Most CAS support at least the Mathematical Markup Language (MathML), since it is the current web standard to encode mathematical formulae. With numerous tools available to transfer LaTeX to MathML [18], a translation from LaTeX to CAS syntaxes should not be a difficult task. However, none of these available translation techniques are reliable

[2]In the sequel, the acronym CAS is used interchangeably with its plural.
[3]https://www.latex-project.org/ [accessed 2021-10-01]

Table 1.2: Examples of Mathematica's LaTeX import function `ToExpression["x", TeXForm]`. Tested with Mathematica [393] v.12.1.1. The second sum in row 8 (marked with ?) is only partially correct. Since the second summand contains the summation index n, the second summand should be part of the sum.

LaTeX	Rendering	Import	Result
`\int_a^b x dx`	$\int_a^b x dx$	`Error`	✗
`\int_a^b x \mathrm{d}x`	$\int_a^b x \mathrm{d}x$	`Error`	✗
`\int_a^b x\, dx`	$\int_a^b x\, dx$	`Integrate[x, {x, a, b}]`	✓
`\int_a^b x\; dx`	$\int_a^b x\; dx$	`Error`	✗
`\int_a^b x\, \mathrm{d}x`	$\int_a^b x\, \mathrm{d}x$	`Error`	✗
`\int_a^b \frac{dx}{x}`	$\int_a^b \frac{dx}{x}$	`Error`	✗
`\sum_{n=0}^N n^2`	$\sum_{n=0}^N n^2$	`Sum[n^2, {n, 0, N}]`	✓
`\sum_{n=0}^N n^2 + n`	$\sum_{n=0}^N n^2 + n$	`Sum[n^2, {n, 0, N}] + n`	?
`{n \choose m}`	${n \choose m}$	`Error`	✗
`\binom{n}{m}`	$\binom{n}{m}$	`Binomial[n, m]`	✓

and comprehensive. Table 1.2 illustrates how Mathematica, one of the major proprietary CAS, fails to import even simple formulae. Another option is SnuggleTeX [251], a LaTeX to MathML converter which also supports translations to Maxima [324]. SnuggleTeX fail to translate all expressions in Table 1.2. Alternative translations via MathML as an intermediate format perform similarly (as we will show later in Section 2.3).

While the simple cases shown in Table 1.2 could be solved with a more comprehensive and flexible parser and mapping strategy, such a solution would ignore the real challenge of translating mathematics to CAS, the ambiguity. The interpretation of the majority of mathematical expressions is *context-dependent*, i.e., the same formula may refer to different concepts in different contexts. Take the expressions $\pi(x + y)$ as an example. In number theory, the expression most likely refers to the number of primes less than or equal to $x + y$. In another context, however, it may just refer to a multiplication $\pi x + \pi y$. Without considering the context, an appropriate translation of this ambiguous expression is infeasible. Today's translation solutions, however, do not consider the context of an input. Instead, they translate the expression based on internal decisions, which are often not transparent to a user.

Table 1.3 shows the results of importing $\pi(x + y)$ to different CAS. Each CAS in Table 1.3 interprets π as a function call but does not associate it with the prime counting function (nor any other predefined function). Only SnuggleTeX translated π as the mathematical constant to Maxima syntax. However, Maxima does not contain a prime counting function. The CAS import functions consider the expression as a generic function with the name π. Mathematica surprisingly links π still with the mathematical constant which results in a peculiar behaviour for numeric evaluations. The expression `N[Pi[x+y]]` (numeric evaluation of the imported expression) is evaluated to $3.14159[x + y]$. Associating the variables x and y with numbers, say $x, y = 1$, would result in the rather odd expression $3.14159[2]$.

Table 1.3: The results of importing $\pi(x+y)$ in different CAS. For Maple, a MathML representation was used. Content MathML was not tested, since there is no content dictionary available that defines the prime counting function. SnuggleTeX translated the expression to the CAS Maxima. The two right most columns show the expected expressions in the context of the prime counting function or a multiplication. None of the CAS choose any of the two expected interpretations. Note that the prime counting function in Maple can also be written with `pi(x+y)` and requires to pre-load the extra package `NumberTheory`. Nonetheless, this function `pi(x+y)` is still different to the actual imported expression `Pi(x+y)`. Note further that Maxima does not define a prime counting function.

