Computational linguistics as natural science

Revised manuscript, 2021

Pauli Brattico

Abstract. Theoretical linguistics uses an epistemological framework consisting of the modern experimental method combined with premodern ordinary language justification. Natural language syllogisms are not accepted for justification in the hard sciences, however. It is argued that they should not be accepted in theoretical linguistics either. Computational linguistics is proposed as a possible way for bridging the gap. The study of Romance clitics is used an example to illustrate the merits and challenges of the proposed methodology. Specifically, a Python based analysis of the clitic data is presented whose correctness is verified by deductive calculations performed by a computer.[1]

Keywords: computational linguistics; deductive reasoning; natural science; clitics

1 Natural science, deductive reasoning and theoretical linguistics

The 17th century natural philosophers who invented and then developed modern science took great care to follow the canons of mathematical rigor in their work. Galileo, after introducing the notion of uniformly accelerated motion makes Simplicio, one of the three protagonists in the *Two New Sciences*, to inquire "whether this acceleration is that which one meets in nature" (Galileo 1954 [1638]: 178) and then describes a set of experiments demonstrating that balls falling on an inclined plane have the properties deduced from his mathematical construct. Most of Galileo's work is dedicated to rigorous proofs. A century later Isaac Newton observed how his science "sets forth mathematical principles of natural philosophy" such that "the motions of the planets, the comets, the moon, and the sea are deduced

---

[1] An earlier version of this manuscript was submitted to *Language Sciences*, where it received one positive and one negative review. The editor invited a revised version. During the revision process the editorship of the journal changed hands, however, and the new editor rejected my resubmission (this version) virtually overnight. The stated reason for rejection was that the Newtonian scientific framework, proposed in this paper, was not considered to provide a "sufficiently secure foundation" for this journal's standards.

from these forces by propositions that are also mathematical" (Newton 1999 [1726]: 382).[2] Newton's work, like that of Galileo's, consists mostly of mathematical proofs.

The notion of mathematical rigor and formalization as such were not new, however. Both Galileo and Newton could take mathematical rigor for granted, in fact both were inspired by Euclid's axiomatic geometry.[3] The maxim that explanation follows observation was invented by Aristotle and was followed throughout the medieval period. Empirical facts were gathered into large encyclopedic tomes, while there were many attempts at coming to terms with basic physical phenomena such as movement by applying the Aristotelian-Archimedean tradition.[4] What eventually substituted the medieval Aristotelian natural philosophy was the way theory and observation were organized into the new epistemological system. While the pre-moderns collected observations and categorized and formalized them within aprioristic theoretical prisms, often synthesizing Aristotle and the theologians of their era, modern science uses controlled experimentation to establish facts and mathematical reasoning to demonstrate that the theory does capture them. It is against this background, then, that it is of some interest to note that while a substantial portion of modern linguistics works within the context of the established experimental hypothesis-testing framework, they seldom use rigorous, deductive demonstrations.[5] What justifies the neglect?

---

[2] This citation from the Author's preface for the English translation of Newton's *Principia* (3rd edition), translated by I. Bernard Cohen, Anne Whitman and Julia Budenz.

[3] See Cuomo (2001). Newton constructed *Principia Mathematica* by using geometrical proofs instead of the algebraic methods typical of modern calculus and perhaps did so for the admiration of the ancient geometry as reported by his assistant Henry Pemberton.

[4] Clagett (1961). I would like to quote here from Edward Grant: "It must not be thought that during the Middle Ages there was little interest in seeking knowledge of physical reality. On the contrary, since Aristotle himself was convinced that he had arrived at a system which represented physical reality, his many followers in the thirteenth century, most notable Thomas Aquinas, were also physical realists, much like Copernicus [...] Their physical realism was, for the most part, indistinguishable from their wholehearted acceptance of Aristotle's physics and cosmology" (Grant 1971: 88). The pre-modern relationship between mathematics, mathematical reasoning and nature was established in Aristotle's *Physics* and *Posterior Analytics* and was preserved in that form virtually intact until the 17th century science. See Bochner (1966).

[5] There exists a subgenre of linguistics that denies the applicability of the modern scientific method to language and linguistics (see, for example, Itkonen 1983). The methodological point I wish to establish in this article applies irrespectively of whether

In answering this puzzle, we must first reject the possibility that linguists were unaware of the fact that deduction and rigorous calculation play a role in science. Evidence is not hard to find. Chomsky, writing in the preface of *Syntactic Structures*, points out that his work seeks to "construct a formalized general theory of linguistic structure" because by "pushing a precise but inadequate formulation to an unacceptable conclusion, we can often expose the exact source of this inadequacy and, consequently, gain a deeper understanding of the linguistic data. More positively, a formalized theory may automatically provide solutions for many problems other than those for which it was explicitly designed" (Chomsky 1957:5). He furthermore pointed out, in agreement with Galileo and Newton, that "obscure and intuition-bound notions can neither lead to absurd conclusions nor provide new and correct ones, and hence they fail to be useful in two important respects" (p. 5). The method was followed fruitfully in the early years: thus, in a classic 1993 linguistics textbook Bartee et al. correctly observed that a "formal grammar [...] is essentially a deductive system of axioms and rules of inference which generates the sentences of a language as its theorems" (Partee et al. 1993: 435). Yet, actual rigorous demonstrations had all but disappeared from concrete linguistic work in Chomsky (1981). The stated grounds for the neglect was that the author was only interested in "leading ideas" without any regard for "explicit theory" or "specific realization." Emphasis on concrete realization was now claimed to be "misleading and perhaps even pointless" (p. 3) because it focuses attention to differences that are, it was claimed, empirically irrelevant. But why not to demonstrate that the "leading ideas" deduce the empirical facts? It is inconceivable that Newton would have rejected calculus on such grounds.

This criticism should perhaps be amended with a few remarks and clarifications before trying to find an answer. Although mathematical rigor plays a role in scientific discovery (see, e.g., Bangu 2012:110–44; Dyson 1964), its role in the scientific literature proper is justification. I will mostly ignore the role of discovery in this article and focus on justification, since the former can play a meaningful role only insofar as the latter is present. Furthermore, a collection of ambiguously formulated leading ideas could turn out to be true and revolutionary; and even if not reaching anything beyond ordinary they can still have substantial empirical and experimental support, leading into research avenues that would otherwise receive no attention. There is, in other words, no science without leading ideas. What is missing is a

---

language itself is regarded as a natural phenomenon or rather as a derivative of some type of socio-normative construct. Formal deductive methods are routinely used in non-empirical sciences, such as in logic, philosophy, mathematics, and computer science, to mention a few.

chain of rigorous reasoning that takes the reader from the hypothesized theoretical construct into empirical facts and which ultimately justifies, in the scientific sense of the term, the assertion that the leading ideas do describe and explain the facts. In short, my concern in this essay is justification, not discovery or truth.

A second point worth mentioning is that my concern has little relevance to the empirical and experimental part of linguistics or any science whose job it is to furnish us with observations and raw data; useful experiments and other data gathering missions can be conducted without mathematical frameworks or any systems of rigorous calculation. Much of modern linguistics could be seen as an attempt to come up with useful facts, although that would be both a misleading characterization of the actual practices in the field and too modest way of envisioning the role of linguistics. The authors in this field, as in any field, make sweeping theoretical claims. Indeed, they should. I also want to emphasize that my concern is not whether one uses formalization or presents a formal grammar of some type. Mathematical rigor was invented long before the 17th century scientific revolution took place. Plenty of formalized grammars exist today from which to choose.[6] My concern in this article is, instead, the use of formalization in empirical justification: how to calculate, i.e., to show with certainty, that the facts do follow from a theory. That was the point of formalization in Galileo’s and Newton’s work.

In addition to validating a theoretical idea, demand for rigor has several additional benefits worth mentioning before considering the situation in linguistics more closely. One benefit involves theory comparison. The use of mathematical rigor “forces clarity and precision […] enabling meaningful comparison between the consequences of basic assumptions and the empirical facts,” as observed by May (2004:791). The field of modern linguistics is divided into subdisciplines or paradigms all which begin with what looks to be fundamentally different conceptions of what language is and its place within other natural phenomena. Due to the lack of rigor the proponents of these approaches can pontificate freely on what they believe language to be and whether and to what extent any of these frameworks can or cannot explain some facts. All such arguments again depend on the reader’s willingness to fill in the blanks. This was true of most medieval theorizing before Galileo and Newton. But if fully rigorous demonstrations were required, these approaches could be compared without the speculative dimension. We could, instead, examine if a theory can be formulated in such a way that all the facts do, as a matter of fact, follow from it. It is, moreover,

---

[6] For a particular valuable formalization of a (group of) recent minimalist grammars, relevant also to the analysis of clitics discussed later in this article, see Collins and Stabler (2016).

extremely irrational to refuse to do this. Once some theoretical construct can demonstratively deduce the facts, we would be in a position to use mathematical elegance and empirical coverage, both quite rigorous concepts, in theory comparison.

An additional benefit is that rigorous demonstration can potentially eliminate subjective bias. It is no secret to anybody working in any scientific discipline that we are biased when it comes to the leading ideas of our own. Science is produced by humans whose behavior is based on a number of irrational and rational but nonscientific tendencies that can never be eliminated, many of which play a useful role in other ways and should not be treated with hostility (e.g., intellectual curiosity, willingness to speculate). Demand for rigorous demonstration can, however, mitigate some of these issues as it removes nonscientific gaps from the justification and replaces them with unbiased mechanical deduction. This creates a potentially beneficial process in which author’s own convictions are subjected to rigorous tests before they are presented to the larger scientific community. In the hard sciences, ideas are verified by calculation before they are presented to the scientific community.

One constant problem in abstract theorizing is the fact that almost any theoretical idea can be cast in a positive or even persuasive light by ignoring empirical coverage and/or by silently modifying theoretical constructs and assumptions elsewhere. A fully formal theory provides a reliable way to keep these impulses under control. It forces one to remove free parameters and intuitive gaps from the system. Thus, by formalizing the leading ideas within a system of (potentially less relevant) auxiliary conjectures one can examine how they work in connection with other possible components of the theory without merely asserting that they might work. I agree with Chomsky that many auxiliary conjectures could be less relevant content-wise, and subject to alternative formulations, but merely having them allows one at least to demonstrate that the leading ideas do not work only when many other issues are put aside or when they are combined with innumerable free parameters that nobody is controlling. We will see examples of this below.

Given all the possible benefits and an impressive proven track record of the natural scientific methodology, why linguists have not adopted it? Gingras (2001), in an interesting paper, provides two reasons why natural philosophers resisted at first the mathematical methods of Newton. One reason, he argues, had to do with sociology of science reasons and stemmed from the fact that the use of mathematics excluded interested participants from the field, hence those participants quite predicably reacted by rejecting all mathematization as irrelevant or misleading. Another reason was that Newton’s deductive system changed the criteria of what passes for an explanation in science. For example, Newton’s theory of celestial mechanics required that there exists an ‘action over distance’ in a vacuum (the gravitational force field), as this was required in the mathematical deductions; it was not required because it agreed with common sense. In fact, the notion that there could exist an action over a distance was ridiculed as mysticism.[7] Perhaps the factors mentioned by Gingras do play a role in linguistics today but, even if they do, I do not think that they can be said to constitute a justification.

Let me, finally, consider one possible answer to this puzzle that appears to have at least some degree of legitimacy. Linguistic data is inherently combinatorial in nature. Sentences are put together by combining words without any upper bound on their complexity, and thus the theories put forward to account such phenomena tend to posit combinatorial operations that can become extremely laborious to perform by hand. This problem remains even when one considers grammars and theoretical constructs that do not involve unbounded recursion: symbolic computations are cumbersome and by their nature unilluminating if compared to, say, solving standard differential equations or crafting mathematical proofs. Due to the combinatorial nature of the phenomena the amount of calculations required to handle the phenomena deductively – calculating rigorously from the theory into observations – increases exponentially as a function of the size and complexity of the phenomenon. The required deductive chains become long, cumbersome, and prone to errors; so long, in fact, that it is inconceivable that they be performed by the paper-and-pencil method in any *practical* research project. Just to consider an example, checking that a particular formal theory, discussed later in this article, correctly predicts a small set of 1361 datapoints (test sentences) required that we perform 252.244 computational steps.[8] Not only are the calculations labor-insensitive to do, but also potentially unilluminating in the sense that most of them consists of nothing but a huge number of mechanical and repetitive symbol manipulation. What I would like to argue in this article, however, is that modern *computational linguistics* and the tools developed within this discipline provide a feasible solution to this problem and thus arguably removes the last barrier between theoretical linguistics and modern science.

