Statistical Computing with **R**

Eric Slud, Math. Dept., UMCP

October 21, 2009

Overview of Course

This course was originally developed jointly with Benjamin Kedem and Paul Smith. It consists of modules as indicated on the Course Syllabus. These fall roughly into three main headings:

(A). **R** (& **SAS**) language elements and functionality, including computer-science ideas;

(B). Numerical analysis ideas and implementation of statistical algorithms, primarily in **R**; and

(C). Data analysis and statistical applications of (A)-(B).

The object of the course is to reach a point where students have some facility in generating statistically meaningful models and outputs. Wherever possible, the use of **R** and numerical-analysis concepts is illustrated in the context of analysis of real or simulated data. The assigned homework problems will have the same flavor.

The course formerly introduced **Splus**, where now we emphasize the use of **R**. The syntax is very much the same for the two packages, but **R** costs nothing and by now has much greater capabilities. Also, in past terms **SAS** has been introduced primarily in the context of linear and generalized-linear models, to contrast its treatment of those models with the treatment in **R**. Students in this course have often had a separate and more detailed introduction to **SAS** in some other course, so in the present term we will

not present details about **SAS**, in order to leave time for interesting data-analytic topics such as Markov Chain Monte Carlo (`MCMC`) and multi-level modeling in **R**.

Various public datasets will be made available for illustration, homework problems and data analysis projects, as indicated on the course web-page.

The contents of these notes, not all of which are posted currently, and which will be augmented as the term progresses, are:

1. `Introduction to R`
   Unix and R preliminaries, R language basics, inputting data, lists and data-frames, factors, functions.
2. `Random Number Generation & Simulation`
   Pseudo-random number generators, shuffling, goodness of fit testing.
3. `Graphics`
4. `Simulation Speedup Methods`
5. `Numerical Maximization & Root-finding`
   (respectively for log-likelihoods and estimating equations)
6. `Commands for Subsetting`
   Manipulating Arrays and Data Frames
7. `Spline Smoothing Methods`
8. `EM Algorithm`
9. `The Bootstrap Idea`
10. `Markov Chain Monte Carlo`
    Metropolis and Gibbs Sampling Algorithms
    Convergence Diagnostics for MCMC
    Bayesian Data Analysis applications using `WinBugs`
11. `Multi-level Model Data Analysis`
    Linear and Generalized Linear Model Fitting and Interpretation

A few Exercises are contained in these notes, but all formal Homework assignments are posted separately in the course web-page Homework directory.

3 Introductory Graphics

We discussed in class the commands **plot**, **points**, **lines**, **legend**, along with the graphical parameters **type** (= "p" points, "n" none, "l" lines, "b" both), **xlab**, **ylab**, **main**, **xlim**, **ylim**, **pch**, **lty**. These and lots more can be studied by invoking the help window on "par" once the **R** help is started.

An extended example, for you to repeat and study, is given in the Rlogs directory of the course web-page

http://www.math.umd.edu/∼evs/s705/Rlogs

as *Rlog2.txt*.

4 Simulation Speed-up Methods

We have already discussed the mechanics of (Uniform) pseudo-random number generation. Most interesting applications of Simulation or the 'Monte Carlo Method' in applied probability and statistics concern evaluation of expectations of combinations of random variables and stochastic-process trajectories

$$E\,(\ f((Z(t),\, t>0),\ Y_1,\ Y_2,\ \ldots)\ ) \tag{1}$$

which are (much) too complicated to evaluate or approximate analytically. The function f will often be an indicator for occurrence of a particular event within a single realization of a data structure. In useful statistical and probabilistic simulations, there is usually need to simulate complicated data structures or large batches with a view to tabulating relative frequencies of occurrence, sometimes of relatively rare events such as those associated with type I and type II errors in statistical hypothesis tests. This can be slow, not only because of the need for looping in R, but because of complicated dependencies among data or nontrivial analyses for each simulated batch of data.

There are several well-recognized methods for speeding up generation of random data, especially with respect to specific purposes involving uncommon events. Two good general references (the first of which is much longer and more detailed, and the second of which will serve us later as a reference

for more elaborate Monte Carlo methods) are:

Devroye, L., **Non-Uniform Random Variate Generation**. Springer, 1986.

Robert, C. & Casella, G. , **Monte-Carlo Statistical Methods**. Springer, 1999.

