Common Lisp Statistics

Using History to design better data analysis environments

Anthony (Tony) Rossini

Group Head, Modeling and Simulation
Novartis Pharma AG, Switzerland

Affiliate Assoc Prof, Biomedical and Health Informatics
University of Washington, USA

Rice, Mar 2009

Outline

Goals for this Talk

(define, strategic approach, justify)

- To describe the concept of computable and executable statistics, placing it in a historical context.
- To demonstrate that a research program implemented through simple steps can increase the efficiency of statistical computing approaches by clearly describing both:
  - numerical characteristics of procedures,
  - statistical concepts driving them.
- To justify that the approach is worthwhile and represents a staged effort towards increased use of best practices.

(unfortunately, the last is still incomplete)

Historical Computing Languages

- FORTRAN : FORmula TRANslator. Original numerical computing language, designed for clean implementation of numerical algorithms
- LISP : LISt Processor. Associated with symbolic manipulation, AI, and knowledge approaches

They represent the 2 generalized needs of statistical computing, which could be summarized as

- algorithms/numerics,
- elicitation, communication, and generation of knowledge (“data analysis”)

Statistical Computing Environments

Past:

- SPSS / BMDP / SAS
- S (S, S-PLUS, R)
- LispStat (XLispStat, ViSta, ARC , CommonLispStat) ; QUAIL
- XGobi (Orca / GGobi / Statistical Reality Engine)
- MiniTab
- Stata
- DataDesk
- Augsburg Impressionist series (MANET,
- Excel

many others...

How many are left?

- R
- SAS
- SPSS
- Stata
- Minitab
- very few others...

"R is the Microsoft of the statistical computing world" – anonymous.

Selection Pressure

- the R user population is growing rapidly, fueled by critical mass, quality, and value
- R is a great system for applied data analysis
- R is not such a great system for research into statistical computing (backwards compatibility, inertia due to user population)

There is a need for alternative experiments for developing new approaches/ideas/concepts.

Philosophically, why Common Lisp?

Philosophically:

- Lisp can cleanly present computational intentions, both symbolically and numerically.
- Semantics and context are important: well supported by Lisp paradigms.
- Lisp's parentheses describe singular, multi-scale, complete thoughts.

Technically, why Common Lisp?

- interactive COMPILED language ("R with a compiler")
- CLOS is R's S4 object system "done right".
- clean semantics: modality, typing, can be expressed the way one wants it.
- programs are data, data are programs, leading to
- Most modern computing tools available (XML, WWW technologies)
- "executable XML"

Common Lisp is very close in usage to how people currently use R (mostly interactive, some batch, and a wish for compilation efficiency).

Desire: Semantics and Statistics

- The semantic web (content which is self-descriptive) is an interesting and potentially useful idea.
- Biological informatics support (GO, Entrez) has allowed for precise definitions of concepts in biology.
- It is a shame that a field like statistics, requiring such precision, has less than an imprecise and temporally instable field such as biology...

How can we express statistical work (research, applied work) which is both human and computer readable (perhaps subject to transformations first)?

Can we compute with them?

3 Examples:

- Research
- Consulting
- Reimplementation

Consider whether one can “compute” with the information given?

Example 1: Theory...

Let $f(x;\theta)$ describe the likelihood of XX under the following assumptions.

1. assumption-1
2. assumption-2

Then if we use the following algorithm:

1. step-1
2. step-2

then $\hat{\theta}$ should be $N(0, \hat{\sigma}^2)$ with the following characteristics...

Can we compute, using this description?

Given the information at hand:

- we ought to have a framework for initial coding for the actual simulations (test-first!)
- the implementation is somewhat clear
- We should ask: what theorems have similar assumptions?
- We should ask: what theorems have similar conclusions but different assumptions?