---

[7] This matter was addressed in the General Scholium of the third edition of the *Principia*, where Newton wrote that he did not know the ultimate cause of the gravitational force, hence it remained a mystery, but was certain that it existed as it was "sufficient to explain all the motions of the heavenly bodies" (Newton 1999 [1727]: 943). In this context "explanation" meant a process in which empirical observations, such as celestial mechanics, are deduced from mathematical assumptions and postulates.

[8] These numbers come from a theory that had been optimized with respect to the number of computational operations required for verification. When the same theory was tested without optimization, this number was close to ten million.

The rest of the article is structured in the following way. Section 2 motivates and further defines the proposal, with the explicit aim of addressing the feasibility of the proposed methodological principle. I will begin by elucidating in some more abstract sense what it could mean to use computational linguistics methodologies for the purposes of theory justification in scientific linguistics, and then move on to address the concrete details, such as implementation and execution. The discussion in this section presents an overall framework that will be then applied to the study of Romance clitics in Section 3, where a fully formal and testable analysis of the data is developed. The purpose of this discussion is to demonstrate that the ideas are feasible in the context of a practical research project. Then, in Section 4, I will go through the steps required to deductively test the correctness of the proposed analysis, the main point being again not so much the analysis itself but the practical and methodological aspects of the framework. I will discuss both pros and cons of the computational methodology.

2 Computational linguistics as a tool for linguistic calculation

I would like to argue that modern computational linguistics provides the means to perform the required calculations in a feasible way, and that we could use these methods to support concrete research work. The use of computers and computer simulation in the hard sciences is by no means a new idea. As pointed out by Stephen Wolfram, a “new paradigm” was born in which “scientific laws give algorithms, or procedures, for determining how systems behave. The computer program is a medium in which the algorithms can be expressed and applied [...] It thereby allows the consequences of the laws to be deduced” (Wolfram 1984: 204). That was forty years ago. The situation has only improved since.

Let us first examine how we could approach the problem of deductive justification of a linguistic theory or hypothesis by using a computer, thus allowing “the consequences of the [linguistic] laws to be deduced.” A linguistic theory or hypothesis is typically constructed in the form of a *grammar* that articulates a set of rules or patterns attested in some fragment of linguistic behavior. The grammar could be intuitive and descriptive or fully formal and generative; insofar as we are concerned with empirical and not normative linguistics their purpose is always to describe the combinatorial rules attested in some relevant fragment of real natural language use. It does not matter for the purposes of the present argument if the grammar produces concrete expressions, morphological forms, semantically interpretable structures, pairs of expressions and meanings, structural descriptions, complete derivations or cognitive structures and/or communicative intentions, as they all capture attested linguistic patterns. It is also immaterial to the argument at hand if the theory is based on an innate grammar (UG) or posits elaborate and complex learning mechanisms which, together with a learning history, produce the observed patterns the theory is designed to capture. To construct a rigorous demonstration that a hypothetical grammar captures an empirical phenomenon, we can use one of the two computational approaches:

1. We let the grammar generate all expressions available from a given set of atomic elements and verify that the output contains the attested observations and furthermore contains nothing else. The testing procedure consists of a recursion that examines all combinatorial solutions incorporated into the theory, given a lexical base. If the grammar is formal, adding the required recursion and implementing it on a computer can be characterized as a trivial task. If the grammar has a learning component, then verification contains an additional step in which the learning component selects a model from a pre-defined class of models on the basis of its learning history and then reproduces the data we are attempting to capture;
2. we let the theory to decide for any input whether it is accepted or rejected, i.e. belongs to the attested set of expressions. If the model incorporates learning, then the demonstration has an additional step in which the recognition principles are parametrized based on external data.

The first method could be called enumeration, because it works by literally enumerating the consequences of the theory and then by comparing the output with observation. It corresponds to a standard proof technique in the natural sciences, in which the axioms of the theory are employed to calculate empirical consequences as theorems. The term "generative" in generative grammar means enumeration in this sense (and, furthermore, means nothing else). The latter strategy (2) could be called recognition, because it looks at the data and decides if the theory predicts its existence. These methods are mathematically equivalent because they define – "explain" in the sense of Newton's science – a set of observations. Given this background it is easy to see how computational complexity could play a prohibitive role. The amount of computations required to produce and check all derivations from a set of lexical items is astronomical even on a modestly complex input, while a recognition procedure requires all possible input sequences to be judged and, moreover, often by an algorithm that rejects an input only after an exhaustive search of possible parses. It is not sufficient that the researcher performs these computations once; they must be performed potentially after each change in the theory, no matter how small.

To see the enormity of the problem, let us consider some of the complexity properties of the linguistic algorithm that will be discussed later in this article. To verify that one grammatical sentence with one center-embedded relative clause belongs to the set of well-defined sentences requires the algorithm to perform approximately two hundred linguistic operations, while a grammatical sentence with five embeddings has increased the demand to close to one hundred thousand, taking two minutes from a fast desktop computer. These numbers are associated with one sentence; in a typical linguistic study the number of relevant data points is going to be in the range of thousands or more. Suppose, for example, that we were interested in developing a linguistic theory (grammar) of relative clause constructions. To be completely rigorous, any such model would have to be tested with both grammatical and ungrammatical relative clause constructions. How many relevant ungrammatical relative clauses are there that we must test? If we assume that the set of ungrammatical relative clauses to be tested is constructed by generating all possible word order permutations from the set of grammatical relative clauses, the amount of test sentences is going to be more than a million. Many of these sentences will take thousands of operations to verify; furthermore, a verification procedure that demonstrates that an ungrammatical sentence is ungrammatical will typically require the algorithm to consider all possible parses of the input sentence. In addition to these rough estimates, I tested the algorithm also with a realistic test corpus that contained 1361 regular, canonical test sentences in English, Finnish and Italian, and counted the number of linguistically meaning operations performed by the algorithm when it checked if a linguistic theory predicted that dataset correctly. The number of operations that had to performed turned out to be 252.244, which is a large number if we were required to perform it several times and without making any mistakes along the way, let alone if performed by hand.

There are several mitigating issues, however. One is that the cost of computational resources has fallen to a point at which modestly complex tasks have become feasible, especially in the light of the fact that we can exploit parallel processing in this task domain. The amount of processing power available today for only a moderate cost is substantial and increasing. Executing the 252.244 linguistic operations (in addition to millions of auxiliary operations) took two minutes from a normal PC computer using one processing threat, no optimization, and relatively slow Python based implementation. By using code optimization this number can be reduced further into a fraction. Another and perhaps more significant issue is that in practical work it is often not necessary to work with overly complex expressions and/or extremely large test corpora, as research work tends to focus on narrowly defined hypotheses. While constructing the theory, we might work with only a small fragment, and only test the whole dataset after a reasonably correct initial guess is available, perhaps even accepting a few problems here and there as long as they are reported as such. If the test set is necessarily very large, we could use smaller samples and/or dissect the corpus into several partitions and utilize parallel computation. In addition, exhaustive enumeration or recognition provides only an ideal against which the less optimal but feasible approaches should be measured. I will discuss some of these practical aspects further below.

The entry barriers have also come down. Expensive special-purpose hardware requirements are no longer an issue. A standard laptop can perform the computations required for a focused empirical study. Another and perhaps more important reason is the emergence of abstract programming languages, existing software libraries and development tools that have for the most part eliminated all machine-specific coding practices from the picture. It is possible to write programs that has very little linguistically irrelevant boiler plate code and differs from actual linguistic formalization only to a modest degree, if at all. The situation will only improve in the future as the tools improve. This means that the gap between writing a formal linguistic theory on a paper and writing the same theory on a computer has become so small that it can no longer be considered an obstacle.

Before we look at the issue more concretely, I want to establish one additional point. Rigorous demonstration constitutes a requirement that should be added to linguistic theorizing rather than substitute something that is neglected. The usual criteria of observational, descriptive, and explanatory adequacy (Chomsky 1965) provide a minimal framework for cognitive plausibility. The algorithm must reproduce the attested observations and only those (observational adequacy), provide them with structural descriptions or derivations that are cognitively plausible according to some explicit or implicit framework (descriptive adequacy), and the theory must be in agreement with what can be regarded as empirically defensible in the light of constraints on language acquisition (explanatory adequacy). Descriptive adequacy and explanatory adequacy are theory-internal criteria that depend on the theoretical framework but given any such framework they will always play a role in a theory of (human) language.[9] Observational adequacy, the ability of the hypothesis to cover some data, is a theory-external requirement in any scientific theory, in any discipline. It is still a strong requirement. For example, it rules out most practical parsing algorithms available today, as they are not required to detect ungrammaticality and thus fail already the most elementary condition of observational adequacy. In addition, the point is not to substitute scientific theorizing with brute data mining or blind big data approaches which tackle any problem by increasing the number of free parameters and hence have very little to say about any specific subject matter. A theory or algorithm that applies equally well to

---

[9] I am not advocating any particular generative theory in this article; the criteria for cognitive plausibility set forth by Chomsky (1965) apply to any serious scientific linguistic theory.

astronomical, economic and linguistic data is not automatically any better than Newton’s laws, Phillips’ curve or Greenberg’s linguistic universals. Robert May, who was concerned with the use of computational simulations in biology, warned against drawing “sweeping conclusions [...] from the alleged working of a mathematical model, without clear understanding of what is actually going on” (May 2004, p. 792). That warning remains pertinent. But my point is that we should add the deductive structure to the existing best practices in the field instead of using them to greenlight the elimination of such practices.

Methodological arguments carry little weight unless the proposed methodology is applied to a concrete empirical phenomenon. I will illustrate these issues by considering a concrete linguistic phenomenon and then approach the topic by using computer as a calculation tool. All empirical justification reported in this study will be based on a deductive work performed by a computer program, so we can take a closer look at how the system works and what possible challenges we face. Romance clitics, in particular clitics in the standard Italian, were selected as an empirical phenomenon. Various problems of the methodology are then considered, with a particular focus on practical feasibility. I will argue that linguistic theorizing in this arena benefits from deductive demonstration and, vice versa, rejecting such methods no longer has rational justification and should not be resisted irrationally. I will also point out that the methodology does not come unchallenged, and that there are constraints on its use.

3 Italian clitics and computational linguistics

Let us consider how we might approach the problem of clitics (or any other linguistic phenomenon) from a computational point of view, keeping the guidelines discussed earlier in mind. We could approach this problem either by using the enumerative strategy or by the recognition strategy (strategies 1-2 discussed in the previous section). Suppose we select the enumerative methodology. We would formalize a productive linguistic theory, say one consisting of a system of primitive lexical elements together with the rules that combine them, and then add trivial recursion deploying the axioms and the lexicon to produce an output set. The size of the output set can be controlled by selecting the initial lexical set (also called the “numeration”) from which all derivations must begin. The resulting set is compared with attested observations. The theory is changed if and when it over- and undergeneralizes. This procedure is used in most linguistic studies published today, although the demonstration is practically never rigorous, only a sketch. A recognition method, in contrast, works by reading grammatical and ungrammatical (attested and unattested) inputs and decides for each such input whether it should be accepted or not. The results are again compared with observation, in a straightforward and unproblematic way. While the enumerative strategy generates a set of observations, the recognition algorithm defines a characteristic function for that set. As I have already pointed out, the two methods are mathematically equivalent: they define (potentially the same) the set of observations.

Both methods have pros and cons. One problem with the enumerative method is that it produces a rather voluminous amount of data: each run, unless limited in some artificial way, will produce all possible derivations from the given lexical set. The data is relatively unintuitive to handle. The recognition method, on the other hand, handles the data in a more intuitive way because it allows one to construct an explicit test corpus and only compute and test selected inputs, even just single sentences in isolation. In addition, the procedure is potentially very fast and convenient to use within the context of a real research project. The tradeoff is that it requires a parser, a component that uses information available in the input (words, their properties, word order, prosody) to arrive at a set of syntactic and semantic analyses. The parser component has many performance related properties that are irrelevant to grammatical theorizing and therefore involves extra work and specifications that are not necessarily the main focus in the study itself.