There are also many texts on 'Discrete-Event Simulation', concerning the methodology of designing and programming (in some high-level languages especially suited to the purpose) simulations which arise in queueing, inventory, finance, and operations research generally. One major author who has written several well-regarded texts is George Fishman. Our resident expert on this at UMCP is Dr. Michael Fu of BMGT.

We concentrate our attention on defining and giving brief examples of three general simulation speed-up methods:

(1). Rejection Methods

(2). Control-Variate Methods

(3). Importance-sampling Methods

Of these methods, the first and third involve directly simulating something different from data with the desired behavior: we simulate something easier, and then discard or modify certain cases systematically, using a theoretical idea to justify that the expectation in formula (1) is obtained from what is actually simulated. The second method is a pure variance-reducer: it is essentially a regression of a difficult-to-obtain quantity (the integrand in Equation (1)) on one or several related quantities *whose corresponding expectations are known* from some other (typically theoretical) considerations.

4.1 *Rejection Methods*

An example of this method — generation of Normal deviates from Cacuhy — has already been covered in class. Here we give an abstract description. Suppose that we know how to generate a random vector W from a specified probability density g and that we define a modified variable Y as being

equal to W when the latter falls in a pre-specified region D (discarding or 'rejecting' any simulated values $W \notin D$. Then it is easy to calculate that the density of Y is

$$P(Y \in (y,\, y+dy)) \;=\; P(W \in (y,\, y+dy))\, I_D(y) \,\Big/\, \int_D f_W(w)\, dw$$

or

$$f_Y(y) \;=\; f_W(y)\, I_D(y) \,\Big/\, \int_D f_W(w)\, dw$$

The case considered in class is where $W = (X, U)$, with X an easily generated type of random variable and U an independent Uniform[0,1], and where $D = \{(x,u) :\; 0 \le u \le c\, f(x)/f_X(x)\}$ for an arbitrary positive constant c. Here the desired density to simulate is $f(x)$, and to make the method work it must be *assumed known* that for all x, $c\, f(x) \le f_X(x)$. It is then easy to check [Devroye 1986, pp. 40-42] that the first component of Y defined as above has density f.

As another particular example, suppose it is desired to simulate a set of ten random variables $Z_1, \ldots, Z_{10}$ which behave like independent standard-normals subject to the condition that

$$\max_k Z_k \;-\; \min_k Z_k \ge 3\,|\overline{Z}|$$

This is not a particularly natural condition to impose; but because it is not an improbable, there is no doubt that the most efficient way to simulate it repeatedly is to generate batches **rnorm(10)** of standard-normal independent variates and to keep only those batches which satisfy the condition.

4.2 *Control Variates*

In general, we seek in Monte-Carlo studies to approximate $E(Z)$, where $Z = f(X)$ is a random variable which may summarize some complicated statistical experiment. Suppose there is a variable Y defined from the same experiment which is closely related to (i.e., highly correlated with) Z and for which we happen either to know $E(Y) = m$, or to be able to find it with much less effort than we could hope to use in simulating $E(Z)$ directly. Then the method is to simulate pairs (Y_i, Z_i), $i = 1, \ldots, N$ and from these 'data' we 'fit' the linear regression model

$$Z_i = a + b(Y_i - m) + e_i \quad , \qquad E(e_i) = E(e_i Y_i) = 0$$

via least-squares, recalling that $a = E(Z)$ is what we are after. Once we have assumed that Y_i has the exactly known expectation m, this is an 'orthogonal regression design' in the sense that the design column consisting of all 1's is uncorrelated with (and therefore in large samples approximately orthogonal to) the design column $(Y_i - m,\ i = 1, \ldots, N)$. From well known results of linear regression theory, the variance of $\hat{a}$ conditionally given $(Y_i - m,\ i = 1, \ldots, N)$ (and therefore also unconditionally) is

$$\frac{1}{N}\,\mathrm{Var}(e_1) \;=\; \frac{1}{N}\left(\mathrm{Var}(Z_1) - b^2\mathrm{Var}(Y_1 - m)\right) \;=\; \frac{1}{N}\,\mathrm{Var}(Z_1)\,(1 - \mathrm{Corr}^2(Y_1, Z_1))$$

So the benefit of this approach is to decrease the width of simulation-based confidence intervals for $E(Z)$ by a factor $\sqrt{1 - (\mathrm{Corr}(Y, Z))^2}$. The benefit can sometimes be amplified by regressing on more than one control variate, as we indicate in the next Example.