Realizing Theory

```
(define-theorem my-proposed-theorem
   (:theorem-type '(distribution-properties
                    frequentist
                    likelihood))
   (:assumes '(assumption-1 assumption-2))
   (:likelihood-form
      (defun likelihood (data theta gamma)
        (exponential-family theta gamma)))
   (:compute-by
      '(progn
         (compute-starting-values thetahat gammahat)
         (until (convergence)
           (setf convergence
                 (or (step-1 thetahat)
                     (step-2 gammahat))))))
   (:claim (assert
             (and (equal-distribution thetahat 'normal)
                  (equal-distribution gammahat 'normal))))
```

It would be nice to have

```
(theorem-veracity 'my-proposed-theorem)
```

and why not...?

```
(when (theorem-veracity
            'my-proposed-theorem)
   (write-paper 'my-proposed-theorem
                :style :JASA
                :output-format
                      '(LaTeX MSWord)))
```

Comments

- The general problem is very difficult
- Some progress has been made in small areas of basic statistics: currently working on linear regression (LS-based, Normal-bayesian) and the T-test.
- Areas targetted for medium-term future: resampling methods and similar algorithms.

Example 2: Practice...

The dataset comes from a series of clinical trials. We model the primary endpoint, “relief”, as a binary random variable. There is a random trial effect on relief as well as severity due to differences in recruitment and inclusion/exclusion criteria.

Can we compute, using this description?

- With a real such description, it is clear what some of the potential models might be for this dataset
- It should be clear how to start thinking of a data dictionary for this problem.

Can we compute?

```
(dataset-metadata paper-1
  :context 'clinical-trials
  :variables '((relief :model-type dependent
                       :distribution binary)
               (trial  :model-type independent
                       :distribution categorical)
               (disease-severity))
  :metadata '(inclusion-criteria
              exclusion-criteria
              recruitment-rate))
(propose-analysis paper-1)
   ; => '(tables
   ;       (logistic regression))
```

Example 3: The Round-trip...

The first examples describe "ideas → code"
Consider the last time you read someone else's implementation of a statistical procedure (i.e. R package code). When you read the code, could you see:

- the assumptions used?
- the algorithm implemented?
- practical guidance for when you might select the algorithm over others?
- practical guidance for when you might select the implementation over others?

These are usually components of any reasonable journal article. *(Q: have you actually read an R package that wasn't yours?)*

Exercise left to the reader!

(aside: I have been looking at the **stats** and **lme4** packages recently – *for me*, very clear numerically, much less so statistically)

Literate Statistical Practice.

1. Literate Programming applied to data analysis (Rossini, 1997/2001)
2. among the most annoying techniques to integrate into work-flow if one is not perfectly methodological.
3. Some tools:
   - ESS: supports interactive creation of literate programs.
   - Sweave: tool which exemplifies reporting context; odfWeave primarily simplifies reporting.
   - Roxygen: primarily supports a literate programming documentation style, not a literate data analysis programming style.
4. ROI demonstrated in specialized cases: BioConductor.
5. usually done after the fact (final step of work-flow) as a documentation/computational reproducibility technique, rarely integrated into work-flow.

Many contributors: Knuth, Claerbout, Carey, de Leeuw, Leisch, Gentleman, Temple-Lang, ...

Literate Programming

Why isn't it enough for Data Analysis?

Only 2 contexts: (executable) code and documentation. Fine for application programming, but for data analysis, we could benefit from:

- classification of statistical procedures
- descriptions of assumptions
- pragmatic recommendations
- inheritance of structure through the work-flow of a statistical methodology or data analysis project
- datasets and metadata

Concept: ontologies describing mathematical assumptions, applications of methods, work-flow, and statistical data structures can enable machine communication.
(i.e. informatics framework ala biology)

Communication in Statistical Practice

. . . is essential for . . .

- finding
- explanations
- agreement
- receiving information

"machine-readable" communication/computation lets the computer help

Semantic Web is about "machine-enabled computability".

Semantics

One definition: description and context

Interoperability is the key, with respect to

- "Finding things"
- Applications and activities with related functionality
  - moving information from one state to another (paper, journal article, computer program)
  - computer programs which implement solutions to similar tasks

Statistical Practice is somewhat restricted

...but in a good sense, enabling potential for semantics...

There is a restrictable set of intended actions for what can be done – the critical goal is to be able to make a difference by accelerating activities that should be "computable":

- restricted natural language processing
- mathematical translation
- common description of activities for simpler programming/data analysis (S approach to objects and methods)

R is a good basic start (model formulation approach, simple "programming with data" paradigm); we should see if we can do better!

