A position paper on intuition (sketch)

Preface. The following is a slightly edited version of a letter I wrote to my advisor in the linguistics program at UCSC, inspired by a conversation we had had earlier that day. My advisor felt strongly that diagrams and such are hallmarks of good mathematical writing, in fact of all good technical writing; I tried to point out why such devices often do more harm than good. This led to a discussion of methodology in general, and what my hopes are for a science of language. The focus of that discussion was on the role of intuition in science, and this letter is mainly a summary of that dimension of our conversation. (End of Preface .)

I got the —hopefully correct— impression that some of the ideas I put forth today struck a chord with you. I hope so, because they represent a truly new approach to human inquiry that I find thrilling. Here's a little summary of the points I made.

Intuition is no substitute for an argument; on that much we all agree. However, many feel that intuition is our only means of grappling with and finding solutions to scientific problems. (Occasionally we are given the Hobson's choice of intuition versus exhaustive case analysis.) I agree that for thousands of years we haven't been able to do much better, but I see this not as a "fact of life" but as a "status quo" which can and should be improved upon.

Why? What is the problem with using our intuition to help us find solutions?

- Because intuition is a reactionary and implicit process, we cannot depend on it to work when we need it to. On the other hand, explicit heuristics can always be applied where they are applicable, and can always be refined when they do not work.
- Because intuition is reactionary, it bears the burden of our past experiences, often thus introducing irrelevant overspecificities into our arguments. Moreover, analogy is not suitable for dealing with radically new phenomena which are not analogous to anything we already understand. On the other hand, a methodology that forces us to justify every step we take allows us to limit what we let creep in from our past experience —even to nothing, if we so desire— . And if we cannot understand a phenomenon in terms of what we already know, what can we do but confront the details of that phenomenon head-on? This is precisely what a constructive methodology prescribes.
- Because intuition is implicit, it can not be explicitly taught to others. Rather, just as in the guilds, a student of an intuitive science must serve as apprentice to masters, then watch and imitate. He never gains explicit understanding of what he is doing, but must simply have faith that time-tested methods —which may have no convincing justification— will get the job done. Those who cannot pick up the craft by imitation will never learn it. On the other hand, an explicit methodology is eminently teachable, by definition. (Which is not to say it is easily learned or mastered.)
- Because there are most likely no "deep truths" hidden in our brains, it is a matter of chance whether a step or solution our intuition provides is viable or not. If not, we may be led astray for a long period of time, in which case the temptation to shove a square peg in a round hole becomes irresistible. On the other hand, a constructive

methodology aims to avoid this element of chance by analyzing the problem in order to find the next step. Often such a step is forced, or at least sweetly reasonable.

- Because there is no guarantee an intuitive solution will not introduce irrelevant, over-specific detail, intuitive approaches often miss out on generalizations. On the other hand, a methodology that strives for concision and simplicity often finds those generalizations by not introducing the irrelevant detail in the first place.
- Although the use of intuition is supposed to be limited to the finding of solutions, it inevitably affects the argumentation process itself. We can easily be led to the incorrect conclusion that an abstract claim is true, simply because our intuitive understanding of that claim uses a more specific model in which it is true. Very often, the incorrect steps in an argument are those which are supposed to be "intuitively clear" . A constructive approach avoids these pitfalls by being as explicit as possible.
- etc.

Most scientists' reaction to the above is something like, "It would be nice to have a methodology that avoided all the problems of an intuitive approach, but unfortunately there's no other way." . And as I said, two years ago, I would have agreed. (It was this fact that made math eventually very distasteful to me.) But then I came across the methodological programs for mathematics and computing science, designed over the past few decades by several Dutch professors. It took a long time for me to absorb the flavor of these programs, but now that I have, I cannot go back. Now that I have seen constructive ways of solving problems, judicious ways of using data, I cannot be content with the intuitive method in science. In fact, I even feel safe saying that reliance on intuition is pre-scientific. I would be happy to direct you to many excellent readings on these methodologies; if you have an open mind, I've no doubt they will inspire you to think about how they can be applied to a science of language.

There is a tendency for natural scientists to confront these ideas and say, "This is relevant for math and computing science, but not for our science." . I would strongly advise against making such judgements, because when it comes down to it, I think all sciences face the same problems. Many of the issues raised by the Dutch professors are not particular to math or computing science, but are in fact general to all of human reasoning. It may surprise you that many of the same criticisms and laments you feel are particular to linguistics are strikingly similar to those offered against this methodology in mathematics, and those that abounded at the birth of computing science.