4.2.1 *A Toy Clinical Trial Case-Study Example*

We describe an example in which the variable Z of interest is the 'total time on test' of all patients in a small clinical trial conducted as follows. Sixty (60) patients are 'accrued' (i.e. brought) into the trial at times U_i which are uniformly distributed within the first year and randomly allocated (by an independent fair coin toss for each patient) to group $Z_i = 1$ or 0. From the time of their entry into the study, the individuals have lifetimes T_i which in group 0 are Exponentially distributed with rate λ_0, and which in group 1 are Exponential lifetimes with rate-parameter $\lambda_0 \cdot e^{\Delta}$. At the end of each of $t = 1,\ 2$ years a statistic is calculated, defined by

$$S_t = \frac{TT(1,t)\,D(0,t) \;-\; TT(0,t)\,D(1,t)}{(TT(0,t) + TT(1,t))^2} \;\cdot\; \sqrt{D(1,t) + D(0,t)}$$

where $TT(z,t)$ denotes the total time on test in group z up to time t, and $D(z,t)$ denotes the total number in group z who have died by time t. Note that in large-sample settings, S_1 is approximately a normal variable (with mean and variance which are easily calculated numerically, no matter what Δ is), and S_1 is approximately standard-normal if $\Delta = 0$.

This statistic S_t is a variant of some commonly used (*weighted logrank*) test statistics commonly used in clinical trials: large positive values indicate

that there are fewer group-1 deaths than would have been expected if both groups have the same survival distribution, i.e., if $\Delta = 0$.

Here is a decision rule for the clinical trial which allows 'early stopping': stop and reject at $t = 1$ if $S_1 \geq 2.4$, otherwise continue to $t = 2$ and reject if $S_2 > 1.7$. (Accept at $t = 2$ if neither of these rejection conditions holds.)

One problem of interest is: for clinical trials like this one, what is the expected total time on test, for various choices of Δ ? (This total time on test is the sum over patients of the number of years each patient is in the trial before the trial stops. It is one measure of clinical trial cost.) Although I have approximate formulas, I do not know how to calculate this (even asymptotically, if 60 is replaced by n going to ∞).

So our Z is taken equal to total time on test. What is a relevant set of control variates ? A simple and natural choice is the pair of variables

$$TT_1 \;=\; TT(0,1) + TT(1,1) \qquad , \qquad TT_2 \;=\; TT(0,2) + TT(1,2)$$

This choice is useful because the expected time on test variables up to the fixed times 1, 2 are easily shown to be given by simple formulas which under the null hypothesis (after an integration by parts) become

$$E(TT_1) \;=\; 60 \int_0^1 (1-t) e^{-\lambda_0 t}\, dt \;=\; \frac{30}{\lambda_0} \left[1 \;-\; \frac{1 - e^{-\lambda_0}}{\lambda_0} \right]$$

$$E(TT_2) \;=\; 60 \int_0^2 \min(2-t, 1) e^{-\lambda_0 t}\, dt \;=\; e^{-\lambda_0}\, E(TT_1) \;+\; 60\, \frac{1 - e^{-\lambda_0}}{\lambda_0}$$

The simulation of such a clinical trial including all relevant statistics, was implemented in an **Splus** function when **Stat 705** was taught in 2005. The defaults, which were used here, included $\lambda_0 = 0.7$ and $\Delta = 0$. The outputted data, from which we can not only calculate the desired expectation two ways but can see the factor by which the contol-variate method reduces variance, is as follows [where a few irrelevant entries have been deleted]:

```
> unlist(TrialSim(0, Nrep=1000))
 Power       ET      VarT   S1sq1   S1sq2   S2sq1   S2sq2
 0.053 0.915158 0.0069185  1.2471  4.7832  1.0317  2.28242
```

```
    TT1mean       TT2mean      TTvar1       TTvar2      TTvar3
0.00067195    0.00021994    0.001327    0.0016275   0.0016275
    TTvar4      Et12Tim1     Et12Tim2      TimEst     VarFac
0.00476656     0.0017799    0.004986    0.9148676     0.2449
```

The successive entries of this list were calculated empirically from the 1000 simulation-replication. Observe first that the relative frequency of overall ('experimentwise') rejection in this example is at 0.053, quite close to the nominal 0.05. Next, the terms **S1sq1** and **S1sq2** are respectively the emipirical mean and variance of the *square* of the supposedly standard-normal S_1 statistic (and so should be 1, 2 respectively). With the parameter settings used, S_1 evidently has much fatter tails than it is supposed to. The corresponding empirical moments for S_2 do not look bad, though.

