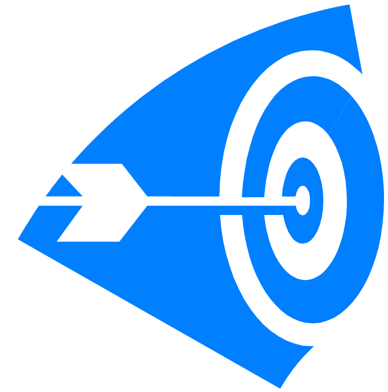
R Modules for Accurate and Reliable Statistical Computing

Micah Altman
(Harvard University)
Jeff Gill
(University of California, Davis)
Michael P. McDonald
(George Mason University; Brookings Institution)

Accurate Statistical Computing

- Why be concerned with accuracy?
  - Bugs
  - Inaccuracies
  - Too little entropy
  - All optimization is local
- What can be done?
  - Numerical Benchmarks
  - Entropy Collection
  - Global optimality tests
  - Sensitivity Analysis
  - Universal Numeric Fingerprints



Statistical Software Has Bugs*

- Estimation bugs (from a survey in 2002):
  - Gauss (ML 4.24): t-statistics for maximum likelihood estimations were half the correct values.
  - SAS (SAS 7.0): produced incorrect results for regressions involving variables with long names, performs exponentiation incorrectly, and commits other statistical errors.
  - SPSS (SPSS 8.01) calculated t-tests incorrectly, and incorrectly dropped cases from crosstabs.
- Data Transfer Bugs (from a survey of 10 packages)
  - Silent truncation
  - Dropped observations
  - Dropped variables
  - Format transformation
  - Rounding errors

* All software has bugs

Correct Software can be Inaccurate

- Correct programs can produce inaccurate results
- Computer arithmetic is subject to rounding error
- *Overflow* occurs when an arithmetic operation yields a result too big for the current storage type
  - *Underflow* occurs when an operation produces a result too small to be represented
  - *Rounding* occurs when a result cannot be precisely represented
  - *Special* values may result from ill-defined operations that do not yield real numbers
  - ***Often, these errors are processed silently.***
- Accumulated errors can dramatically affect estimates, inferences, e.g.:

$$\sqrt{\frac{\sum (x - \bar{x})^2}{n}}$$

Inaccuracies in Stdevp in Microsoft Excel

Digits	*2*	*8*	*9*	*10*	*15*
	1	10000001	100000001	1000000001	100000000000001
	2	10000002	100000002	1000000002	100000000000002
	1	10000001	100000001	1000000001	100000000000001
Values	2	10000002	100000002	1000000002	100000000000002
	1	10000001	100000001	1000000001	100000000000001
	2	10000002	100000002	1000000002	100000000000002
	1	10000001	100000001	1000000001	100000000000001
	2	10000002	100000002	1000000002	100000000000002
	1	10000001	100000001	1000000001	100000000000001
	2	10000002	100000002	1000000002	100000000000002
SD	**0.5**	**0.51**	**0.00**	**12.80**	**1186328.32**

Easy Inaccuracy in R

```
# Three formulas for the standard deviation of the population

> sdp.formula1 <- function(x) { n = length(x); sqrt(n * sum(x^2) - sum(x)^2)/n }
> sdp.formula2 <- function(x) { sum(sqrt((x - sum(x)/length(x))^2))/length(x) }
> sdp.formula3 <- function(x) {  sqrt(var(x) * (length(x) - 1)/length(x)) }

> dat = testMat(50)
> print(rbind(sapply(dat, sdp.formula1), sapply(dat, sdp.formula2), sapply(dat,
sdp.formula3)), digits = 3)

       3   5    7    9   11  13         15       17  19
[1,] 0.5 0.5 0.48 NaN NaN 265271.1 4.43e+07 NaN 4.31e+11
[2,] 0.5 0.5 0.50 0.5 0.5 0.5      0.5      0   0
[3,] 0.5 0.5 0.50 0.5 0.5 0.5      0.5      0   0
```


Random Numbers Aren't

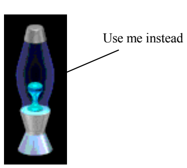
- Pseudo Random number generators are assumed
  - Deterministic
  - **Meant to be seeded with true random values**
  - Generate sequences of fixed length
- Period puts (theoretical) limits on size of sample, before correlation may occur among sub sequences