	Translated	**Expected Expression**	
System	**Expression**	**Number of primes**	**Multip.**
Maple [36] v.2019	`Pi(x+y)`	`PrimeCounting(x+y)`	`Pi*(x+y)`
Mathematica [393] v.12.1.1	`Pi[x+y]`	`NPrimes[x+y]`	`Pi*(x+y)`
SymPy [252] v.1.8	`pi(x+y)`	`primepi(x+y)`	`pi*(x+y)`
SnuggleTeX [251] v.1.2.2	`%pi*(x+y)`	-	`%pi*(x+y)`

Why do existing translation techniques not allow to specify a context? Mainly because it is an open research question of what this context is or needs to be. The exact information needs to perform translation to CAS syntaxes, and where to find them is unlcear [11]. Some required information is indeed encoded in the structure of the expression itself. Consider a simple fraction $\frac{1}{2}$. This expression is context-independent and can be directly translated. The expression $P_n^{(\alpha,\beta)}(x)$ in the context of OPSF is also often unambiguous for general-purpose CAS. Since Mathematica supports no other formula with this presentational structure, i.e., P followed by a subscript and superscript with paranthesis, Mathematica is able to correctly associate $P_{\bullet}^{(\bullet,\bullet)}(\bullet)$, where $\bullet$ are wildcards, with the function `JacobiP`. In other cases, the immediate textual context of the formula provides sufficient information to disambiguate the expression [54, 329]. Consider, an author explicitly declares $\pi(x)$ as the prime counting function right before she uses it with $\pi(x+y)$. In this case, it might be sufficient to scan the surrounding context for key phrases [183, 214, 329], like 'prime counting function' in order to map π to, for instance, `NPrimes` in Mathematica.

Often, the semantic explanations of mathematical objects in an article are scattered around in the context or absent entirely [394]. An interested reader needs to retrieve sufficient semantic explanations and correctly link them with mathematical objects in order to comprehend the meaning of a complex formula. Sometimes, an author presumes the interpretation of an expression can be considered as common knowledge and, therefore, does not require further explanations. Consider $\pi(x+y)$ refers to a multiplication between π and $(x+y)$. In general, an author may consider π (the mathematical constant) as common knowledge and does not explicitly declare its meaning. The same could be true for scientific articles, where the length is often limited. An article about prime numbers probably not explicitely declare the meaning of $\pi(x+y)$ because the author presumes the semantics are unambiguis given the overall context of the article.

In other cases, the information needs go beyond a simple text analysis. Consider $\pi(x+y)$ as a generic function that was previously defined in the article and simply has no name. An appropriate translation would require to retrieve the definition of the function from the context. But even if a function is well-known and supported by a CAS, a direct translation might be inappropriate because the definition in the CAS is not what our researcher expected [3, 13]. Legendre's incomplete elliptic integral of the first kind $F(\phi, k)$, for example, is defined with the amplitude ϕ as its first argument in the DLMF and Mathematica. In Maple, however, one needs to use the sine of the amplitude $\sin(\phi)$ for the first argument[4]. In turn, an appropriate translation to Maple might be `EllipticF(sin(phi), k)` rather than `EllipticF(phi, k)` depending on the source of the original expression. The English Wikipedia article about elliptic integrals[5] contains both versions and refers to them with $F(\phi, k)$ and $F(x; k)$ respectively. Even though both versions in Wikipedia refer to the same function, correct translations to Maple of $F(\phi, k)$ and $F(x; k)$ are not the same.

In cases of multi-valued functions, translations between different systems can become eminently more complex [83, 91, 172]. Even for simple cases, such as the arccotangent function $\operatorname{arccot}(x)$, the behavior of different CAS might be confusing. For example, since $\operatorname{arccot}(x)$ is multi-valued, there are multiple solutions of $\operatorname{arccot}(-1)$. CAS, like any general calculator too, only compute values on the principle branches and, therefore, return only a single value. The principle branches, however, are not necessarily uniformly positioned among multiple systems [84, 172]. In turn, the returned value of a multi-valued function may depends on the system, see Table 1.4. A translation of $\operatorname{arccot}(x)$ from the DLMF to `arccot(x)` in Maple would be only correct for $\Re x > 0$. Finally, CAS may also compute irrational looking expressions without objections, e.g., $\operatorname{arccot}\left(\frac{1}{0}\right)$ returns 1.5708 in MATLAB[6]. Even for field experts, it can be challenging to keep track of every property and characteristic of CAS [20, 100].

Table 1.4: Different computation results for $\operatorname{arccot}(-1)$ (inspired by [84]).

System or Source	$\operatorname{arccot}(-1)$
[276] 1st printing	$3\pi/4$
[276] 9th printing	$-\pi/4$
Maple [36] v.2020.2	$3\pi/4$
Mathematica [393] v.12.1.1	$-\pi/4$
SymPy [252] v.1.5.1	$-\pi/4$
Axiom [173] v.Aug.2014	$3\pi/4$
Reduce [151] v.5865	$3\pi/4$
MATLAB [246] v.R2021a	$-\pi/4$

Problem: Existing LaTeX to CAS converters are context-agnostic, inflexible, limited to simple expressions, and nontransparent.