The quantity of interest to us here was the expected total time-on-test, labelled ET, which we estimate from raw data as 0.915. The empirical variance based on the 1000 simulation-replications was 0.0069185, so we can interpret the quantity ET we desire as having been given within a 95% tolerance interval

$$0.915 \ \pm \ 1.96 \sqrt{\frac{0.00692}{1000}} \ = \ 0.915 \ \pm \ 0.00516$$

The 'control variate' regression relationship

$$T \ = \ a \ + \ b_1 \, (TT_1 - E(TT_1)) \ + \ b_2 \, (TT_2 - E(TT_2)) \ + \ \epsilon$$

had its coefficients estimated through the empirical covariances and correlations given in the output-list above:

$$\begin{pmatrix} \mathrm{Cov}(T, TT_1) \\ \mathrm{Cov}(T, TT_2) \end{pmatrix} \ = \ \begin{pmatrix} \mathrm{Et12Tim1} \\ \mathrm{Et12Tim2} \end{pmatrix} \ = \ 60 \cdot \begin{pmatrix} 0.00178 \\ 0.00499 \end{pmatrix}$$

and

$$\mathrm{Var} \begin{pmatrix} TT_1 \\ TT_2 \end{pmatrix} \ = \ 60^2 \cdot \begin{pmatrix} \mathrm{TTvar1} & \mathrm{TTvar2} \\ \mathrm{TTvar3} & \mathrm{TTvar4} \end{pmatrix} \ = \ 60^2 \cdot \begin{pmatrix} 0.0001327 & 0.001628 \\ 0.001628 & 0.04767 \end{pmatrix}$$

Using fitted coefficients and the observed empirical values 0.00067195 and 0.00021994 for $(TT_1 - E(TT_1)/60$ and $(TT_2 - E(TT_2)/60$, we obtain the control-predictor estimate 0.91487 for ET. This differs from the other estimator by very little, but the factor 0.245 (equal to 1 minus the multiple-correlation in the regression), by which the variance is reduced over that of the naive estimator, is substantial.

4.3 *Importance Sampling*

The idea of Importance Sampling is that, when we are simulating the occurrence of a rather unlikely event, we may be able to simulate instead under a different probability law, and find the expectation of a suitable multiple of the indicator-function of the event of interest. So if we are interested in the occurrence of an event D with respect to a vector $\mathbf{X}$ of variables with a joint density f, and if D is unlikely according to f but not according to a simple related joint density g, then instead of

$$\text{simulating } X_1, \ldots, X_N \sim f \quad , \quad \text{estimating } \hat{P}(D) = N^{-1} \sum_{i=1}^{N} I_D(X_i)$$

we could

$$\text{simulate } Y_1, \ldots, Y_N \sim g \quad , \quad \text{and estimate } \hat{P}(D) = N^{-1} \sum_{j=1}^{N} \frac{f(Y_j)}{g(Y_j)} I_D(Y_j)$$

4.3.1 Example: Power of a Test

Suppose that we plan to have a moderately large ($n = 40$) sample of *Exponential*(λ_0) data-values for which we want to test $\lambda_0 = 1$ versus $\lambda_0 = 0.6$. The Neyman-Pearson test statistic is $S = X_1 + ... + X_{40}$, and we would often use the CLT to approximate the rejection region by

$$\{ \mathbf{X} : \ (S - 40)/\sqrt{40} > 1.645 \}$$

We want to know whether the significance level is really 0.05 for this small a sample-size, whether the related rejection-region $\{(S - 40)/\sqrt{S} > 1.645\}$ has the same significance level, and which of the two regions has the better power. (Normal theory suggests that this power ought to be roughly $1 - \Phi(50.40 - 40 \cdot 5/3)/\sqrt{40 \cdot (5/3)^2} = 0.9386$.)