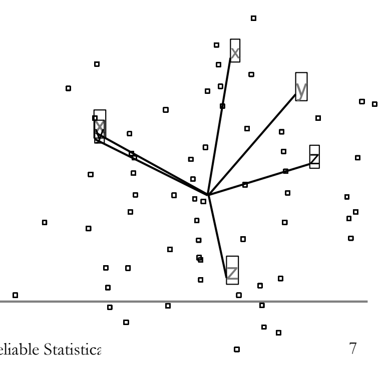
A basic linear congruential generator

$$X_{t+1} = \left(aX_t * + b\right) \bmod m$$

What can go wrong with PRNGS?

- Seed isn't chosen randomly.
- Too many draws.
- Used for t-dimensional point for t large.
- Draws do not follow a uniform distribution.
- Hidden structure to supposed randomness.
- We need more entropy!



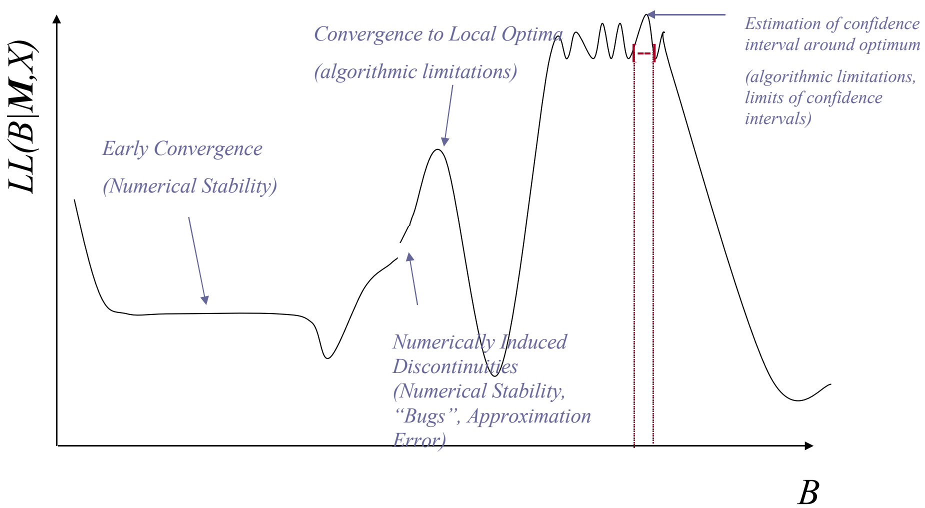


Easy Optimization

- Applications of maximum likelihood, nonlinear least squares, etc implicitly assume:
- There is a single global optimum
- We'll find it.
- Local optima, if they exist, are substantively unimportant

Computational Threats to Inference*

Accurate Computing: Why be concerned?



* All optimization is local

Statistical Benchmarks

Accurate Computing: Benchmark

- Statistical benchmark: feed the computer a set of difficult problems for which you *know* the right answer
- If the answers given back are accurate, you can have more confidence

Statistical Benchmarks from Accuracy

- Tables of basic distributions
- Supplements NIST StrD, Diehard, TestUO1

```
# compute log-relative-error (LRE) of  qt() results, compared to  correct values
data(ttst)
lrq = LRE(qt(ttst$p,ttst$df),ttst$invt)

# if there are low LRE's avoid qt() in those areas
table(trunc(lrq))

-Inf   3   4   5     6      7     8     9   Inf
   2   1  17  1558 5143   650    40     7    34

# can use LRE to explore stability of inverse functions
> p.rand=runif(100000)
> df=trunc(runif(10000,min=1,max=200))
> p.rand=runif(100000)
> df.rand=trunc(runif(100000,min=1,max=200))
> table(trunc(LRE(p.rand,qt(pt(p.rand,df.rand),df.rand) )))
```