In combination, all of the issues underline that an accurate manual translation to the syntax of CAS is challenging, time-consuming, error-prone, and requires deep and substantial knowledge about the target system. Especially with the increasing complexity of the translated expressions, errors during the translation process might be inevitable. Real-world scenarios often include

[4]https://www.maplesoft.com/support/help/maple/view.aspx?path=EllipticF [accessed 2021-10-01]

[5]https://en.wikipedia.org/wiki/Elliptic_integral [accessed 2021-10-01]

[6]MATLAB evaluates $\frac{1}{0}$ to infinity and the limit in positive infinity of the arccotangent function is $\frac{\pi}{2}$ (or roughly 1.5708). Yet, the interpretation of the division by zero is not wrong, since it follows the official IEEE 754 standard for floating-point arithmetic [170].

much more complicated formulae compared to the expressions in Table 1.2 or even equation (1.1). Moreover, if an error occurs, the cause of the error can be very challenging to detect and traced back to its origin. The issue of translating $\operatorname{arccot}(x)$ to Maple, for example, may remain undiscovered until a user calculates negative values. If the function is embedded into a more complex equation, even experts can lose track of potential issues. In combination with unreliable translation tools, working with CAS may even be frustrating. Mathematica, for example, is able to import our test expression (1.1) mentioned earlier without throwing an error[7]. However, investigating the imported expression reveals an incorrect translation due to an issue with factorials. To productively work with CAS, our hypothetical researcher from above needs to carefully evaluate if the automatically imported expression was correct. As a consequence, existing translation approaches are not practically useful.

In this thesis, I will focus on discovering the information needs to perform correct translations from presentational formats, here mainly LaTeX, to computational formats, here mainly CAS syntaxes. My personal motivation is to improve the workflow of researchers by providing them a reliable translation tool that offers crucial additional information about the translation process. Further, I limit the support of such a translation tool to general-purpose CAS, since many general mathematical expressions simply cannot be translated to appropriate CAS expressions for task-specific CAS (or other mathematical software, such as theorem provers). The focus on general-purpose CAS allows me to provide a broad solution to a general audience. Note further that, in this thesis, I mostly focus on the two major CAS Maple and Mathematica. However, the goal is to provide a translation tool that is easy to extend and support more CAS.

Further, the real-world applications of such a translation tool go far beyond an improved workflow with CAS. A computable formula can be automatically verified with CAS [51, 52, 2, 8, 13, 153, 184, 414, 415], translated to other semantically enhanced formats, such as OpenMath [53, 57, 119, 152, 303, 361], content MathML [59, 60, 159, 270, 318, 342] or other CAS syntaxes [110, 361], imported to theorem prover [35, 57, 152, 163, 338, 375], or embedded in interactive documents [85, 131, 150, 162, 201, 284]. Since an appropriate translation is generally context-dependent, a translator must use MathIR [141] techniques to access sufficient semantic information. Hence, advances in translating LaTeX to CAS syntaxes also contribute directly towards related MathIR tasks, including entity linking [150, 208, 212, 316, 319, 321, 322], math search engines [92, 181, 182, 203, 211, 236, 274], semantic tagging of math formulae [71, 402], recommendation systems [30, 31, 50, 319], type assistance systems [103, 106, 14, 321, 400], and even plagiarism detection platforms [253, 254, 334].

1.2 Research Gap

Existing translation approaches from presentational formats to computable formats share the same issues. Currently, these translation approaches are

1. context-independent, i.e., a translation of an expression is unique regardless of the context from where the expression came from (see the $\pi(x+y)$ example mentioned earlier);
2. nontransparent, i.e., the internal translation decisions are not communicated to the user, which makes the translation untrustworthy and errors challenging to trace or detect;

[7] If the binomial is given with the `\binom` macro rather than `\choose`.

3. inflexible, i.e., slight changes in the notation can cause the translation to fail (see the integral imports from Table 1.2); and
4. limited to simple expression due to missing mappings between function definition sources, i.e., even with semantic information, a translation often fails.