Regardless of the chosen approach, harvesting computational power for linguistic calculation requires that we *translate our leading ideas into a machine-readable formalization*. This step is made easier by the fact that in many cases we do not need to develop everything from scratch, as many suitable tools already exist. The choice depends on theoretical preferences. I use the linear phase parser toolkit (Brattico 2019, 2020, 2021; Brattico & Chesi 2020) as a starting point in this article.[10] The parser toolkit was written in the Python programming language. It works by decomposing each input sentence into morphemes and inflectional features, which then guides the application of the

---

[10] The linear phase algorithm will not be the focus here and therefore not examined in detail; I will use it as an example of one possible direction. The theory is nevertheless based on the following assumptions. It assumes, following Phillips (1996), that Merge can operate incrementally by consuming the linguistic sensory input in a left-to-right order. It uses an inverse Merge operation (call it Merge-1) recursively, by using backtracking if necessarily, to create a set of phrase structure objects that represent the possible surface parses of the input set; if the set is empty, the input is judged ungrammatical. This assumption is shared by the Dynamic Syntax approach (Cann et al. 2005). Because the input is read from left to right, Merge-1 must be able to operate in a top-down order, a fact that has been emphasized in the top-down grammars (e.g., Chesi 2004, 2012; see Merchant 2019, Chapter 5, for a useful recent discussion and references therein). The linear phase algorithm uses the top-down mechanism but is not limited to it. Inverse transformations (inverse chains) map surface parses into syntax-semantic interface objects. Methodologically the approach follows Lewis et al. (2005), who propose to work with fully computational language comprehension models. The author maintains the computational implementation at https://github.com/pajubrat/parser-grammar.

computational-cognitive operations of the theory such as Merge, Move and Agree to generate a set of derivations. The resulting parses are filtered by grammatical and lexical constraints. The operation of the algorithm is illustrated schematically in Figure 1.

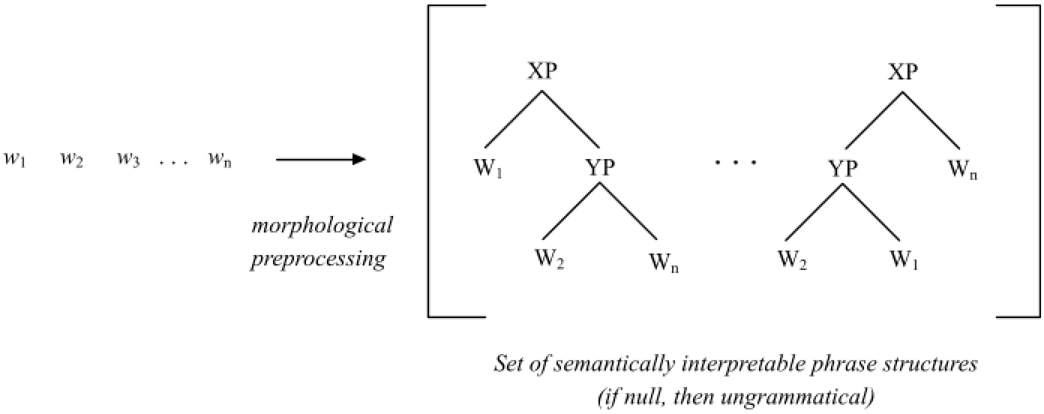
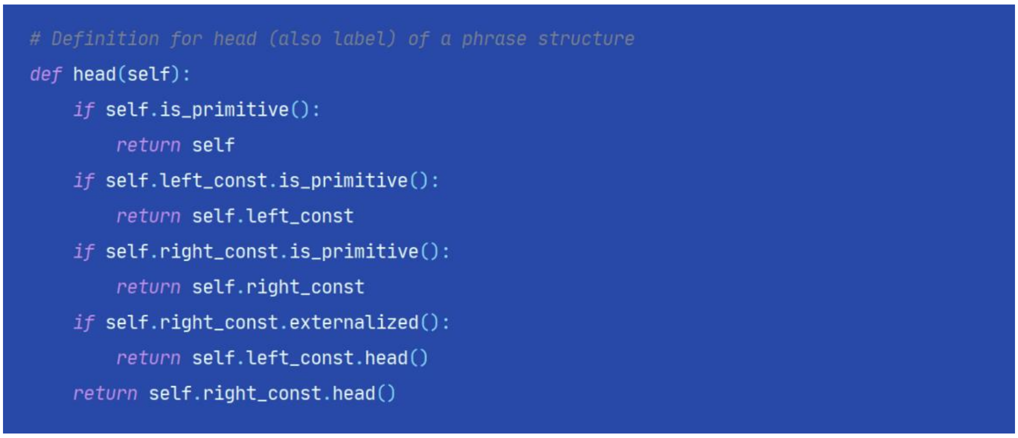


Figure 1. The general structure of the LP language comprehension algorithm. Each input sentence is fed to the algorithm as a linear string of phonological words w_1, w_2, w_3, . . ., w_n. The string is preprocessed, after which the computational operations of the theory (e.g., Merge, Move and Agree) are applied to derive phrase structure representations.

Once the general computational framework is in place, the next step is to translate our theoretical ideas into the chosen framework. This consists of specifying the theory formally and then implementing that formalization in some machine-readable framework. Both steps are almost trivial: the first, because formalization belongs to the standard toolkit of theoretical linguistics, and the latter because today several special-purpose and even general-purpose programming languages exist that are so abstract and modular that the two formalizations need not to depart from each other all that much. What were ten or twenty years ago implemented by means of complex machine-specific class-inheritance systems involving detailed variable declarations and type checks, thus coding practices that were concerned with the computer rather than with the subject matter, are today expressed with readable, simple and intuitive Python code that to my eye does not depart much from standard linguistic theorizing. To illustrate this point in a concrete way, Figure 3 shows the Python formalization for the labelling algorithm used in the linear phase parser. The code can be easily understood by anybody who has worked with labeling.



```python
# Definition for head (also label) of a phrase structure
def head(self):
    if self.is_primitive():
        return self
    if self.left_const.is_primitive():
        return self.left_const
    if self.right_const.is_primitive():
        return self.right_const
    if self.right_const.externalized():
        return self.left_const.head()
    return self.right_const.head()
```

Figure 3. A screenshot of the Python formalization for a simple labelling algorithm. The algorithm defines the notion of "head" for an arbitrary bare phrase structure α, as follows (reading from top to bottom): if α is primitive, it will be the head; else if α has a primitive left constituent, it will be the head; else if α has a primitive right constituent, it will be the head; else if α has a right constituent that is an adjunct, we apply the algorithm recursively to the left constituent; else we apply it recursively to the right constituent.

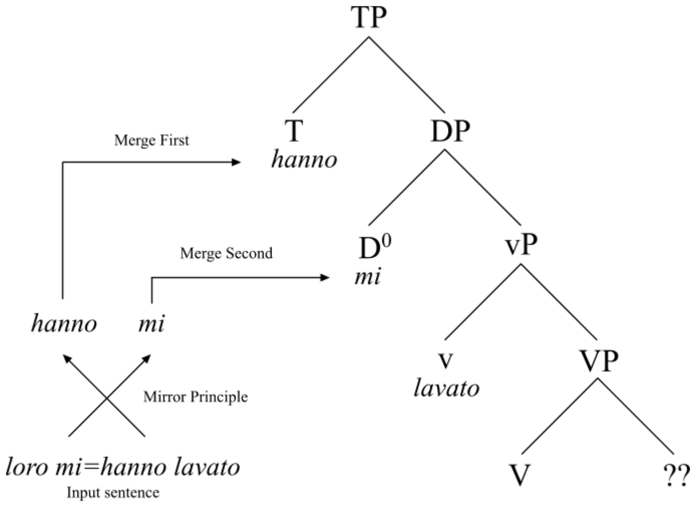
My point is not that the above labelling algorithm is correct but that the above formalization is not much more complex than one written on paper (see the English description in the figure legend) and, even if it were slightly more complex, we have the benefit that it calculates a label for any constituent fast and never making any errors. In the end, all components of the linguistic theory must be translated into machine-readable format, Python or some other language, after which the system can perform the required deductions.

Before looking at the actual deductive work, I want to address one possible objection. In the work I cited earlier (Chomsky 1981), the author defended the position that this step would involve too many technical details to distract from the development of the author's key concerns, his or her "leading ideas." In addition, some of the present author's colleagues have objected that the approach is not or could not be "feasible." The requirement that a computational theory processes concrete linguistic data presupposes that it handles several things that are not necessarily the ones the researcher is primarily interested in. To begin with, the researcher must acquire some level of competence in some programming language and computer usage that goes beyond Excel tabulation and maintaining email correspondence. Although these problems can be alleviated to some extend by imposing artificial restrictions and abstractions, the level of detail presupposed in any machine-readable implementation is greater than it is in any theoretical work operating with natural language syllogisms and pseudo-formalisms. That being said, almost every time I had to solve an issue I felt to be tangential or otherwise a distraction from the main focus, or acquire a programming technique that I wasn’t familiar with, there was also a sense in which not clearing that issue would have meant that the core ideas were after all not yet in an enough crystallized form. For it was only when the model produced *wrong* outputs that many of these problems were recognized in the first place; they did not occur out of the blue. Thus, what was at first assumed to be a tangential issue was in reality something, often something dubious or outright wrong, that the theory presupposed implicitly. This concern, then, presents a dilemma for those who consider that the burden of computational implementation is too much: if it were impossible or unfeasible to formulate one’s theory in this way, wouldn’t this imply that there is also something wrong with the theory? In other words, what possible rationale could there be for a *true* theory to be such that it could not be completely rigorous and unambiguous?

Let us now turn to the actual verification procedure. To understand how the verification protocol works, we first examine what the linear phase parser does “out of the box” when presented with standard Italian object clitic constructions such as *loro mi=hanno lavato* ‘they me-have washed’ that I will use as an example in this article. The nontrivial property of a clitic sentence such as this is the fact that the direct object ‘me’ of the verb ‘wash’ is prefixed to the auxiliary (*mi=hanno*) instead of occurring at the canonical postverbal position (as in *loro hanno lavato me* ‘they have washed me’). This option has not been grammaticalized in English, for example, and the linear phase model has never been tested with such sentences. This type of clitic construction provide a useful specimen for the present purposes as the linear phase model has never been tested with them before.

Because the order of morphemes inside phonological words tends to mirror their ordering in hierarchical structure (Baker 1985; Julien 2002), the linear phase algorithm performs morpheme reversal inside each word and thus maps an input word such as *mi=hanno* ‘me-have’ into *hanno* + *mi*, which in turn means that the clitic arrives to the syntactic combinatorial system *after* the auxiliary, not before. The system implements a type of “clitic climbing” inside the phonological branch. The problem, however, is that because the clitic occurs as part of another word, the algorithm interprets it as a *morpheme* and generates [*loro* [*hanno*^{0} [*mi*^{0} [*lavato*]]]) which it correctly rejects due to the missing complement of *lavato* ‘wash’. The semantic component is unable to understand who was washed. The process is illustrated in (1). This describes the (wrong) output produced by the algorithm if nothing was changed.

(1)



The algorithm interprets the finite clause ‘they have washed me’ as a ‘have + DP’ structure (perhaps something like ‘they have my=washing’) which, no matter what underlying linguistic theory we begin with, would be considered linguistically implausible and unintuitive. The existing framework does not, therefore, understand clitics. But since we can test the model with concrete clitic sentences there is a sense in which we have also *proved* that the axioms of the theory, as they have been encapsulated into the Python based implementation, are not sufficient to deduce the empirical facts. The algorithm produces the exact same output every time it is presented with this input sentence and does this by performing the exact same series of computational operations, all which are in turn determined and fully defined by the axioms of the theory and also visible in a derivational log files that the algorithm writes run-time when it processes the input. There is *no point* at which a human observer could readjust the operation of the model against observation by refurnishing it with natural language corrections or speculative conjectures.

One possible solution to the abovementioned problem is to map the clitic into to a phrasal unit instead of a head D^0. This change was trivial to implement by changing an entry in the lexicon.[11] After this minor correction, the algorithm generated [*loro* [*hanno* [CL$_1$ *lavato* __$_1$]]] and accepted the sentence as grammatical with the correct meaning ‘they have washed me’, in which the clitic pronoun was interpreted correctly as the patient of washing. Notice that once the clitic was represented as a phrasal pronoun, the model reconstructed it into the thematic object position. Moreover, the

---

[11] The triviality of this matter depends on the structure of the lexicon. If, on the other hand, a system of lexical representation were assumed that made this assumption impossible, then the same effect could be achieved elsewhere in the theory, for example, in the lexicon-syntax mapping.

representation just cited is not linguistically entirely implausible, although there are, of course, many theories of what the correct linguistic analysis should be. It is easy to see that the solution is still insufficient. One problem is that the system extracts clitic morphemes from any word, leading to unlimited cliticization. The algorithm will accept sentences such as **mi=loro hanno lavato* ‘me-they have washed’. But let us put this issue aside and pause to look into another, arguably a more fundamental matter.