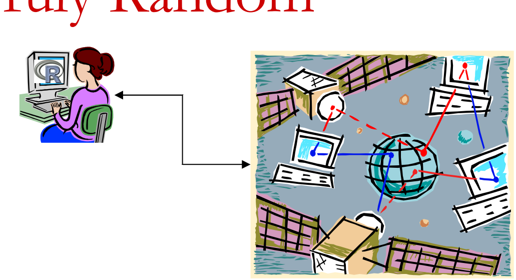

Tests of Global Optimality

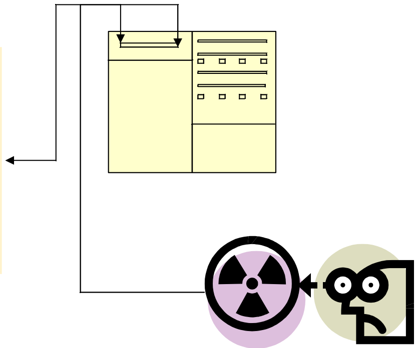
- Count Basins of Attraction for random starting values. Turing; Starr (1979); Finch, Mendell, and Thode (1989)
- Take likelihood at random samples of parameter space, de Haan (1981); Veall (1989).
  - Choose, *n*, samples for the parameter vector using a uniform density, evaluate likelihood L()
  - (1-p) level confidence interval for global max:

$$\left[L_{\max}, L_{\max} + \frac{L_{\max} - L_{2nd\max}}{p^{-1/\sqrt{n}} - 1} \right]$$

- *No guarantees, but acts as a sanity check*

Truly Random





Use entropy gathered from outside sources:

- Local keystrokes and hardware interrupts
- Radioactive decay at Fermilab

(Lead underwear optional)

resetSeed()
Set PRNG seed using true random value
runift()
True random variates, from entropy pool (slow)
runifs()
PRNG sequences, periodically reseeded

Sensitivity Analysis

- Replication on multiple platforms
- *Sensitivity to PRNG choice*
- Sensitivity to choice of optimization algorithm
- *Sensitivity to data perturbations*

Perturbations of Data In Statistical Context

- Cook [1986], Laurent & Cook [1993]
  If L and ω well behaved…
- Straightforward mapping between perturbation of data and perturbation of *model*
- Small normally-distributed noise added to data → small shift to *L*
- Cook defines *worst case* likelihood distance:

$$LD(\omega) = 2\left[L(\hat{\theta}) - L(\hat{\theta}_\omega)\right]$$

Can be interpreted in terms of

$$\left\{\theta \mid 2\left[\mathrm{L}(\hat{\theta}) - \mathrm{L}(\theta)\right] < \chi^2_\alpha(p)\right\}$$

Data Perturbations Interpreted in Other Frameworks

- Beaton, Rubin & Baron [1976]; Gill, et. al [1981]; Chaitin-Chatelin, F, and Traviesas-Caasan [2004]
  - Perturbation in data as sensitivity test for computational problems
- Belsley [1991]; Hendrickx J, Belzer B, te Grotenhuis M, Lammers J (2004)
  - Perturbation/permutation of data as collinearity diagnostic

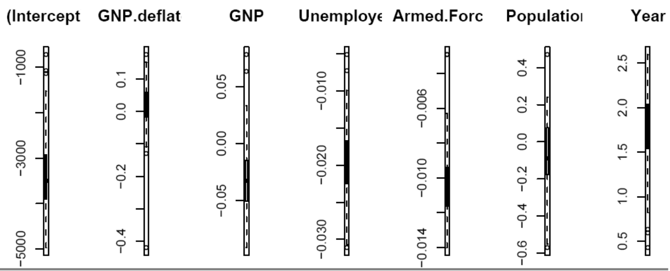
Bottom Line:
If the model is sensitive to a little noise, beware!

Computational Sensitivity Analysis

1. Choose noise (optional)
   - Normal
   - Uniform
   - Repeated Samples
   - Truncated
   - +/- Epsilon
   - Permutation
   - …
2. Add noise
3. Analyze
4. Repeat
5. Summarize

```
> plongley = sensitivity(longley, lm, Employed ~ .)
> print(summary(plongley), digits = 4)

Sensitivity of coefficients to perturbations:
                   mean     stdev        min       2.5%      97.5%        max
(Intercept)  -3.284e+03 9.142e+02 -4.972e+03 -4.729e+03 -1.089e+03 -7.201e+02
GNP.deflator  1.129e-02 8.981e-02 -4.202e-01 -1.255e-01  1.484e-01  1.751e-01
GNP          -2.960e-02 3.514e-02 -9.150e-02 -8.951e-02  5.671e-02  7.823e-02
Unemployed   -1.953e-02 5.265e-03 -3.113e-02 -3.011e-02 -7.942e-03 -5.133e-03
Armed.Forces -1.038e-02 2.308e-03 -1.402e-02 -1.387e-02 -3.675e-03 -2.910e-03
Population   -7.133e-02 2.061e-01 -5.716e-01 -5.462e-01  2.365e-01  4.703e-01
Year          1.728e+00 4.661e-01  4.262e-01  5.987e-01  2.459e+00  2.595e+00
```