Issue 4 raises from the fact that there are semantically enhanced data formats that have been specifically developed to make expressions between CAS interchangeable, such as OpenMath [119, 303, 361] and content MathML [318, 343]. Nonetheless, most CAS do not support OpenMath natively [303] and the support for content MathML is limited to school mathematics [318]. The reason is that such translation requires a database that maps functions between different semantic sources. As discussed above, creating such a comprehensive database can be time-consuming due to slight differences between the systems (e.g., positions of branch cuts, different supported domains, etc.) [361]. Hence, for economic reasons, crafting and maintaining such a library is unreasonable. Translations between semantic enhanced formats, e.g., between CAS syntaxes, OpenMath, or content MathML, are consequentially often unreliable.

In previous research, I was focusing on the issues 2-4 by developing a rule-based LaTeX to CAS translator, called LaCASt. Originally, LaCASt performs translations from semantic LaTeX to Maple. Relying on semantic LaTeX allows LaCASt to largely ignore the ambiguity Issue 1 and focus on the other problems. For this thesis, I continued to develop LaCASt to further mitigate the *limitation* and *inflexibility* issues 3 and 4. Further, I focused on extending LaCASt to become the first context-aware translator to tackle the *context-independency* issue 1.

1.3 Research Objective

This doctoral thesis aims to:

Research Objective

Develop and evaluate an automated context-sensitive process that makes presentational mathematical expressions computable via computer algebra systems.

Hereafter, I consider the semantic information of a mathematical expression as sufficient if a translation of the expression into the syntax of a CAS becomes feasible. To achieve the research objective, I define the following five research tasks:

Research Tasks

I Analyze the strengths and weaknesses of existing *semantification* approaches for translating mathematical expressions to computable formats.

II Develop a semantification process that will improve on the weaknesses of current approaches.

III Implement a system for the automated semantification of mathematical expressions in scientific documents.

IV Implement an extension of the system to provide translations to computer algebra systems.

V Evaluate the effectiveness of the developed semantification and translation system.

1.4 Thesis Outline

Chapter 1 provides an introduction for translating presentational mathematical expressions into computable formats. The chapter further defines the research gap for such translations and defines the research objective and tasks this thesis addresses. Finally, it outlines the structure of the thesis and briefly summarizes the main publications.

Chapter 2 provides an overview of related work by examining existing mathematical formats and translation approaches between them. This chapter focuses on **Research Task I** by analyzing the strengths and weaknesses of existing translation approaches with the main focus on the standard formats LaTeX and MathML.

Chapter 3 addresses **Research Task II** by studying the capability of math embeddings, introducing a new concept to describe the nested structure of mathematical objects, and presenting a novel context-sensitive semantification process for LaTeX expressions.

Chapter 4 presents the first context-sensitive LaTeX to CAS translator: LaCASt. In particular, this chapter focuses on **Research Tasks III** and **IV** by implementing the previously introduced semantification process and integrates it into the rule-based semantic LaTeX to CAS translator LaCASt. In addition, the chapter briefly summarizes a context-independent neural machine translation approach to estimate how much structural information is encoded in mathematical expressions.

Chapter 5 evaluates the new translation tool LaCASt and, therefore, contributes mainly towards **Research Task V**. In particular, it introduces the novel evaluation concept of equation verifications to estimate the appropriateness of translated CAS expressions. Our new evaluation concept not only detects issues in the translation pipeline but is also able to identify errors in the source equation, e.g., from the DLMF or Wikipedia, and the target CAS, e.g., Maple or Mathematica. In order to maximize the number of verifiable DLMF equations via our novel evaluation technique, this chapter also introduces some heuristic extensions to the LaCASt pipeline. Hence, this chapter partially contributes to **Research Task IV** too.

Chapter 6 concludes the thesis by summarizing contributions and their impact on the MathIR community. It further provides a brief overview of the remaining issues and future work.

An Appendix is available in the electronic supplementary material and provides additional information about certain aspects of this thesis including an extended error analysis, result tables, and a summary of bugs and issues we discovered with the help of LaCASt in the DLMF, Maple, Mathematica, and Wikipedia.

1.4.1 Publications

Most parts of this thesis were published in international peer-reviewed conferences and journals. Table 1.5 provides an overview of the publications that are reused in this thesis. The first column identifies the chapter a publication contributed to. The venue rating was taken from the Core ranking[8] for conferences and the Scimago Journal Rank (SJR)[9] for journal articles. Each rank

[8] http://portal.core.edu.au/conf-ranks/ with the ranks: A* – flagship conference (top 5%), A – excellent conference (top 15%), B – good conference (top 27%), and C – remaining conferences [accessed 2021-10-01].