We assumed in the above analysis that the phonological branch understands that *mi* is part of *hanno* and not *loro* (i.e. *loro mi=hanno* instead of *loro=mi hanno*). What was left unsaid is what is there in the sensory input that tells the algorithm this. Should the algorithm consider an alternative reading in which the sentence is analyzed as *loro=mi hanno lavato*? Notice that by reversing the order of morphemes inside the morphological component a completely different analysis would result, in which the object clitic appears in a position higher that the subject *loro*. Should it consider also this parse? Does this matter? This problem is related to another issue that was overlooked: in which component of the grammar are clitic boundaries recognized and represented? In phonology, lexicon, morphology or in syntax? What is a clitic boundary? The algorithm, as described above, provides no answers.

We further assumed that when morphemes are fed into syntax, syntax has access to some type of information determining which of them belong inside which words. How is that access implemented? Does syntax see surface strings as they are processed in the lower-level sensory systems? Do morphemes carry features to the syntactic component indicating that they are part of some word? If so, which word? The word to the left or to the right? Are left and right at the phonological level available inside the syntactic module? The problem is that by looking at the surface strings we tend to take such issues for granted, namely which elements are inside which words and what is “right” or “left,”[12] yet when the same process is looked from the point of view of explicit computation these facts must be made completely explicit, thus formalized, in some way. Exactly how this should be done quickly becomes very nontrivial.

We also assumed in the above natural language explanation that when a clitic decomposes, it no longer belongs inside any other word but exists as an independent pronoun. How does the algorithm (or the language faculty in the human brain) know this and how was this implemented? We assumed in the exposition earlier that morphemes generate heads

---

[12] Of course there is no “right” or “left” in spoken language. This makes the problem discussed in the main text even more challenging, however, since it is unclear what type of asymmetric relations we should ultimately use.

(*hanno* = T_0 and *lavato* = V^0), but how exactly and in what component and by what operation are complex heads decomposed into separate heads in syntax? No computational algorithm knows how to do any of these operations unless we explicitly tell it how they are done. We also assumed in our initial hypothesis that *mi* is mapped into D + N (determiner plus noun configuration), yet the surface vocabulary contains individual entries also for D (e.g., *uno, il*) and N (*macchina*, *casa*). Does the clitic therefore iterate through independent entries D and N in the lexicon? Or is it composed out of 'clitic-specific' D and N elements? Does the surface lexicon have *recursive* structure, in which one lexical entry can point to other entries in the same lexicon, and so on without any upper limit? Are circular pointers possible?

We also evaded the question of how, exactly, is morphological decomposition and morpheme reversal implemented inside phonology/morphology. What type of representations and operations are assumed? Are the representations based on lists, ordered lists, sets, or some other higher-order structures such as lists of lists? How are these structures reversed? The clitic *mi* has an accusative case, but we nowhere said where this fact is represented and how do we prevent the generation of an accusative head Acc^0. We again just assumed, as if it were self-evident, that while morphemes are mapped into heads, inflectional morphemes are processed differently: there was simply no accusative head in the phrase structure, although its generation was nowhere prevented in our English based description of the hypothesis. If the accusative is a feature, is it a feature of D or N? Or is it a feature of *mi*? If it is part of *mi*, how does it end up to the decomposed elements D and N? Which one? How? Is it reversed, too? Does the accusative case play *any* role in the derivation or is it a cosmetic diacritic that the speaker attaches to some words to make them (in the lack of better term) prettier? Perhaps most important of all, we assumed that *mi* is decomposed into D + N = DP but did not explicitly prevent the generation of [*loro* [*hanno*^{0} [D^0 [N^0 *lavato*]]]], an implausible but theoretically possible parse. How was this option prevented? Was it generated and then ruled out? Should it be ruled out if it were generated? Are there situations in with pronouns do, as a matter of empirical fact, decompose like this? Furthermore, the above list is only the beginning. Each item in the list opens up a further can of worms. How are word boundaries represented? How does the clitic boundary "=" differ from a regular word boundary? What exactly happens if a word is followed by *two* adjacent inflectional morphemes? What exactly happens, according to the analysis provided in English, if we have a sequence of clitics?

And so on. My point is not that every analysis must answer all such questions, or that any analysis must take a stance on all such issues. I am also not trying to emphasize the nontrivial nature of the data. Romance clitics constitute a controversial and difficult linguistic subject matter to deal with. Most of these issues are, in addition, irrelevant to the issue at hand, perhaps minor technical problems at best. This was, after all, what prompted Chomsky (1981) to evade rigorous formalization; indeed, during the present authors' first encounters with this specimen I was at first focusing only on the "leading ideas" and could not anticipate many of the problems cited above. Perhaps some of these issues can be judged irrelevant. I take this objection for granted; my point here is different. It is that until we move from natural language syllogisms that rely on human intuition into deductive reasoning we have *no way of knowing how many similar intuitive assumptions, problems and unsolved issues remain in our analysis*. Thus, it is not unusual that an intuitive "leading idea" that works on paper encounters a number of unforeseen problems once a rigorous treatment is attempted. Too many consequences of our assumptions remained unnoticed; too few assumptions were made explicit. The only way to solve these issues, I think, is to create a deductive structure that catches all such issues automatically and without any human intervention.

At this point I bifurcate the argument into two separate tracks, each discussed in its own section. The first issue is how to solve the empirical problems mentioned above (and others like it). I construct a formal analysis of clitics that has at least some hope of working with concrete data. This matter will be attended to in Section 4. This section contains a Python based formalized linguistic theory of Romance clitics that has at least some chance of working. The discussion will be tied to the chosen framework, in which I approach the problem from a "parsing-friendly" performance perspective, but the correctness of that particular analysis is not the point. I will also not refer to the underlying Python implementation but add a layer of abstraction and use standard linguistic terminology; the actual implementation code is available online. This should make the exposition more readable for linguists (although unfortunately less penetrable for computer scientists and programmers). The second and more important issue concerns the actual computational justification of the analysis, which will be addressed in Section 5. In it, I describe how the deductive calculations were used in verifying the analysis.

4 Linguistic perspective

The working hypothesis that clitics are reversed like ordinary morphemes in the phonological branch but are treated like ordinary pronouns in syntax left many problems unsolved that we will have to tackled before testing the analysis. This is because in order to examine the use of computational methodology in connection within a realistic research context we need a formal analysis that has at least some chance of working with a nontrivial dataset. The analysis presented here was developed for this purpose and has not been published elsewhere. Notice that the correctness of this particular analysis is not in the focus here, although I do think that it represents a possible empirical starting point that should not be implausible to the point of being trivially incorrect.

The linear phase algorithm, used as a background tool in constructing this analysis, receives a linear string of phonological words as input. A *lexico-morphological component* first decomposes complex words into their constituent morphemes and retrieves the corresponding primitive lexical items from an external data repository called the *surface vocabulary*. Let us assume that the whole processing pipeline all the way from reading the concrete sensory input into feeding the retrieved lexical items into syntax is called the phonological component or PHON for short. This architecture presupposes that there exists a *morphology-syntax interface* receiving lexical items as they arrive from PHON into syntax. Although the existence of the morphology-syntax interface (abbreviated as MS-interface from now on) is necessary in any theory, as the original phonological words existing in the form of concrete sensory objects must be analyzed at some level before they can be exposed to syntactic and semantic processing,[13] its properties are controversial. Let us assume that polymorphemic words are represented as complex heads at this level. A single phonological word such as *washed* will be represented as a complex head (T, v, V)^0 containing the verbal root (V), causativization or valency (v) and tense information (T), so that all constituent particles will remain inside the highest head (T, v, V)^0. We are positing these assumptions tentatively at this point and essentially without empirical justification, since they will be tested later by the simulation.

The key assumption we now make with regards to the clitic construction specifically is that clitics are special in the sense that they are detached from their phonological hosts *before* the MS-interface by an operation that corresponds to an inverse of the m-merger proposed by Matushansky (2006). We furthermore stipulate, following Matushansky, that the operation is constrained by an adjacency condition restricting the operation to adjacent items at the MS-interface. The following is a reasonably accurate nontechnical formulation of this idea:

(2) *Inverse m-merger*

Clitic morphemes, unlike ordinary morphemes, are detached (literally ‘un-m-merged’) from their hosts inside PHON.

---

[13] We have to solve ambiguities, perform morphological decomposition and analysis, retrieve correct lexical features, separate derivational suffixes from inflection, among other things.

To illustrate, consider an Italian sentence *loro mi=lavano* ‘they me-wash’. Principle (2) presupposes that *mi* ‘me’ is part of a complex phonological word *mi=lavano*. The lexical-morphological component matches the input with the morphological decomposition ‘*mi*=V#v#T’, which is reversed into ‘T#v#V=*mi*’ in accordance with the mirror principle. All this happens inside PHON. The MS-structure is created by matching individual constituents in the input with corresponding lexical items and merging them into the structure, in that order. Hypothesis (2) considers that the processing of ordinary morphemes and clitics now diverges: regular morphemes (e.g., V, v, T) are merged into complex heads $(T, v, V)^0$ while clitic objects are detached and treated as independent (weak) pronouns. The input ‘*mi*=V#v#T’ is, therefore, mapped into (3).

(3) [*loro* [$(T, v, V)^0$ *mi*]] (MS-structure, with the clitic detached by un-m-merger)
they wash me

Because clitics are detached into a specifier position of a lower verb, or directly into the correct object position, the problem of motivating the extra A-movement operation bringing the clitic to the preverbal position, discussed at length by Roberts (2010, ch. 3.1), is no longer an issue. The operation does not exist. “Clitic climbing” takes place in PHON in accordance with Chomsky et al. (2019). We also assume, following a rather long research tradition going back to Kayne (1975), that the detached clitic is a phrasal pronoun and is therefore decomposed into a D-part and N-part. It occurs at the MS-interface as a complex head $(D, N)^0$ and forms a trivial chain $[_{DP}(D,\_{}_1)\ N_1]$.[14] The whole architecture is summarized in Figure 1.

---

[14] The clitic represents a regular complex phrase, not a “maximal and minimal constituent” as assumed in some studies (e.g., Matushansky 2006; Muysken 1982; Roberts 2010). This issue is too complex to be dealt with here in a satisfactory way.

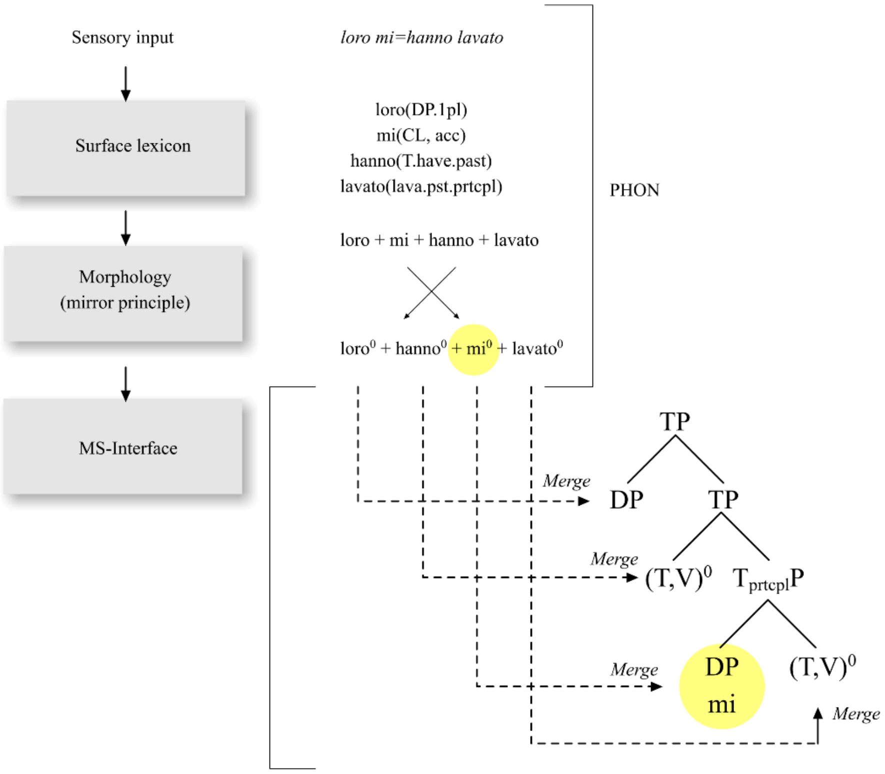


Figure 1. A nontechnical illustration of the assumptions underlying the proposed analysis. The processing begins with a linear string of phonological words that are matched with entries in a surface lexicon and decomposed morphologically if necessary. Word-internal morpheme order is reversed by the mirror principle. Clitics, unlike regular morphemes, are detached from their host words before the construction of the MS-interface object. All subsequent syntactic operations treat clitics like regular pronouns.