For example, consider this quotation by Christopher Strachey from 1969:

> I am quite convinced that in fact computing will become a very important science. But at the moment we are in a very primitive state of development; we don't know the basic principles yet and we must learn them first. If universities spend their time teaching the state of the art, they will not discover these principles and that, surely, is what academics should be doing.

Then in 1972, Edsger Dijkstra, after giving a rather long and detailed solution to a very simple computing science problem, writes:

> Before turning our attention away from this example [. . .] I should like to add some remarks, because I have the uneasy feeling that by now some of my readers (in particular experienced and competent programmers) will be terribly irritated, viz. those readers for whom [my] program is so obviously correct that they wonder what all the fuss is about: "Why his pompous statement? [...] Certainly he does not expect us, who have work to do, to supply such lengthy proofs, with all the mathematical dressing, whenever we use such a simple loop as that?" [...] To tell the honest truth: the pomp and length of the above proof infuriate me as well! But at present I cannot do much better if I really try to prove the correctness of the program.

In 1972, there was no science of programming; there was practically no alternative to intuition or case analysis, although several computing scientists had eloquently pointed out the drawbacks to such approaches. Four years later, in 1976, Dijkstra published his seminal *A Discipline of Programming* , which forever changed the field of computing science —though not necessarily most or even many computing scientists— . Computing science now had a methodology, a beautiful formal framework that could handle the abstraction of a wide variety of problems, and a beautiful calculational system that could solve problems like the above one concisely, efficiently, and elegantly.

And lest you think computing science doesn't have to deal with "the real world" as does linguistics, let me quote from a 1977 paper by DeMillo, Lipton, and Perlis, which argues against the sort of framework Dijkstra created:

"*Real* programs deal with *real* human activity and are thus detailed and messy." .

This is a pre-scientific attitude at its most flagrant. A similar view of linguistics was expressed to me recently by Chris Kennedy. I had mentioned that the theories we were developing in our classes were quickly getting messy and ugly. He replied, "Of course! Language is ugly!" . I forgive Chris, but I must maintain that such attitudes are unproductive if one wishes to improve on the status quo. Natural phenomena seem ugly only because our intuition —read: immediate apprehension— cannot pick out the beautiful patterns at first glance; it often takes years of careful abstraction and analysis to find that beauty. But if we follow DeMillo, Lipton, Perlis, Kennedy, and countless others in eliminating beauty and perfection as (perhaps unattainable) goals, what is the point of doing science?

Chomsky, summarizing Gallileo, once said something like, "If the data don't fit the theory, the data are probably wrong." . To some extent, I agree with that. But in Chomsky's case, it could just as easily have been a bad theory. We can do better.

Addendum

You mentioned that one needs to use diagrams and examples when doing mathematics. Indeed, I know a lot of people who feel this way, yet I think the need for overspecific detail is just a reflex of our training. In general, we have few tools for grappling with the abstract directly, and thus we are not equipped to handle it. When we do, it is usually by analogy. (For example, $\mathbf{R}^n$ in terms of $\mathbf{R}^2$ or $\mathbf{R}^3$.)

Speaking more broadly: when, in a scientific domain, arguments become difficult to

construct without reference to a diagram or example, this is a telltale sign that that domain has not matured yet. Geometry is perhaps the most infamous example of this problem in mathematics, and as we saw above, it dominated the whole of computing science for many years. In short, the question of how to reason without relying on intuition or pictorial aids is the question of how to turn a domain of human inquiry into a science.

I am proud to say that the group I am aligned with has made tremendous advances in the domains of computing science and mathematics methodology, to the point where we can now give explicit, teachable, constructive methods for designing beautiful, concise, easily-verifiable programs and proofs, without resorting to case analysis or intuition.

It goes without saying that when I first confronted this methodology, I was overwhelmed with abstraction; I longed for superfluous examples. But over time this reflex died away, and I found I could do abstract mathematics more efficiently, because I was no longer distracted with the irrelevant. (For more on this, see "Why Johnny Can't Understand", by Edsger Dijkstra. Available at

http://www.cs.utexas.edu/users/EWD/ewd09xx/EWD991.PDF .)

There are many social reasons why these methodologies do not and may never have widespread acceptance in science —not least of which is the emphasis most sciences place on "state of the art" results; the "qua" rather than the "quo modo" —, but I am convinced that if sciences are to mature, they must embrace such methodologies.

Santa Cruz, 23–24 February 2005

Jeremy Weissmann
11260 Overland Ave. #21A
Culver City, CA 90230
USA

jeremy@mathmeth.com