[9] https://www.scimagojr.com/ with the ranks Q1 – Q4 where Q1 refer to the best 25% of journals in the field, Q2 to the second best quarter, and so on [accessed 2021-10-01].

was retrieved for the year of publication (or year of submission, in case the paper has not been published yet). Table 1.6 similarly shows publications that partially contributed towards the goal of this thesis but are not reused within a chapter. Note that the publication [3] (in Table 1.6) was part of my Master's thesis and contributed towards this doctoral thesis as a preliminary project. The Journal publication [13] (also in Table 1.6) is an extended and (with new results) updated version of the thesis and the mentioned article [3]. The venue abbreviations in both tables are explained in the glossary. Lastly, note that the TPAMI journal [11] is reused in Chapter 4 (for the methodology) and in Chapter 5 (for the evaluation) to provide a coherent structure. My publications, talks, and submissions are separated from the general bibliography in the back matter and can be found on page 171.

Table 1.5: Overview of the primary publications in this thesis.

Ch.	Venue	Year	Type	Length	Author Position	Venue Rating	Ref.
2	SIGIR	2019	Workshop	Full	1 of 6	Core A*	[9]
	JCDL	2018	Conference	Full	2 of 6	Core A*	[18]
3	Scientometrics	2020	Journal	Full	1 of 7	SJR Q1	[15]
	WWW	2020	Conference	Full	1 of 7	Core A*	[14]
	ICMS	2020	Conference	Full	1 of 4	n/a	[10]
4	TPAMI[10]	2021	Journal	Full	1 of 6	SJR Q1	[11]
5	TACAS	2021	Conference	Full	1 of 8	Core A	[8]
	CICM	2018	Conference	Full	2 of 3	n/a	[2]
6	JCDL	2020	Conference	Poster	2 of 5	Core A*	[17]

Table 1.6: Overview of secondary publications that partially contributed to this thesis.

Year	Venue	Type	Length	Author Position	Venue Rating	Ref.
2020	CLEF	Workshop	Full	4 of 6	n/a	[16]
	EMNLP	Workshop	Full	2 of 4	Core A	[1]
2019	AJIM	Journal	Full	1 of 4	SJR Q1	[13]
2018	CICM	Conference	Short	1 of 4	n/a	[12]
2017	CICM	Conference	Full	4 of 9	n/a	[3]

1.4.2 Research Path

This section provides a brief overview of my research path that led to this thesis, i.e., it discusses the primary publications and the motivations behind them. Every publication is marked with the associated chapter and a reference. This research path is logically (not chronologically) divided into three sections: preliminary work, the semantification of LaTeX, and the evaluation of translations.

Preliminary Work I had the first contact with the problem of translating LaTeX to CAS syntaxes during my undergraduate studies in mathematics. During that time, I regularly used

[10]The methodology part of this journal is reused in Chapter 4 while the evaluation part is reused in Chapter 5.

CAS like MATLAB and SymPy for numeric simulations and for plotting results. At the same time, we were required to hand in our homework as LaTeX files. While exporting content from the CAS to LaTeX files was rather straight forward, the other way around, i.e., importing LaTeX into the CAS, required manual conversions. I decided to explore the reasons for this shortcoming in my Master's thesis. During that time, I developed the first version of a semantic LaTeX to CAS translator, which was later coined LaCASt[11]. The results from this first study were published at the Conference of Intelligent Computer Mathematics (CICM) in 2017.

"Semantic Preserving Bijective Mappings of Mathematical Formulae Between Document Preparation Systems and Computer Algebra Systems" by Howard S. Cohl, Moritz Schubotz, Abdou Youssef, **André Greiner-Petter**, Jürgen Gerhard, Bonita Saunders, Marjorie McClain, Joon Bang, and Kevin Chen. **In:** *Proceedings of the International Conference of Intelligent Computer Mathematics* (CICM), 2017.

Not Reused — [3]

This first version of LaCASt focused specifically on the CAS Maple but was designed modularly to allow later extensions to other CAS. The main limitation of LaCASt, however, was the requirement of using semantic LaTeX macros to disambiguate mathematical expressions manually. An automatic disambiguation process did not exist at the time. Moreover, only a few previous projects focused on a semantification for translating mathematical formats. Hence, I continued my research in this direction.

In the following, I will use 'we' rather than 'I' in the subsequent parts of this thesis, since none of the presented contributions would have been possible without the tremendous and fruitful discussions and help from advisors, colleagues, students, and friends.

Semantification of LaTeX As an alternative for semantic LaTeX, we closely investigated existing converters for MathML first (see Section 2.2.1). Since MathML was (and still is) the standard encoding for mathematical expressions in the web, most CAS support MathML. MathML uses two markups, presentation and content MathML. The former visualizes a formula, while the latter describes the semantic content. Hence, content MathML can disambiguate math much like semantic LaTeX. Since MathML is the official web standard and LaTeX the de-facto standard for writing math, there are numerous of converters available that translate LaTeX to MathML. As our first contribution, we developed MathMLben, a benchmark dataset for measuring the quality of MathML markup that appears in a textual context. With this benchmark, we evaluated nine state-of-the-art LaTeX to MathML converters, including Mathematica as a major CAS. We published our results in the Joint Conference on Digital Libraries (JCDL) in 2018.