Note that the events which we are trying to simulate have small probabilities ! We will use $N = 10000$ and explore the gain in accuracy which is possible by importance sampling, when we recognize that we know something about likelihood ratios in the particular problem. Note that f and

g respectively denote the joint densities of 40 exponential variates with different exponential parameters λ, λ', so

$$\frac{f(\mathrm{y},\,\lambda)}{g(\mathrm{y},\,\lambda')} \;=\; \left(\frac{\lambda}{\lambda'}\right)^{40} \exp\left(-(\lambda-\lambda')\sum_i y_i\right)$$

The function used to accomplish the simulation is called **SmExpTst**, and it is also available in the course Splus-data directory. The first input parameter is the value λ' used to generate simulated variates (Y_i with density g), and the second parameter is the value λ for which we want to calculate a rejection tail-probability. (There are two further input parameters: the size *nsiz* of each data-sample, with default value 40, and the number *Nrep* of simulation-replications with default value 10000. A typical simulated result (in this case corresponding to the direct null-hypothesis simulation) is as follows:

```
> SmExpTst(1,1)[1:2]
[1]    0.05890000    0.05543633
```

So for the direct null-hypothesis simulation, we have (for the original rejection region) the slightly above-nominal size 0.059. The second output component is the empirical variance of the rejection indicator (essentially $p(1-p)$ for $p = .0589$), which implies because of the large number of replications that the confidence interval for the actual significance level does lie above the nominal value .95.

The corresponding information for power (based on a direct simulation) is:

```
> SmExpTst(0.6,0.6, Upper=F)[1:2]
[1] 9.4860e-01 4.8763e-02
```

The theoretical prediction of 0.95 turned out to be very accurate: the empirical answer is around 0.948, with confidence interval tolerances around $\pm\, 1.96\sqrt{.04876/10^4} \approx \pm 0.004$.

Now let us simulate these with the importance-sampling approach. The simplest idea for λ' is to use a value for which bare rejection is the most likely occurrence, i.e., to use as rate parameter the value given by the cutoff:

lamsamp = nsiz/cutoff $= 40/50.404 = 0.7936$.

Timing does not change at all, but the accuracy changes dramatically!

```
> SmExpTst(0.7936, 1)[1:2]
[1] 0.0584142 0.0074292
```

The simulation confidence intervals for significance level now have tolerance equal to $\pm 1.96 \cdot \sqrt{.0074/10000} = 0.0016746$, quite a bit more accurate (a factor of about 3) than before. In particular, we now learn that the true significance level under the null hypothesis is about 0.058. Let us check this using the **exact** calculation from **R**:

```
P(gamma(40,1) > 50.40389) = 1 - pgamma(50.40389,40) = 0.05793
```

To see what is going on, we do two more runs, this time showing the numbers of nonzero cases, which is the third output-component of the function **SmExpTst**:

```
> SmExpTst(1,1)
[1]   0.057300   0.054022 573.000000
> SmExpTst(0.7936, 1)
[1] 5.8931e-02 7.4927e-03 4.8560e+03
```

The point is: we had 4856 nonzero things to average in the importance-sampling simulation, but only 573 in the naive case.

Just to round out the illustration, we note that the true power in this example, simulated above by naive relative frequencies to be around 0.949, is in fact 0.9491 according to the true $Gamma(40, .6)$ distribution for the sum-statistic S. However, a much more efficient simulation of the same power (as can be seen by comparing the previously simulated variance of 0.0488 for the power-estimator to the one below, is:

```
> SmExpTst(0.7936, 0.6, Upper=F)
[1] 9.484727e-01 4.542103e-03 5.265000e+03
```

Further small comparisons show that it does not help to choose the importance-sampling λ' still smaller than 0.7936, even though the number of nonzero items to average goes up, because those nonzero things have bigger variance !

Additional simulations to find the actual power of the test-statistic with denominator $\sqrt{S}$ could be done using a completely analogous importance-sampling method, and we leave this as an optional exercise.

For an R log implementing many of these ideas, see `Rlog.ImpSamp` in the Stat 705 Course web-page.

We described in class that for further examples of Importance Sampling in more realistic examples involving multivariate densities, the initial density $f(\mathbf{x})$ can often be 'exponentially tilted' in the direction of a function $h(\mathbf{x})$ when we are interested in restricting attention to empirical averages involving indicators $I_{[h(\mathbf{x}) \geq C]}$ for large C : this means choosing $g(\mathbf{x})$ whenever possible (i.e. when this form is not too difficult to simulate from) of the form

$$g(\mathbf{x}) \;=\; K\, f(\mathbf{x})\, \exp(\lambda\, h(\mathbf{x}))$$

Here K is chosen as a function of $\lambda > 0$ so that g integrates to 1, and λ is generally chosen so that $\int h(\mathbf{x})\, g(\mathbf{x})\, d\mathbf{x} \approx C$.