The system overgeneralizes because it allows the clitic to incorporate with any host. In the best case all illicit input configurations are ruled out by independent properties of the UG (e.g., ECP, CED).[15] Indeed, such principles suffice to rule out several ungrammatical clitic constructions such as nonlocal clitic climbing or clitic extraction from subjects or adjuncts, as this type of island constraints were already part of the existing linear phase algorithm. Yet, there are languages in which the Romance type argument clitics are all ungrammatical, and much variation even within the Romance language group. For example, clitic climbing to a finite verb in a T + V configuration is limited in modern French while more general in Italian (Kayne 1989). While standard Italian is more liberal than French, there is much

---

[15] We follow Baker (1988) and assume that our goal will be reduction of observed facts into independently justified syntactic principles (e.g ECP on chain formation) instead of positing construction-specific rules.

dialectal and between-speaker variation even within Italian.[16] Universal principles are unable to capture such variation. I suggest that the variation is *lexical*. I propose that m-merger and its inverse operation are licensed by a lexical-morphological feature. The feature licenses the type of elements clitics may left or right incorporate at the MS-interface, following Matushansky's adjacency condition. Left incorporation feature 'LEFT:α,β' licenses incorporation to the closest word with features α, β to the left, right incorporation feature 'RIGHT:α, β' licenses incorporation to right, in the manner illustrated by the example (4).

(4) $(...\alpha, \beta...)^0$ CL $(...\alpha, \beta...)^0$ (MS-structure)
└──── left / right ────┘

The feature is, furthermore, checked at the MS-structure. As a consequence, sentences such as (5a) below are ruled out at the MS-interface due to the fact that adjunction to a participle verb is not licensed by a lexical left-incorporation feature in standard Italian. Notice again the application of the mirror principle (*mi=lavato* → [lavato mi]) during the mapping from PHON into syntax, thus if we look at the process from the point of view of language production, output (5a) presupposes that the clitic was m-merged to left.

(5) a. *loro hanno mi=lavato
they have.1PL me-washed
↓
b. [loro [hanno [$(T_{prt}, v, V)^0$ mi]]] (MS-structure, not well-formed)
└───┘

The matter is still more complex, however. A clitic left-incorporates to a selecting finite verb but only under certain circumstances illustrated by the data in (6a-f).

(6) a. Loro mi$_1$=hanno lavato __$_1$ b. *Loro hanno mi$_1$=lavato __$_1$
they me-have washed they have me-washed
'They have washed me.'

---

[16] The present author is neither a native speaker of Italian nor a specialist in Italian dialectology. For comments concerning the Italian dialects, see Rizzi (1982:41 note 6) and Cardinaletti and Shlonsky (2004), p. 540, note 1.

c. *Loro hanno lavato=mi. d. *Loro mi$_1$=detestano lavare __1
they have washed-me they me-detest wash
e. Loro mi$_1$= detestano __$_1$ f. Loro detestano lavar=mi.
they me-detest they detest wash-me
‘They detest me.’ ‘They detest washing me.’

Examples (6b-c) are correctly ruled out, as these operations are not licensed by an incorporation feature. Examples (6a, e) exhibit LEFT:T/FIN and are correctly derived as grammatical. The pair (6a, d) constitutes a problem: some verbs force “clitic climbing” into the quasi-auxiliary while others reject it, yet both are tensed finite verbs. The analysis proposed thus far cannot account for this contrast.

One possibility is to assume an independent condition preventing the clitic from occurring at the Spec of the *are*-infinitival (6d), but this assumption fails to capture (6f), which shows that clitics can occur at this position. The data indicates, instead, that some V + V combinations represent a “single verbal complex” (Rizzi 1982: 5) and allow the clitic to climb in virtue of some related property (Cardinaletti and Shlonsky 2004; Rizzi 1982). The problem is to make this notion rigorous. I propose (7) as a necessary condition on clitic climbing:

(7) *A necessary condition on argument incorporation (nontechnical)*
A thematic argument of predicate P cannot incorporate into an element assigning thematic roles, or projecting arguments, independent of P.

The intuition is that the auxiliary *hanno*, unlike *detestare* ‘to detest’, does not project independent thematic roles. The idea, following for example Cardinaletti and Shlonsky (2004), Cinque (2006) and Roberts (1997), is that the higher verbal unit is deficient in its ability to project independent thematic argument structure. Because there exists so much variation with respect to clitic climbing, this property too, it seems, must be formalized by means of lexical features that can be modulated by changing the corresponding lexical entries. One relevant feature seems to be obligatory subject control (OC), which forces the higher and lower verb to share the subject thematic role (as in the case of verbs such as *try, hate*). The assumption is supported by the fact that object control verbs do not license clitic climbing. Yet there are subject control verbs (e.g., *hate*, *desire*) which do not license clitic climbing in standard Italian. A distinction between auxiliary verbs and lexical verbs seems to be the key, since pure auxiliary verbs always license clitic climbing. I propose that the feature licensing climbing in the case of auxiliary verbs is generalized to an exceptional subset of lexical OC verbs.[17] One feature that distinguishes auxiliary verbs from lexical verbs in the present framework is that the latter, but not the former, contains a lexical feature [ASP] corresponding to the projection of an independent event structure.[18] I assume that clitic climbing verbs are “quasi-functional”[19] in that they lack this feature. Suppose X^0 and Y^0 are simple or complex heads, and that the clitic is a thematic argument of Y^0; then

(8) *Clitic climbing (=rigorous reformulation of (7))*

In a configuration X^0 CL Y^0 at MS, the clitic can left-incorporate to X^0 if and only if

a. X^0 has a feature triggering obligatory subject control (OC);

b. X^0 does not have an independent event structure (feature ASP);

c. CL has the left-incorporation feature licensing the operation.

The code corresponding to this definition was added into the algorithm. We will examine the consequences and correctness of this formalization in the next section. Data (9a-b) shows that Italian argument clitics agree with the past participle verb.

(9)	a.	Loro	mi=hanno	lavato/*a.	b.	Loro	la=hanno	lavata/*-o.
		they	me-have.1PL	washed-M.SG/*F.SG		they	her-have.1PL	washed-F.SG/*M.SG
		‘They have washed me.’				‘They have washed her.’		

This data is captured, because the detached clitic lands at the Spec position of the participle predicate and triggers Spec-Head agreement at that position. Notice again the application of the mirror principle: *mi=hanno* will arrive to syntax in the order $\text{hanno}^0 + \text{mi}^0$ with the clitic occurring at the specifier position of the next verbal element (10).

---

[17] This hypothesis differs from a more radical version, namely, that the climbing verbs are auxiliaries. This does not seem correct, as many climbing verbs have several properties of lexical verbs, for example some of them may passivize, they have separate infinitival forms, and so on.

[18] This feature is responsible for the *Aktionsart* class of the verb, together with case assignment when the direct object case reflects the aspectual properties of the verb.

[19] This term comes from Cardinaletti and Shlonsky (2004).

(10) [loro [hanno^0 [lo$_1$ lavato^0 _$_1$]]] (MS-structure)
they have he.M.SG washed.M.SG
$\longleftrightarrow$ (Agree)

This analysis presupposes that there is a chain between the canonical direct object position and a specifier position of the lower verb (where it agrees), as shown in (10). Chain formation was part of the original algorithm. Clitic climbing from the position 'V1 CL V2' is rejected if V1 assigns its thematic roles independently of V2, which is codified by features OC and –ASP at V1. The analysis does not require movement for clitic incorporation (Matushansky 2006, ch. 5.1.2). Thus, movement bleeds clitic incorporation: incorporation takes place during lexical-MS mapping, not sooner.

Clitics can occur in clusters, as shown by the data in (11).

(11) Loro gli=la=hanno lavata.
they them-her-have-1pl washed
'They washed her for them.'

We assume that in standard Italian a clitic can left incorporate to a DP clitic (*la*)(i.e., it has a lexical feature LEFT:CL, D). The analysis derives (11) in the manner illustrated in (12a).[20]

(12)
a. loro gli=la=hanno lavata. (Input)
[loro [(T,V)^0 [la [⟨gli⟩ (T,v,V)^0]]]] (MS-structure)
[loro T$_{\text{fin}}$ V [la$_1$ [gli$_2$ [T$_{\text{prt}}$ v V __$_1$]] __$_2$]] (Final LF-structure)

b. *loro la=gli=hanno lavata.
they her-them-have-1pl washed

---

[20] The dative clitic is analyzed as an adjunct PP. If it were not analyzed as an adjunct, then the past participle head would have two specifiers *la* 'her' and *gli* 'to-them' and extra assumptions would be needed to handle the resulting double DP specifier construction. Being an adjunct PP does not deprive the argument from a thematic role: adjuncts are linked with their predicates via a grammatical dependency that transmits thematic roles (Brattico 2021).

c. *loro=mi hanno lavata./ *mi=loro hanno lavata
they-me have-1pl washed me-they have-1pl washed

Reverse clitic clusters are correctly classified as ungrammatical in the absence of a feature LEFT:CL,P licensing left incorporation to a PP clitic (b). The same explanation bans incorporation (both left and right) to ordinary pronouns (c). Right incorporation into an *are*-infinitival (*loro devo lavar#mi* ‘they must wash me’) is licensed by a right incorporation feature RIGHT:ARE/INF (ARE/INF being the label of the Italian *are/ere/ire*-infinitival). Right adjunction to a finite verb is ungrammatical in standard Italian. Italian lexicon lacks feature RIGHT:T/FIN preventing the operation.

A clitic cannot appear as an independent pronoun, thus a sentence such as (13a) is ungrammatical; the correct form is (13b).

(13) a. *Loro lavano mi. b. Loro lavano me.
they washed me they washed me
‘They washed me.’

This is data usually explained by relying on some notion of phonological weakness. The condition is captured here by assuming that unless the clitic is incorporated, and hence licensed by an incorporation feature, it is ungrammatical at the MS-interface. Phonological weakness is formulated as a principle that applies at PHON-MS mapping.

This completes the nontechnical description of the clitic analysis that was constructed for the purposes of this study by combining an existing computational framework with adjustments that were required by the clitic data. I do not claim that this analysis presents an optimal solution to this problem, and furthermore recognize that there exist alternatives. But since the whole analysis was implemented in a machine-readable way, we can now test it by using a rigorous protocol.

5 Computational perspective

5.1 Introduction

A possible analysis of Romance clitics was presented in the previous section, in which it was argued that clitics are detached from their hosts after the morpheme order is reversed but before they are send to syntax, then positioned into syntax into the corresponding reversed positions as phrasal pronouns and processed from that point as regular pronouns. Some evidence was cited that suggests that the analysis could derive the properties of Romance clitics, but does it?

To answer this question in any definite way we use mechanical calculation. I will go through the verification process here. The process consists of the following steps. First, the clitic data to be explained is collected into a *test corpus*. This corpus contains both grammatical and ungrammatical constructions and its purpose is to represent the core linguistic phenomenon we want the analysis to capture. Second, this data is fed to the algorithm encapsulating the theory, one sentence at a time, by a computational script that handles all sentences from the test corpus so that no human intervention plays any role. Third, the output of the algorithm is compared with native speaker judgment and native speaker intuition. All three steps are elucidated in the sections below.

5.2 Stimulus

The test corpus is a text file that contains constructions (sentences or phrases) we want to test in order to verify that the analysis works. It contains both grammatical and ungrammatical sentences. It is organized into two categories. The first category contains sentences that verify the correctness of the basic grammatical mechanisms that are *presupposed* in the critical tests. In the present study it contained tests for all lexical elements, agreement paradigms, pronouns, pro-drop constructions, word order, control and post-verbal subjects in Italian, all without clitics. This is followed by critical tests *with* clitics. The test suite corpus should be constructed by following some explicit mechanical method to prevent intentional or unintentional biases in the selection of test materials. It should furthermore cover all core examples that the analysis is purported to capture, including every example sentence cited in the main article. The whole corpus used in this study, containing 279 construction types, is provided as a supplementary to this article. The data came from several published sources, such as Lepschy and Lepschy (1998), Lorusso (2020), Serianni (2005) and from informant work by the present author. Table 1 provides a summary of the contents of the test corpus.