"Improving the Representation and Conversion of Mathematical Formulae by Considering their Textual Context" by Moritz Schubotz, **André Greiner-Petter**, Philipp Scharpf, Norman Meuschke, Howard S. Cohl, and Bela Gipp. **In:** *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries* (JCDL), 2018.

Chapter 2 — [18]

[11] ***La****TeX to* ***CAS*** *Translator.*

We discovered that three of the nine tools were able to generate content MathML but with insufficient accuracy. None of the available tools were capable of analyzing a context for a given formula. Hence, the converters were unable to conclude the correct semantic information for most of the symbols and functions. In our study, we proposed a manual semantification approach that semantically enriches the translation process of existing converters by feeding them semantic information from the surrounding context of a formula. The enrichment process was manually illustrated via the converter LaTeXML, which allowed us to add custom semantic macros to improve the generated MathML data. In fact, we used this manual approach to create the entries of MathMLben in the first place.

Naturally, our next goal was to automatically retrieve semantic information from the context of a given formula. Around this time, word embeddings [256] began to gain interest in the MathIR community [121, 215, 242, 400, 404]. It seems that vector representations were able to capture some semantic properties of tokens in natural languages. Can we create such semantic vector representations of mathematical expressions too? Unfortunately, we discovered that the related work in this new area of interest did not discuss a crucial underlying issue with embedding mathematical expressions. In math expressions, certain symbols or entire groups of tokens are fixed, such as the red tokens in the Gamma function $\Gamma(x)$ or the Jacobi polynomial $P_n^{(\alpha,\beta)}(x)$, while other may vary (gray). Inspired by words in natural languages, we call these fixed tokens the stem of a mathematical object or operation. Unfortunately, in mathematics, this stem is context-dependent. If π is a function, the red tokens are its stem $\pi(x+y)$. However, if π is not a function, the stem is just the symbol itself $\pi(x+y)$. If we do not know the stem of a mathematical object, how can we group them so that a trained model understands the connection between variations like $\Gamma(z)$ and $\Gamma(x)$? The answer is: we cannot. The only alternative is to use context-independent representations, e.g., we only embed the identifiers or the entire expression. Each of these approaches has advantages and disadvantages. We shared our discussion with the community at the BIRNDL Workshop at the conference on Research and Development in Information Retrieval (SIGIR) in 2019.

"Why Machines Cannot Learn Mathematics, Yet" by **André Greiner-Petter**, Terry Raus, Moritz Schubotz, Akiko Aizawa, William I. Grosky, and Bela Gipp. **In:** *Proceedings of the 4th Joint Workshop on Bibliometric-Enhanced Information Retrieval and Natural Language Processing for Digital Libraries* (BIRNDL@SIGIR), 2019.

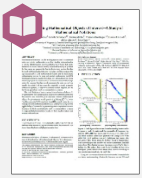
Chapter 2 — [9]

Nonetheless, context-independent math embeddings still have many valuable applications. Search engines, for example, can profit from a vector representation that represents a mathematical expression in a particular context. Such a trained model would still be unable to tell us what the expression is, but it can tell us efficiently if the expression is *semantically similar* (e.g., because the surrounding text is similar) to another expression. Further, embedding semantic LaTeX allows us to overcome the issue of unknown stems for most functions since the macro unambiguously defines the stem. Youssef and Miller [404] trained such a model on the DLMF formulae. Later, we published an extended version of our workshop paper together with Youssef and Miller in the *Scientometrics* journal.

"Math-Word Embedding in Math Search and Semantic Extraction" by **André Greiner-Petter**, Abdou Youssef, Terry Raus, Bruce R. Miller, Moritz Schubotz, Akiko Aizawa, and Bela Gipp. **In:** *Scientometrics* 125(3): 3017-3046, 2020.