Computable and Executable Statistics requires

- approaches to describe data and metadata ("data")
  - semantic WWW
  - metadata management and integration, driving
  - data integration
- approaches to describe data analysis methods ("models")
  - quantitatively: many ontologies (AMS, etc), few meeting statistical needs.
  - many substantive fields have implementations (bioinformatics, etc) but not well focused.
- approaches to describe the specific form of interaction ("instances of models")
  - Original idea behind "Literate Statistical Analysis".
  - That idea is suboptimal, more structure needed (not necessarily built upon existing...).

Interactive Programming

Everything goes back to being Lisp-like

- Interactive programming (as originating with Lisp): works extremely well for data analysis (Lisp being the original "programming with data" language).
- Theories/methods for how to do this are reflected in styles for using R.

Lisp

Lisp (LISt Processor) is different than most high-level computing languages, and is very old (1956). Lisp is built on lists of things which are evaluatable.

```
(functionName data1 data2 data3)
```

or "quoted":

```
'(functionName data1 data2 data3)
```

which is shorthand for

```
(list functionName data1 data2 data3)
```

The difference is important – lists of data (the second/third) are not (yet?!) functions applied to (unencapsulated lists of) data (the first).

Features

- Data and Functions semantically the same
- Natural interactive use through functional programming with side effects
- Batch is a simplification of interactive – not a special mode!

Representation: XML and Lisp

executing your data

Many people are familiar with XML:

```
<name phone="+41793674557">Tony Rossini</name>
```

which is shorter in Lisp:

```
(name "Tony Rossini" :phone "+41613674557")
```

- Lisp “parens”, universally hated by unbelievers, are wonderful for denoting when a “concept is complete”.
- Why can’t your data self-execute?

Numerics with Lisp

- addition of rational numbers and arithmetic
- example for mean

```
(defun mean (x)
   (checktype x 'vector-like)
   (/ (loop for i from 0 to (- (nelts *x*) 1)
  summing (vref *x* i))
        (nelts *x*)))
```

- example for variance

```
(defun variance (x)
  (let ((meanx (mean x))
(nm1 (1- (nelts x))))
     (/ (loop for i from 0 to nm1
    summing (power (- (vref *x* i) meanx) 2)
          nm1))))
```

- But through macros, `(vref *x* i)` could be `#V(X[i])` or your favorite syntax.

Common Lisp Statistics 1

- Originally based on LispStat (reusability)
- Re-factored structure (some numerics worked with a 1990-era code base).
- Current activities:
  1. numerics redone using CFFI-based BLAS/LAPLACK (cl-blapack)
  2. matrix interface based on MatLisp
  3. starting design of a user interface system (interfaces, visuals).
  4. general framework for model specification (regression, likelihood, ODEs)
  5. general framework for algorithm specification (bootstrap, MLE, algorithmic data anaylsis methods).

Common Lisp Statistics 2

- Implemented using SBCL. Contributed fixes for Clozure/OpenMCL. Goal to target CLISP
- Supports LispStat prototype object system
- Package-based design – only use the components you need, or the components whose API you like.

Outlook

- Semantics and Computability have captured a great deal of attention in the informatics and business computing R&D worlds
- Statistically-driven Decision Making and Knowledge Discovery is, with high likelihood, the next challenging stage after data integration.
- Statistical practice (theory and application) can be enhanced, made more efficient, providing increased benefit to organizations and groups using appropriate methods.
- Lisp as a language, shares characteristics of both Latin (difficult dead language useful for classical training) and German (difficult living language useful for general life). Of course, for some people, they are not difficult.

The research program described in this talk is currently driving the design of CommonLisp Stat, which leverages concepts and approaches from the dead and moribund LispStat project.

- `http://repo.or.cz/w/CommonLispStat.git/`
- `http://www.github.com/blindglobe/`

Final Comment

- In the Pharma industry, it is all about getting the right drugs to the patient faster. Data analysis systems seriously impact this process, being potentially an impediment or an accelerator.
  - Information technologies can increase the efficiency of statistical practice, though innovation change management must be taking into account. (i.e. Statistical practice, while considered by some an “art form”, can benefit from industrialization).
  - Lisp’s features match the basic requirements we need (dichotomy: programs as data, data as programs). Sales pitch, though...
  - Outlook: Lots of work and experimentation to do!
- Gratuitous Advert: We are hiring, have student internships (undergrad, grad students), and a visiting faculty program. Talk with me if possibly interested.