Table 1. Clitic constructions used in the present study to test the analysis. # = number identifier in the input and output files; *construction* = construction type with an example.

#	Construction
1-11	Lexical items and argument structures (intransitives, transitive, ditransitives)
12-19	Agreement paradigms and pronouns
20-32	pro-drop sentences
33-34	Free word order examples (in Finnish language)[a]
35-36	Control sentences [a]
37-54	Post-verbal subject sentences in Italian
55-59	Finite agreement errors
60-67	Word order errors
68-75	Direct accusative clitics (*loro mi=lavano*), with or without grammatical subject (*mi=lavano*)

76-78	Direct accusative clitics with postverbal subject (*mi=lavano loro*)
79-84	Clitic position errors, various types (e.g., **loro lavano-ni*, *mi=loro lavano*)
85	Clitic and agreement error (**io mi=lavano*)
86	Ungrammatical clitic position with postverbal subject (**lavano=mi loro*)
87	Agreement error with postverbal subject (**mi=lavano io*)
88	Agreement error, clitic position error and postverbal subject (**lavano=mi io*)
89-91	Agreement error and clitic position error (**io lavano=mi*)
92-126	Two-verb structures (*hanno + lavato*, *devono + lavare*, *stanno + lanvado*) …with or without postverbal subject …with grammatical and ungrammatical clitic incorporation … with or without overt subject (pro-drop)
127-189	Clitic in wrong position
190-198	Clitic-past participle agreement errors (**loro la=hanno lavato*)
199-210	Agreement errors and clitic placement errors (**loro hanno la=lavato*)
219-222	Subject in wrong position (??*mi=hanno loro lavato*)
223-256	Three verb structures (*loro mi=hanno volute lavare*), both grammatical and ungrammatical
257-268	Indirect clitics and clitic clusters (*loro gli=lavano Luisa*, *la=gli=hanno lavata*)
269-285	Restructuring (*loro sembrano lavare=mi*)
286-296	Reflexive clitics (*io mi=lavo*)
297	Si-subject constructions (*si=dorme*)

[a] These were included because I wanted the analysis to provide correct solutions also for certain word order facts and standard cases of control.

The test corpus fills several roles. Suppose we wanted to compare two theories of clitics. We can now do this by using the same, agreed-upon core test corpus that any linguistically plausible model is minimally required to handle. Indeed, construction of such core datasets constitutes one of the most important aspects of a computational study of this kind and is routinely used by the CL community when comparing the performance of various engineering models, such as practical parsers.[21] In addition, by using a fixed test corpus it is possible to test the consequences of any change proposed into the analysis. Suppose, for example, it were proposed that a better analysis would result if the clitic pronouns were interpreted as NPs and not full DPs. We can modify the analysis (the program code) accordingly and then run the modified algorithm *with the same test corpus* and compare the results without ambiguity. Similarly, it will be easy to test what happens if we perform clitic detachment before the application of the mirror principle. In the short term such theory comparison could be done by measuring the relationship between the complexity of the analysis and the size and heterogeneity of the data the analysis captures, measuring in some sense the “depth” of the analysis, although in the long run such comparisons will be done by the scientific community as a whole by using whatever criteria they come to consider as most compelling. The main point here, however, is that there is a reason why the CL

---

[21] Linguists cannot typically use the datasets employed by the CL community, as they do not contain ungrammatical sentences and do not target specific linguistic constructions or topics in a controlled manner. Linguistically relevant datasets must be specifically curated, as they are in any empirical science, so that they cover the empirical phenomenon the proposed hypothesis is purported to explain, and furthermore exclude all irrelevant and uncontrolled properties and constructions.

community relies on established test corpora in comparing theoretical approaches. Theoretical linguists should do the same.

5.3 Procedure

The test environment was built in the following way. A complete test is launched by running the main script. The main script initializes the test program, reads external information from separate files (test corpus, lexical information) and sends each input sentence from the test suite corpus to the recognition algorithm. The recognition algorithm calls for a recursive parser that guides the application of computational operations (e.g., Merge, Move, Agree) on the basis of the input strings and by doing this produces a set of acceptable and semantically interpretable parses, which it sends back to the main script. The testing architecture is illustrated in Figure 2.

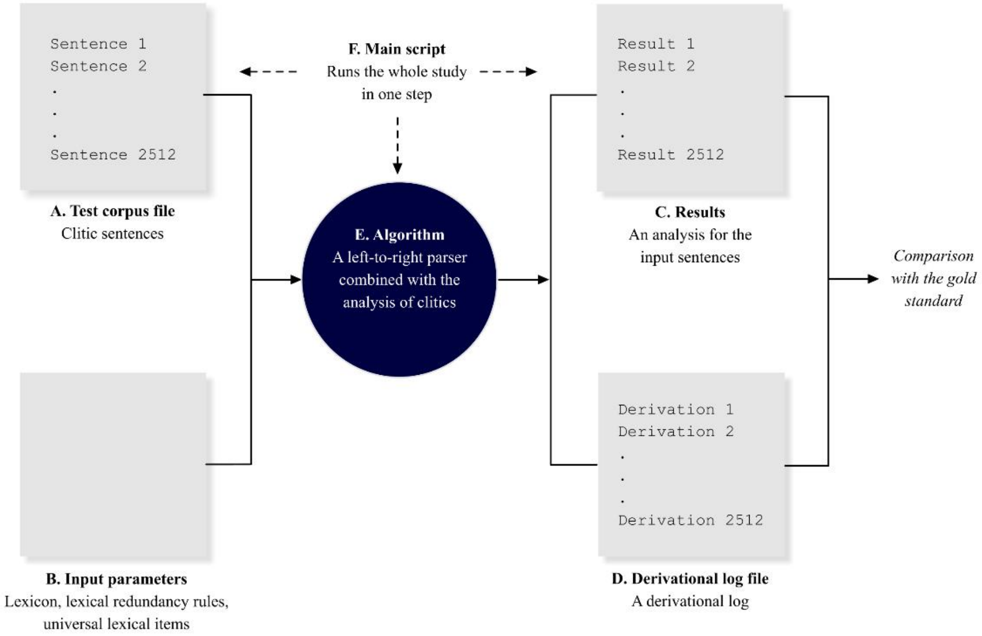


Figure 2. A main script handles the pipeline, in which the input files (test suite corpus together with the lexicon) are read and fed into the recognition algorithm that utilizes a recursive parser consisting of the phonological branch and narrow syntax. Two output files are provided: summarized results, which contains grammatical judgments and structural analyses, and a log file, which holds a record of complete derivations, thus all linguistically relevant computational steps that happened run-time during the analysis.

The algorithm provides several types of output; here we are specifically interested in the grammaticality judgments, syntactic analyses, semantic interpretations (Results, Figure 1, also *Supplementary 2 Results.txt*) and detailed derivations (Derivational log file, also *Supplementary 3 Derivations.txt*).[22] The algorithm provides detailed output also concerning the number and type of computational operations it consumes while processing each sentence, allowing the researcher to compare its performance with psycholinguistic experiments and thus to assess, if such performance factors were useful from the point of view of the study, also its psycholinguistic plausibility. Regardless of the type of output that is eventually evaluated, the idea is to compare the output with data from native speakers to make sure that the two matches and thus that the model satisfies observational adequacy.

The test corpus and the testing procedure used in this study were both feasible from the point of view of practical research project. It takes an average computer less than one minute to go through a test corpus of this size. Practical work with an algorithm like this has shown that a test corpus containing anywhere between one and three thousand constructions is manageable for one person to verify and analyze. Although the number might seem low at first, it is important to keep in mind that any change to the theory, no matter how small, requires that the whole test corpus is analyzed anew. Furthermore, verifying the correctness of the output of this type (grammaticality judgments, syntactic analyses and meanings) requires considerable resources from a human participant.

5.4 Results

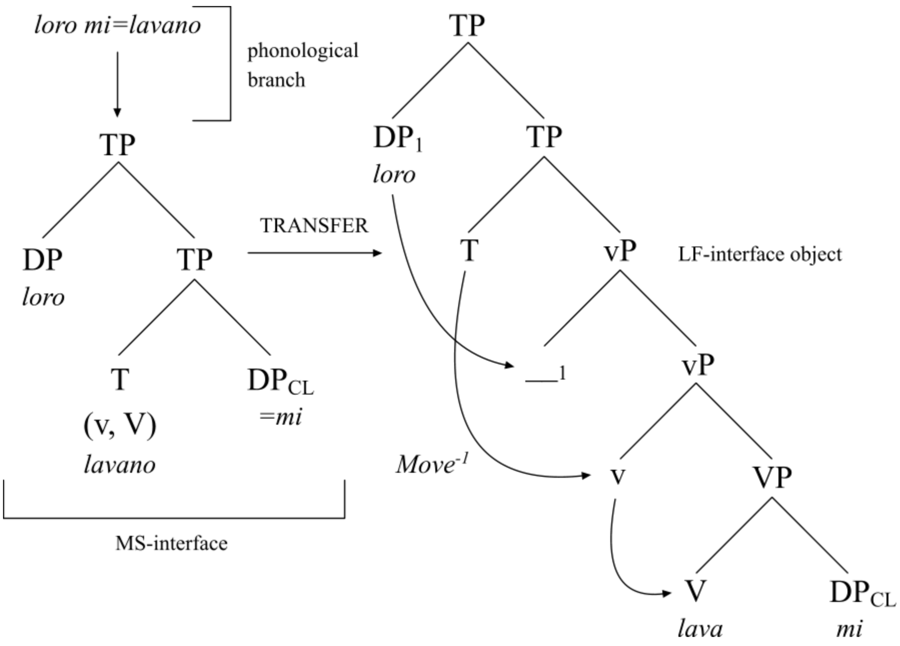
Inspection of the output of the algorithm shows that it produced correct results with few exceptions, discussed at the end of this section. Let us consider some of the key examples first. A simple direct object clitic sentence such as (14) is analysed by the recognition algorithm as shown in (15).

(14) loro mi=lavano (#68)
they me=wash
'They washed me.'

---

[22] These are raw datafiles. Perhaps it is better to replace these with one supplementary document that summarizes the results and then leave the raw data online.

(15)