Chapter 3 — [15]

Unfortunately, this sets us back to the beginning, where we need manually crafted semantic LaTeX. We started to investigate the issue of interpreting the semantics of mathematical expressions from a different perspective. As we will see later in Section 2.2.4, humans tend to visualize mathematical expressions in a tree structure, where operators, functions, or relations are parent nodes of their components. Identifiers and other terminal symbols are the leaves of these trees. The MathML tree data structure comes close to these so-called *expression trees* (see Section 2.2.4) but does not strictly follow the same idea [331]. The two aforementioned context-independent approaches to embed mathematical expressions take either the leaves or the roots of such trees. The subtrees in between are the context-dependent mathematical objects we need. Not all subtrees, however, are meaningful, and the mentioned expression trees are only theoretical interpretations. In searching for an approach to discover meaningful subexpressions, which we call Mathematical Objects of Interest (MOI), we performed the first large-scale study of mathematical notations on real-world scientific articles. In this study, we followed the assumption that every subexpression with at least one identifier can be semantically important. Hence, we split every formula into their MathML subtrees and analyzed their frequency in the corpora. Overall, we analyzed over 2.5 Billion subexpressions in 300 Million documents and showed that the frequency distribution of mathematical subexpressions is similar to words in natural language corpora. By applying known frequency-based ranking functions, such as BM25, we were also able to discover topic-relevant notations. We published these results at The Web Conference (WWW) in 2020.



"Discovering Mathematical Objects of Interest — A Study of Mathematical Notations" by **André Greiner-Petter**, Moritz Schubotz, Fabien Müller, Corinna Bretinger, Howard S. Cohl, Akiko Aizawa, and Bela Gipp. **In:** *Proceedings of the Web Conference* (WWW), 2020.

Chapter 3 — [14]

The applications that we derived from simply counting mathematical notations were surprisingly versatile. For example, with the large set of indexed math notations, we implemented the first type assistant system for math equations, developed a new faceted search engine for zbMATH, and enabled new approaches to measure potential plagiarism in equations. Besides these practical applications, it also gave us the confidence to continue focusing on subexpressions for our LaTeX semantification. Previous projects that aimed to semantically enrich mathematical expressions with information from the surrounding context primarily focused on one of the earlier mentioned extremes, i.e., the leaves or roots in expression trees [139, 214, 279, 329, 330]. Our study also revealed that the majority of unique mathematical formulae are neither single identifier nor highly complex mathematical expressions. Hence, we concluded that we should focus on semantically enriching subexpressions (subtrees) rather than the roots or leaves. We proposed a novel context-sensitive translation approach based on semantically annotated MOI and shared our theoretical concept with the community at the International Conference on Mathematical Software (ICMS) in 2020.

"Making Presentation Math Computable: Proposing a Context Sensitive Approach for Translating LaTeX to Computer Algebra Systems" by **André Greiner-Petter**, Moritz Schubotz, Akiko Aizawa, and Bela Gipp. **In:** *Proceedings of the International Conference on Mathematical Software* (ICMS), 2020.

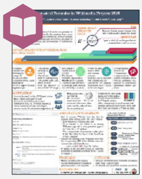
Chapter 3 — [10]

Afterward, we started to realize the proposed pipeline with a specific focus on Wikipedia. We focused on this encyclopedia for two reasons. First, Wikipedia is a free and community-driven encyclopedia and, therefore, (a) less strict on writing styles and (b) more descriptive compared to scientific articles. Second, Wikipedia can actively benefit from our contribution since additional semantic information about mathematical formulae can support users of all experience levels to read and comprehend articles more efficiently [150]. Moreover, a successful translation from a formula in Wikipedia to a CAS makes the formula computable which enables numerous of additional applications. In theory, a mathematical article could be turned into an interactive document to some degree with our translations. However, the most valuable application of a translation of formulae in Wikipedia would be the ability to check equations for their plausibility. With the help of CAS, we are able to analyze if an equation is semantically correct or suspicious. This evaluation would enable existing quality measures in Wikipedia to incorporate mathematical equations for the first time. The results from our novel context-sensitive translator including the plausibility check algorithms have been accepted for publication in the IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) journal and are currently in press.

"Do the Math: Making Mathematics in Wikipedia Computable." **André Greiner-Petter**, Moritz Schubotz, Corinna Bretinger, Philipp Scharpf, Akiko Aizawa, and Bela Gipp. In press: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), 2021.

Chapter 4 & 5 — [11]

Currently, we are also actively working on extending the backbone of Wikipedia itself for presenting additional semantic information about mathematical expressions by hovering over or clicking on the formula. This new feature helps Wikipedia users to better understand the meaning of mathematical formulae by providing details on the elements of formulae. Moreover, it paves the way towards an interface to actively interact with mathematical content in Wikipedia articles. We presented our progress and discussed our plans in the poster session at the JCDL in 2020.



"Mathematical Formulae in Wikimedia Projects 2020." Moritz Schubotz, **André Greiner-Petter**, Norman Meuschke, Olaf Teschke, and Bela Gipp. **In:** Poster Session at the *ACM/IEEE Joint Conference on Digital Libraries* (JCDL), 2020.

