STAT 798c, HW Problem Set #3, due Monday 3/10/03

The theme of this homework set is **Simulation**, including some of the speedup ideas discussed in class. In addition, we will implement some of the goodness-of-fit checks for simulations which we discussed. Wherever possible, implement your simulation *in parallel* rather than in a for-loop.

(A) Simulate by an **accept-reject method** 1000 random points (X_i, Y_i) uniformly distributed in the triangle $\{(x,y) : x \in (0,1),\ 0<y<\min(3x, \frac{3}{2}(1-x))\}$. Do this as efficiently as you can, parallelized as far as possible. Have Splus calculate how long this run took (for comparison with part (B)). Also provide some exhibit (e.g., a plot or tabulation showing that roughly the right proportion of points fell where they were supposed to.)

(B) Simulate X_i and Y_i conditionally given X_i by a *probability integral transform* method, $i = 1, \ldots, 1000$. Again provide checks to show that you did it correctly, and again try to parallelize. Which is faster, this method or the method you used in (A) ? Express, and justify if possible, a conclusion about which method *if optimally programmed* would be faster in this problem *if optimally programmed* and 10000 random pairs were to be generated.

Consider the problem of testing independence of row and column classifications in the 2×2 contingency table based on $n = 100$ samples. Suppose that the row classifications are A with probability 0.7, and A^c with probability 0.3, and the column classifications are B with probability 0.6 and B^c with probability 0.4. The usual test-statistic for row-column independence is the *Pearson chi-square statistic*

$$X^2 = \frac{(n_{AB} - n_{A+}n_{+B}/100)^2}{n_{A+}n_{+B}/100} + \frac{(n_{AB^c} - n_{A+}n_{+B^c}/100)^2}{n_{A+}n_{+B^c}/100}$$

$$+ \frac{(n_{A^cB} - n_{A^c+}n_{+B}/100)^2}{n_{A^c+}n_{+B}/100} + \frac{(n_{A^cB^c} - n_{A^c+}n_{+B^c}/100)^2}{n_{A^c+}n_{+B^c}/100}$$

where the counts in the four cells of the table are $n_{AB}, n_{A^cB}, n_{AB^c}, n_{A^cB^c}$, and

$$n_{A+} = n_{AB} + n_{AB^c} = 100 - n_{A^c+}, \quad n_{+B} = n_{AB} + n_{A^cB} = 100 - n_{+B^c}$$

(C) The statistic X^2 is supposed to have distribution approximately $\chi^2_1 = Gamma(\frac{1}{2}, \frac{1}{2})$, but there are too many discrete possibilities for the distribution to be calculated exactly. Design and carry out a simulation of 1000 replications, in each of which you generate the multinomial dataset $(n_{AB}, n_{A^cB}, n_{AB^c}, n_{A^cB^c})$ with 100 trials and respectively cell-probabilities $(0.42, 0.28, 0.18, 0.12)$, and produce a graphical exhibit showing how close the approximation is between the actual empirical distribution function of X^2 and the theoretical χ^2_1 distribution function. (*Hint: by using a form of the* **sample** *function in Splus, you can generate all of the multinomial data in parallel at once and avoid for-loops.*)

(D) The simulation you did in (C) provides an estimate of the actual significance level of a nominal $\alpha = 0.05$ chi-square test (which rejects when $X^2 \geq 3.84146$) for the data with specified parameters. Write a function to implement the Monte Carlo estimation of this same actual significance level based on simulating data with 1000 iterations from a multinomial 4-cell distribution with 100 trials and with the cell-probabilities as input parameters, using an importance-sampling estimator. (Your function should replicate the same type of simulation of the answer which you generated in (c) for the input parameters $(0.42, 0.28, 0.18, 0.12)$.) Your function should also estimate the sampling variability of your Monte Carlo estimator for the true significance level. Are there some values of the multinomial cell probabilities for which the importance-sampling estimator provided by your function is much more accurate than the simulation in (C) ?

Hint: look at the function **SmExpTst** *for help in setting up your importance-sampling function.*