The derivation can be found from lines 21340-21543 in the log file (*Supplementary 3 Derivations.txt*), part of which is reproduced in Figure 4, which is a more formal replication of the process illustrated in (15).

```
21341 ['loro', 'mi=lavano']
21342 Using lexicon "lexicon.txt".
21343
21344             =None
21345
21346         Next word contains multiple morphemes ['n$', 'def$', '3p$', 'pl$', 'D$', 'loro-']
21347         Storing inflectional feature ['-', 'LANG:EN', 'PHI:GEN:N'] into working memory.
21348
21349     1. Consume "def$"
21350         Storing inflectional feature ['-', 'LANG:EN', 'PHI:DET:DEF'] into working memory.
21351
21352     2. Consume "3p$"
21353         Storing inflectional feature ['-', 'LANG:EN', 'PHI:PER:3'] into working memory.
21354
21355     3. Consume "pl$"
21356         Storing inflectional feature ['-', 'LANG:EN', 'PHI:NUM:PL'] into working memory.
21357
21358     4. Consume "D$"
21359         Adding inflectional features {'LANG:EN', '-', 'PHI:DET:DEF', 'PHI:PER:3', 'PHI:GEN:N',
21360             =D
21361
21362     6. Consume "loro"
21363
21364         D + loro
21365             Filtering out impossible merge sites...
21366             Sink "loro" into D because they are inside the same phonological word.
21367             =D{N}
21368
21369         Next word contains multiple morphemes ['lavano=', 'mi']
21370
21371         Next word contains multiple morphemes ['inc$', '[-no]$', 'T/fin$', 'v$', 'lava-']
21372         Storing inflectional feature ['-', 'INCORPORATED', 'LANG:EN'] into working memory.
21373
```

```
21482     >>> Trying candidate spellout structure [[D loro] [T/fin{v,V} D{N}]]
21483         Checking surface conditions...
21484             Clitic D{N} left-incorporated to T/fin{v,V}
21485         Reconstructing...
21486             Clitic D{N} left-incorporated to T/fin{v,V}
21487                 1. Head movement reconstruction:
21488                     Target v{V} in T/fin
21489                     =[[D loro] [T/fin [v{V} D{N}]]]
21490                     Target lava in v
21491                     =[[D loro] [T/fin [v [lava D{N}]]]]
21492                 =[[D loro] [T/fin [v [lava [D mi(cl)]]]]]
21493                 2. Feature processing:
21494                 =[[D loro] [T/fin [v [lava [D mi(cl)]]]]]
21495                 3. Extraposition:
21496                 =[[D loro] [T/fin [v [lava [D mi(cl)]]]]]
21497                 4. Floater movement reconstruction:
21498                 =[[D loro] [T/fin [v [lava [D mi(cl)]]]]]
21499                 5. Phrasal movement reconstruction:
21500                     Moving "D loro " into memory buffer from SPEC of "T/fin".
21501                     Memory buffer: [[D N]]
21502                     Dropping constituent [D loro] from memory buffer into Spec of v
21503                     Result [[D loro] [T/fin [[D loro] [v [lava [D mi(cl)]]]]]]
21504                 =[[D loro]:3 [T/fin [__:3 [v [lava [D mi(cl)]]]]]]
21505                 6. Agreement reconstruction:
21506                     T/fin acquired PHI:DET:DEF from the edge of T/fin.
21507                     T/fin acquired PHI:GEN:N from the edge of T/fin.
21508                     T/fin acquired PHI:NUM:PL from the edge of T/fin.
21509                     T/fin acquired PHI:PER:3 from the edge of T/fin.
21510                 =[[D loro]:3 [T/fin [__:3 [v [lava [D mi(cl)]]]]]]
21511                 7. Last resort extraposition:
21512             = [[D loro] [T/fin [[D loro] [v [lava [D mi(cl)]]]]]]
21513         Checking LF-interface conditions.
21514             Transferring [[D loro]:3 [T/fin [__:3 [v [lava [D mi(cl)]]]]]] into Conceptual-Inter
21515                 v with ['PHI:DET:_', 'PHI:NUM:_', 'PHI:PER:_'] was associated at LF with:
21516                     1. [D loro]    (alternatives: 2. T/fin )
21517                 lava with ['PHI:DET:_', 'PHI:NUM:_', 'PHI:PER:_'] was associated at LF with:
21518                     1. [D mi(cl)]    (alternatives: 2. [D loro] 3. T/fin )
21519                 Transfer to C-I successful.
21520                 Semantic interpretation/predicates and arguments: {'lava([D mi(cl)])', 'v([D lor
21521 --------------------------------------------------------------------------------------------
21522                                         All tests passed
21523 --------------------------------------------------------------------------------------------
```

Morphological decomposition of "loro"

Morphological decomposition of "mi=lavano"
Application of the mirror principle

MS-interface object

Checking incorporation at MS-interface

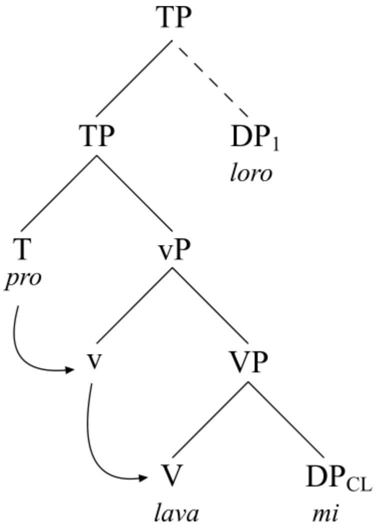
Creation of inverse chains

Result
Semantic interpretation

Figure 4. Screenshot of the derivational log file capturing the derivation of sentence (14)/(15).

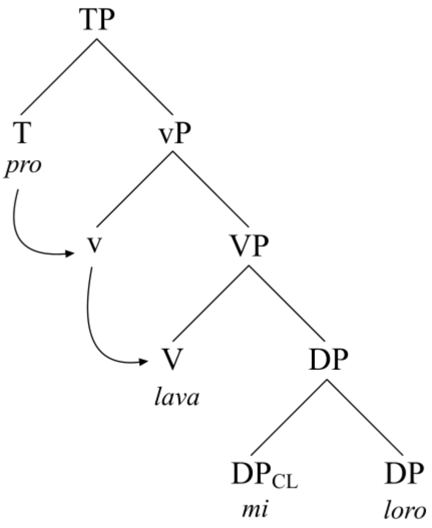
The phonological input string is decomposed into morphemes, the order is reversed, and the resulting morphemes are mapped to lexical items which are fed into syntax, where they are merged (in the incoming order) to a MS-structure representation (structure at the left) (lines 21349-21480). Clitic conditions are checked at the MS-structure (21484-5). The clitic has a feature licensing left incorporation to *lavano*, so the filter lets the derivation to proceed. The enclitic is merged directly to the canonical object position due to the reversal that takes place in the phonological branch (*mi*=*lavano* → *lavano* + *mi*), hence there is no "clitic climbing" in the syntactic component. This can be seen from line 21484. *Transfer* performs reconstruction by creating phrasal (DP) and head (T^0-v^0-V^0) chains, shown in the final phrase structure at right (lines 21489-21519). The same sentence with a postverbal subject construction is deduced as (16)(#76).

(16)



The algorithm analyses the postverbal subject as a right adjunct (adjunction will be marked by a dashed arrow, ⟨ ⟩ in the machine-generated output). The thematic agent is therefore missing, but the ‘agent’ is reconstructed from the pro-element at T which is in turn reconstructed from the overt agreement features at the finite verb (following the suggestion of Alexiadou and Anagnostopoulou 1998 and Brattico 2021). Notice that this solution is attempted only after the parser tries a construction in which the postverbal subject is in the complement position of the verb and the clitic is at its specifier (17) (line 22916). This derivation crashes at LF because the clitic DP cannot be assigned a thematic role (line 22949).

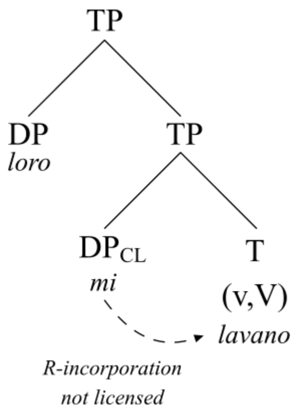
(17)



Sentence **loro lavano=mi* (#79) is correctly ruled ungrammatical because there is no lexical feature licensing right incorporation of a clitic to *lavano* ‘washed’. The relevant MS-interface representation is illustrated at (18)(line 23744).

Most of the wrong clitic placements are ruled out by the same mechanism and thus depend on the lexical features and the MS-structure filter that checks them (line 23747).

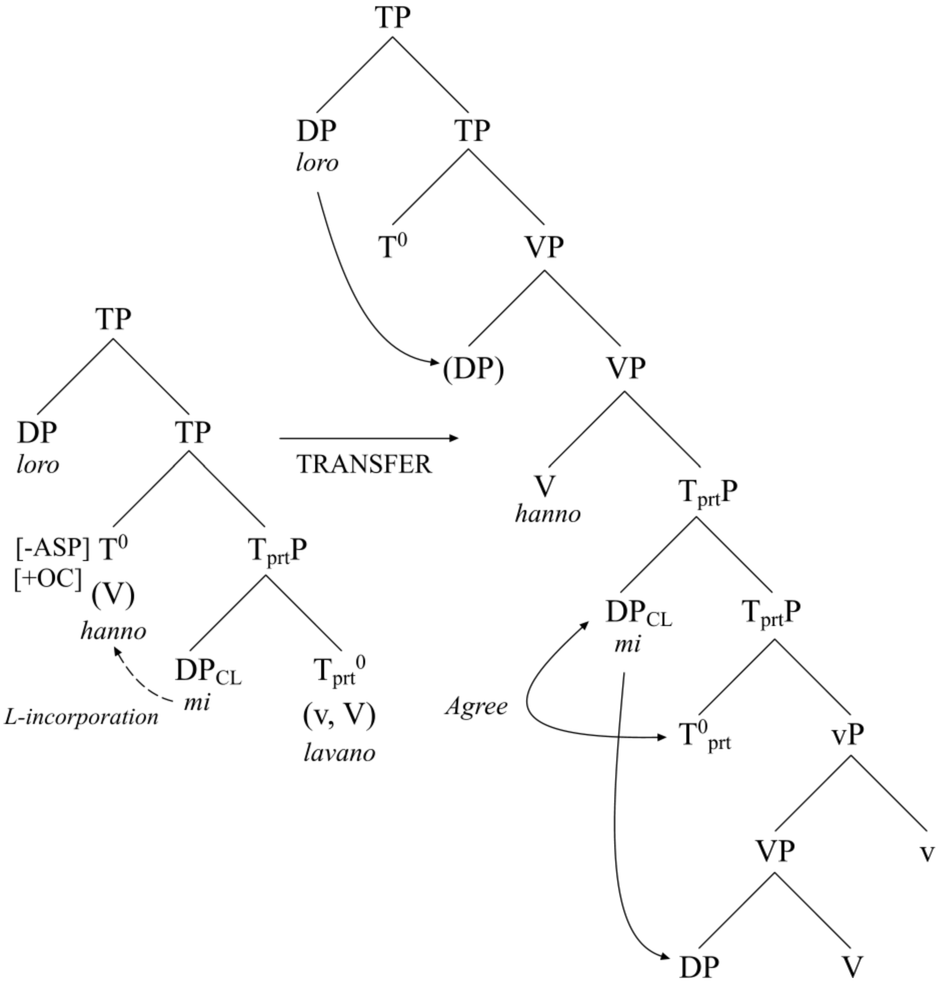
(18)



If we examine the same sentence from the point of view of an enumeration algorithm, then the licensing condition blocking (18) prevents m-merger of DP_{CL} to the complex verbal head to its right. If the clitic is left hanging as an independent pronoun, the recognition algorithm will reject it because a freestanding clitic satisfies no positive licensing condition. As a consequence, the fact that clitics are “weak” pronouns has no other meaning in the current formalization than the fact that they cannot occur as free morphemes.

Next, we examine various cases of two-verb structures exhibiting clitic climbing (#92-222). Sentence *loro mi=hanno lavato* ‘they me-have washed’ (# 92) is analysed as (19).

(19)



The MS-interface object is shown at left, in which the clitic pronoun occurs between the auxiliary verb *hanno* ‘have’ and the participle *lavato* ‘washed’. Its position had been reversed in the phonological branch before the creation of the MS-structure (line 27158). L-incorporation of the clitic to the auxiliary is licensed due to the presence of the two features OC (obligatory control) and –ASP (no independent thematic roles are projected)(line 27160); if the selecting verb lacks on of these features, the operation is not licensed. Reconstruction then creates the LF-interface object at the right (line 27197), in which the clitic pronoun agrees with the participle verb and is reconstructed into the direct object position. The grammatical subject is reconstructed into SpecVP position.[23] These constructions were correctly derived

---

[23] The specifier of v is empty, which triggers an antecedent algorithm that wrongly targets the agreement features (pro-element) at the T_{prt} resulting in a reflexive interpretation ‘me washed myself’. If we force the subject to reconstruct into Spec,vP, it will reconstruct through SpecT_{prt}P and cause S-T_{prt} agreement in the absence of the clitic. Changing the antecedent algorithm so that it

when they occur without the subject, with postverbal subject, and in connection with *devono* + *lavare* (#99-116, 168-179) and *stanno* + *lavando* (#117-126, 180-189). Various errors were successfully tested as follows: clitic in wrong position (#127-148) with postverbal subject (#149-167), and wrong subject position (#211-222). Three-verb sentences (*loro mi=hanno volute lavare*) can be found from #223-256. Wrong clitic positions (e.g., **loro hanno mi=lavato*, **loro hanno lavatomi*, **io=mi sono lavato*, **loro hanno mi=voluto lavare*) were all rejected in the same way as (20). The clitic occurs in a position at MS-level at which it is not licensed (by L-incorporation, R-incorporation). These were all derived correctly. An interesting observation is recorded in #211-222, in which I tried sentences in which the grammatical subject occurs between the auxiliary and the participle verb (**mi=hanno loro lavato* 'me-have they washed')(#211) and which were correctly deduced as ungrammatical (I have obtained conflicting judgments for these sentences and assume that they are ungrammatical or marginal). The recognition algorithm provides (20) as the MS-interface object.