Chapters 6 — [17]

Evaluating Digital Mathematical Libraries Alongside this main research path, we continuously improved and extended LaCAST with new features and new supported CAS. Our first goal was to verify the translated, now computable, formulae in the DLMF. The primary motivation behind this approach was to quantitatively measure the accuracy of LaCAST translations. How can we very if a translation was correct? The well-established Bilingual Evaluation Understudy (BLEU) [282] measure in natural language translations is not directly applicable for mathematical languages because an expression may contain entirely different tokens but is still equivalent to the gold standard. Since the translation is computable, however, we can take advantage of the power of CAS to verify a translation. The basic idea is that a human-verified equation in one system must remain valid in the target system. If this is not the case, only three sources of errors are possible: either the source equation, the translation, or the CAS verification was incorrect. With the assumption that equations in the DLMF and major proprietary CAS are mostly error-free, we can translate equations from the DLMF to discover issues within LaCAST. First, we focused on symbolic verifications, i.e., we used the CAS to symbolically simplify the difference between left- and right-hand side of an equation. If the simplified difference is 0, the CAS symbolically verified the equivalence of the left- and right-hand side and confirmed a correct translation via LaCAST. Additionally, we extended the verification approach to include more precise numeric evaluations. If a symbolic manipulation failed to return 0, it could also mean the CAS was unable to simplify the expression. We numerically calculate the difference on specific test values and check if the difference is below a given threshold to overcome this issue. If all test calculations are below the threshold, we consider it numerically verified. Even though this approach cannot verify equivalence, it is very effective in discovering disparity. We published the first paper with this new verification approach based on Maple at the CICM in 2018.

"Automated Symbolic and Numerical Testing of DLMF Formulae Using Computer Algebra Systems" by Howard S. Cohl, **André Greiner-Petter**, and Moritz Schubotz. **In:** *Proceedings of the International Conference on Intelligent Computer Mathematics* (CICM), 2018.

Chapter 5 — [2]

The extension of the system and the new results led us to an extended journal version of the initial LaCAST publication [3]. This extended version mostly covered parts of my Master's thesis and is not reused in this thesis. For technical details about LaCAST, see the journal publication [13]. In Appendix D available in the electronic supplementary material, we summarized all significant issues and reported bugs we discovered via LaCAST. The section also includes new issues that we discovered during the work on the journal publication. This journal version was published in the Aslib Journal of Information Management in 2019.

"Semantic preserving bijective mappings for expressions involving special functions between computer algebra systems and document preparation systems" by **André Greiner-Petter**, Howard S. Cohl, Moritz Schubotz, and Bela Gipp. **In:** *Aslib Journal of Information Management* 71(3): 415-439, 2019.

Appendix D — [13]

It turned out that LaCASt translations on semantic LaTeX were so stable that we can use the same approach for verifying translations also to specifically search for errors in the DLMF and issues in CAS. To maximize the number of supported DLMF formulae, we implemented additional heuristics to LaCASt, such as a logic to identify the end of a sum or to correctly interpret prime notations as derivatives. Additionally, we added support for translations to Mathematica and SymPy. We extended the support for Mathematica even further to perform the same verifications in Maple also in Mathematica. The Mathematica support finally allows us to identify computational differences in two major proprietary CAS. Moreover, we extended the previously introduced symbolic and numeric evaluation pipeline with more sophisticated variable extraction algorithms, more comprehensive numeric test values, resolved substitutions, and improved constraint-awareness. All discovered issues are summarized in Appendix D available in the electronic supplementary material. We further made all translations of the DLMF formulae publicly available, including the symbolic and numeric verification results. The results of this recent study have been published at the international conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS).

"Comparative Verification of the Digital Library of Mathematical Functions and Computer Algebra Systems" by **André Greiner-Petter**, Howard S. Cohl, Abdou Youssef, Moritz Schubotz, Avi Trost, Rajen Dey, Akiko Aizawa, and Bela Gipp. **In:** *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, 2022.

Chapter 5 — [8]

We also applied the same verification technique to the Wikipedia articles we mentioned earlier, which enabled LaCASt to symbolically and numerically verify even complex equations in Wikipedia articles. This evaluation is also part of the TPAMI submission.

Preprints of my publicationsare available at
https://pub.agp-research.com

My Google Scholar profile is available at
https://scholar.google.com/citations?user=Mq2B9ogAAAAJ

All translations of the DLMF formulae are available at
https://lacast.wmflabs.org

A prototype of LACAST for Wikipedia is available at
https://tpami.wmflabs.org