Sensitivity Analysis With Zelig

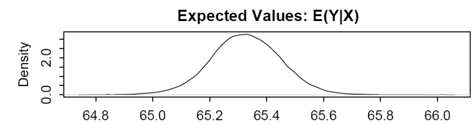
- Zelig provides
  - Uniform syntax to models
  - Easy predictive simulation
- Zelig + Accuracy
  - Easy to analyze sensitivity of predicted values

```
> zelig.out = zelig(Employed ~ GNP.deflator + GNP + Unemployed +
+     Armed.Forces + Population + Year, "ls", longley)
> perturb.zelig.out = sensitivityZelig(zelig.out)

> setx.out = setx(perturb.zelig.out, Year = 1955)
> sim.perturb.zelig.out = psim(perturb.zelig.out, setx.out)

> plot(sim.perturb.zelig.out)

**** 30  COMBINED perturbation simulations
```



Universal Numeric Fingerprints

$$\begin{pmatrix} 1 & 4 & 4 & 21 & \cdots & 121 \\ 1 & 2 & 2 & 91 & \cdots & 212 \\ 1 & 6 & 2 & 12 & \cdots & 204 \\ 1 & 9 & 4 & 52 & \cdots & 311 \\ 0 & 3 & 2 & 23 & \cdots & 92 \\ 0 & 2 & 5 & 91 & \cdots & 212 \\ 0 & 5 & 8 & 91 & \cdots & 91 \\ 1 & 9 & 1 & 72 & \cdots & 104 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 2 & 2 & 91 & \cdots & 212 \end{pmatrix} \Longrightarrow \text{ZNQRI14053UZq389x0Bffg?}$$

```
> library(UNF)
> v = 1:100/10 + 0.0111
> print(unf(v, ndigits = 7))
[1]"UNF:4:7,128:6kK46s059g5dswiRGBM
7yVvo3gwyBVvuBzioK/df72o="
> summary(unf(longley))
[1]"UNF:4:7zq5Q8/mP7z3m2E+mwoOJndVM
8flQmmbuHvvqDK910E="
```

- Same UNF regardless of
  - hardware
  - operating system
  - statistical software,database, or spreadsheet software.
- UNF's combine:
  - generalized rounding (dessication)
  - normalization (canonicalization)
  - fingerprinting (cryptographic hash, e.g. SHA256)
  - presentation (base64)
- **UNF's available for R, Stata, SAS, and standalone use**

Trust, but verify…

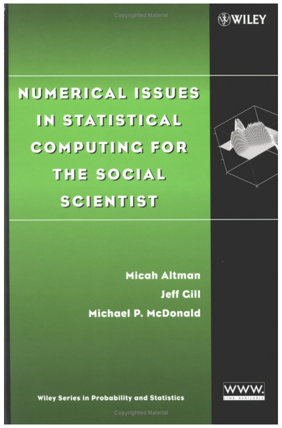
Verify:

1. *Simulations behave properly with true random samples*
2. *Estimated quantities of interest are not sensitive to noise*
3. *Optimization not sensitive to starting*
4. *Reformatting data did not alter it*

Don't Panic*: Most results remain robust.

*☺

Resources



Software

"accuracy" and "UNF" are on CRAN now!

Books

- *Numerical Issues in Statistical Computing …*
  Altman, Gill, McDonald (2003)
- *Elements of Statistical Computation*
  James E Gentle *(2002)*
  *(*And the rest of the computational statistics series)
- *Numerical Methods in Economics*
  Kenneth L. Judd (1998)

Books, journals, mailing lists, software:

<*http://www.hmdc.harvard.edu/numerical_issues/*>