(20) [$_{\mathrm{TP}}$ (T,V)^0 [$_{\mathrm{TP}}$ DP$_{\mathrm{CL}}$ [$_{\mathrm{TP}}$ [*loro*] (T$_{\mathrm{prt}}$,v,V)^0]]](line 73500)

It is interesting to note that the algorithm reconstructs *mi* and *loro* into correct positions inside the VP but is unable to determine the grammatical head hosting the clitic. A structure with unknown/unrecognized elements cannot be interpreted at LF (line 73547).

(21) [$_{\mathrm{TP}}$T [$_{\mathrm{TP}}$ ha [$_{\mathrm{?P}}$ *mi*$_2$ [$_{\mathrm{?P}}$? [$_{\mathrm{TprtP}}$ *loro*$_1$ [$_{\mathrm{TprtP}}$ T$_{\mathrm{prt}}$ [$_{\mathrm{vP}}$ $\_\_\__1$ [$_{\mathrm{vP}}$ v [$_{\mathrm{VP}}$ V $\_\_\__2$]]]]]]]]

I cannot say if this is a right or wrong way to reject these sentences, but this nevertheless is an automatic and unintended consequence of the axioms constituting the proposed analysis.

The analysis deduces the properties of (22) incorrectly, wrongly predicting that the sentence should be grammatical. This reveals a problem in the analysis that I have not been able to solve.

---

skips T$_{\mathrm{prt}}$ would be completely ad hoc. I do not know how to solve this problem in an elegant way. It suggests that fundamental unsolved problems remain.

(22)	*Loro	mi=lavano	Luisa.
	They	me-washed	Luisa
	'*They washed me Luisa'		

The recognition algorithm interprets *Luisa* as a right adjunct, as we cannot outright rule out DP adjuncts (e.g., *the whole day*). If we reject proper name DP adjuncts, then, by using the current assumptions, there is no way to derive Italian postverbal subjects. When the thematic agent is in a right-adjoined position, there is no grammatical link between the adjoined subject and the agreement features at T (v is linked correctly with the agreement cluster at T). I leave this problem for future; it is a clear defect in the analysis as it stands at present.

Based on this small-scale study it is impossible to say whether the analysis has merit and especially how it compares with alternative hypotheses or hypotheses that start off from completely different assumptions. An anonymous reviewer of this article proposes that a better starting point would be historical analysis. My point, however, is not to argue that the analysis itself is superior to any alternative, real or imagined; rather, the analysis was constructed in order to demonstrate that the move from natural language justification into a full natural scientific, formal justification, is both possible and feasible despite the obstacles created by the combinatorial nature of the data. This brings several potential benefits to linguistic theorizing, worth repeating before concluding.

One strength of the proposed methodology is that whether the model is superior to an alternative, say an analysis based on historical patterns, can be examined unambiguously and thus free from appeals to subjective intuition or to other dubious criteria. The procedure would begin by agreeing upon a common core dataset that any theory is required to capture, a step that cannot in my view be objected on any rational grounds in any scientific enterprise, and then by comparing the algorithms (hypotheses) that correctly deduce that dataset. It makes no difference if the hypothesis relies mostly on cognitive computational operations, such as the present analysis, or emphasizes historical patterns. If the latter, then the cognitive part of the theory is likely to rely more on general cognitive operations, such as supramodal pattern recognition, memorization, statistical inference and/or inductive generalization, enriched with a more substantial contribution form a history of observation ("data")(see Lewis et al. 2005) . There is no justification for not demonstrating the adequacy of a theory of this type by using the same type of deductive reasoning. Similarly, it could be objected that the formal computational mechanisms and representations posited by the present proposal are unnecessary complex and unnecessarily rich, and that a simpler theory would result if we assumed something less (e.g., a dependency grammatical formalism, conceptual structure, even a connectionist model). That could be true, but the claim needs to be demonstrated. It can be demonstrated by showing that a simpler model does derive the same dataset. Moreover, all the assumptions posited in the linear phase algorithm that handle the clitic data are necessary in the sense that if we removed any one of them then the whole model would fail immediately. Removing and/or simplifying components of a working computational theory is not a trivial matter and cannot be accomplished by injecting natural language speculation into the mix.[24] That being said, I do not see any a priori reason why the analysis proposed here could not be simplified, even to a point of being implemented by an extremely simple and thus attractive connectionist finite-state machine that learns everything from a simple input dataset.

In addition to its merits, the methodology has limitations. It requires that the leading ideas be converted into rigorous enough format so that a translation into a machine-readable system becomes possible even in principle. Such formalization might serve no purpose in some contexts, for example, if the point is to gather data, conduct linguistic experiments, or merely report linguistic intuitions. Thus, the approach seems applicable and perhaps also justifiable in theoretical linguistics, in which theory construction and testing constitutes the main focus of the enterprise. Furthermore, writing machine-readable code is not resource-free activity and tends to lead into linguistically irrelevant errors ("bugs") that require resources and expertise to correct. This means that some work that is not directly relevant to linguistic theorizing per se will be required from the research team. On the other hand, most scientific disciplines operate under similar conditions. Conducting a psycholinguistic experiment, for example, involves logistic challenges and setting up statistical analyses that have little to do with the subject matter itself.

Although discovery does not constitute the focus of this article, the topic is not irrelevant. The experiment revealed a number of ways in which rigorous calculation can aid discovery. Once the testing protocol in place, it become possible to experiment with alternative hypotheses and assumptions by changing the code and running it through the whole test corpus. By directly comparing the outcomes of the previous and new version it was possible to see with one glance what the logical consequences of the adjustments were. This makes it possible to guide the model towards an analysis with increased coverage and away from ideas that felt intuitively useful at first but led into too many unintended

---

[24] This is because almost every component of the theory depends on what the other components are doing. This is why a small error or change somewhere in the program will almost always crash the model or at the very least lead into many unforeseen consequences. It also means that interpreting what one component of the model is doing is quite impossible when considered in isolation.

consequences. In addition, formalization forced me to focus on a number of empirical and theoretical problems whose existence I was only vaguely aware of, and in some cases, not aware of at all. Second, by incorporating clitic processing into the morphological component forced a number of unanticipated changes into the way morphology was processed. For example, it was not trivial to separate clitics from other morphemes, as the algorithm did not have access to the lexicon before items were literally streamed into the syntactic component.

6 Conclusions

It was argued that scientific justification could play a useful role in theoretical linguistics. Computational tools and hardware have developed to a point at which they can be applied with little cost even to problems with considerable computational complexity. The theory of Romance clitics was used as an example to illustrate the merits and challenges of the methodology, and a recognition algorithm was developed to deduce properties of clitic constructions in standard Italian.

References

Alexiadou, Artemis and Elena Anagnostopoulou. 1998. “Parametrizing AGR: Word Order, V-Movement and EPP-Checking.” *Natural Language & Linguistic Theory* 16:491–539.

Baker, Mark. 1985. “The Mirror Principle and Morphosyntactic Explanation.” *Linguistic Inquiry* 16:373–415.

Baker, Mark. 1988. *Incorporation. A Theory of Grammatical Function Changing*. University of Chicago Press.

Bangu, Sorin. 2012. *The Applicability of Mathematics in Science: Indispensability and Ontology*. New York: Macmillan.

Bochner, Salomon. 1966. *The Role of Mathematics in the Rise of Science*. Princeton: Princeton University Press.

Brattico, Pauli. 2019. *A Computational Implementation of a Linear Phase Parser. The Framework and Technical Documentation*. Pavia.

Brattico, Pauli. 2020. “Finnish Word Order: Does Language Comprehension Matter?” *Nordic Journal of Linguistics* 1-33. doi:10.1017/S0332586520000098

Brattico, Pauli. 2021. “Null arguments and the inverse problem”. *Glossa* 6: 1-29. doi:10.5334/gjgl.1189

Brattico, Pauli and Cristiano Chesi. 2020. “A Top-down, Parser-Friendly Approach to Operator Movement and Pied-Piping.” *Lingua* 233:102760.

Cann, Ronnie, Ruth Kempson, and Lutz Marten. 2005. *The Dynamics of Language: An Introduction (Syntax and Semantics, Volume 35)*. Amsterdam: Elsevier Academic Press.

Cardinaletti, Anna and Ur Shlonsky. 2004. “Clitic Positions and Restructuring in Italian.” *Linguistic Inquiry* 35(4):519–57.

Chesi, Cristiano. 2004. “Phases and Cartography in Linguistic Computation: Toward a Cognitively Motivated Computational Model of Linguistic Competence.” Universita di Siena, Siena.

Chesi, Cristiano. 2012. *Competence and Computation: Toward a Processing Friendly Minimalist Grammar*. Padova: Unipress.

Chomsky, Noam. 1957. *Syntactic Structures*. The Hague: Mouton.

Chomsky, Noam. 1965. *Aspects of the Theory of Syntax*. Cambridge, MA.: MIT Press.

Chomsky, Noam. 1981. *Lectures in Government and Binding: The Pisa Lectures*. Dordrecht: Foris.

Chomsky, Noam. 2000. “Minimalist Inquiries: The Framework.” Pp. 89–156 in *Step by Step: Essays on Minimalist Syntax in Honor of Howard Lasnik*, edited by R. Martin, D. Michaels, and J. Uriagereka. Cambridge, MA.: MIT Press.

Chomsky, Noam, Ángel J. Gallego, and Dennis Ott. 2019. “Generative Grammar and the Faculty of Language: Insights, Questions, and Challenges.” *Catalan Journal of Linguistics* Special Is:229–61.

Cinque, Guglielmo. 2006. *Restructuring and Functional Heads*. Oxford: Oxford University Press.

Clagett, Marshall. 1961. *The Science of Mechanics in the Middle Ages*. London: Oxford University Press.

Collins, Chris and Edward Stabler. 2016. “A Formalization of Minimalist Syntax.” *Syntax* 19(1):43–78.

Cuomo, Serafina. 2001. *Ancient Mathematics*. London: Routledge.

Dyson, Freeman. 1964. “Mathematics in the Physical Sciences.” *Scientific American* 211:128–46.

Gingras, Yves. 2001. “What Did Mathematics Do to Physics?” *History of Science* xxxix:383–416.

Grant, Edward. 1971. *Physical Science in the Middle Ages*. Cambridge: Cambridge University Press.

Itkonen, Esa. 1983. *Causality in Linguistic Theory*. Amsterdam: John Benjamins.

Julien, Marit. 2002. *Syntactic Heads and Word Formation*. Oxford: Oxford University Press.

Kayne, Richard. 1975. *French Syntax*. Cambridge, MA.: MIT Press.

Kayne, Richard. 1989. “Null Subject and Clitic Climbing.” Pp. 239–61 in *The Null Subject Parameter*, edited by O. Jaeggli and K. J. Safir. Amsterdam: Kluwer.

Lepschy, A. and G. Lepschy. 1998. *The Italian Language Today*. London: Routledge.

Lewis, R. L. and S. Vasishth. 2005. “An Activation-based Model of Sentence Processing as Skilled Memory Retrieval.” *Cognitive Science* 29(3):375–419.

Lorusso, Paolo. 2020. *Clitic Inventory*.

Matushansky, Ora. 2006. “Head Movement in Linguistic Theory.” *Linguistic Inquiry* 37:69–109.

May, Robert M. 2004. “Uses and Abuses of Mathematics in Biology.” *Science* 303:790–93.

Merchant, D. C. 2019. Idioms at the interface(s): Towards a psycholinguistically grounded model of sentence generation. Ph.D. disseratation, University of Georgia.

Muysken, Pieter. 1982. “Parametrizating the Notion of ‘Head.’” *Journal of Linguistic Research* 2:57–75.

Newton, Isaac. 1999 [1726]. *The Principia: mathematical Principles of Natural Philosophy* (3rd edition), University of California Press, English translation by I. Bernard Cohen, Anne Whitman and Julia Budenz.

Partee, Barbara H., Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics*. Kluwer Academic Publishers.

Phillips, Colin. 1996. “Order and Structure.” MIT, Cambridge, MA., Ph.D. Dissertation.

Rizzi, Luigi. 1982. *Issues in Italian Syntax*. Amsterdam: Foris.

Roberts, Ian. 1997. “Restructuring, Head Movement and Locality.” *Linguistic Inquiry* 28:423–60.

Roberts, Ian. 2010. *Agreement and Head Movement: Clitics, Incorporation, and Defective Goals*. Cambridge, MA.: MIT Press.

Serianni, L. 2005. *Grammatica Italiana*. Torino: Utet.

Wolfram, Stephen. 1984. “Computer Software in Science and Mathematics.” *Scientific American* 251:188–204.