CS170A — Mathematical Models & Methods for Computer Science
HW#3 — Randomness and Statistics
Due: 5:00pm Wed February 5, 2003

D. Stott Parker, Blai Bonet, Fotios Konstantinidis
stott@cs.ucla.edu, bonet@cs.ucla.edu, fotios@cs.ucla.edu

1. **Benford's law**
In the file `~cs170ata/3:Statistics/BenfordsLaw.txt` is a description of a phenomenon pointed out by Benford (F. Benford, Proc. Am. Philosophical Soc. *78* (1938), 551-572), who noticed something that is disturbing to many people: *the distribution of leading digits in computed numbers is usually not uniform*.

Specifically, looking at decimal (i.e., base 10) numbers, Benford noticed that that the probability that the leading digit is d is

$$Prob[\text{ leading digit is } d\,] \;=\; \log_{10}(d+1) - \log_{10}(d) \;=\; \log_{10}(1+1/d).$$

Thus:

$$\begin{array}{rcccl} Prob[\text{ leading digit is 1 }] & = & \log_{10}(1+1) & = & .30103\ldots \\ Prob[\text{ leading digit is 2 }] & = & \log_{10}(1+1/2) & = & .17609\ldots \\ \vdots & & \vdots \quad \vdots & & \vdots \quad \vdots \\ Prob[\text{ leading digit is 9 }] & = & \log_{10}(1+1/9) & = & .045757\ldots \end{array}$$

In other words, *the leading digit is 1 about 30 percent of the time!* And the leading digit is 9 only about 5 percent of the time.

This law is actually a powerful data mining tool used by the IRS, and by other agencies, to look for fraud. They check whether the histogram of dollar amounts in files (income tax returns, insurance claims, and so forth) match the logarithmic density function above. If the histogram is too far from logarithmic, human experts can then review them for possible fraud.

Your assignment is to write a program in Matlab or in Maple that does the following:

- reads in the values in a file;
- displays a histogram of the values;
- checks if the histogram matches the logarithmic density, using the χ^2 (chi-square) test.

The chi-square test is a way of checking whether an observed density matches a theoretical density, for simple discrete density functions.

Let there be m possible outcomes in the density, and the theoretical probability of the i-th outcome is p_i. Also suppose N observations (samples) are made, and are put into a histogram h with m entries, so h_i is the number of times outcome i was observed.

Intuitively, the shape of the histogram should look very much like the shape of the density. In other words, then, we should have $h_i \approx N\,p_i$ (approximately).

The *chi-squared statistic of h with $(m-1)$ degrees of freedom* is

$$V \;=\; \sum_{i=1}^{m} \frac{(h_i \;-\; N\,p_i)^2}{N\,p_i}.$$

It measures the difference between the two densities. If the statistic is too large, or too small, then the observed histogram does not match the expected density.

More formally, if the statistic exceeds the 99% thresholds in the χ^2 tables, then there is only a chance of 1 in 100 that the observed histogram values are actually drawn from the theoretical distribution. So there is "99% confidence that the density is not logarithmic".

These χ^2 tables are available in the Maple `stats` package. See the files

`~cs170ata/3:Statistics/ChiSquareTest.mws`
`~cs170ata/3:Statistics/chisquared_prob.m`

for examples and ideas about how to implement this.

Finally: Presidents of the United States generally make their tax returns public, and these have been made available at `http://www.tax.org/THP/Presidential/default.htm` (among other places). Choose 2 tax returns, and check whether they follow Benford's law.

Specifically, make a dataset containing *every dollar amount that is entered on some line in the tax return*. (Do *not* include non-dollar amounts, such as numbers appearing in dates, addresses, etc.) Check whether the resulting dataset's histogram matches a logarithmic density.

2. **Central Limit Theorem**
The **convolution** $f \star f_2$ of two functions f_1 and f_2 is the function

$$(f_1 \star f_2)(x) \quad = \quad \int_{-\infty}^{\infty} f_1(x - t)\, f_2(t)\, dt.$$

In Maple, this can be defined as follows:

```
convolution := (f1,f2) ->
     unapply( int( unapply(f1(x-t)*f2(t),t), -infinity..infinity), x);
```

Notice that the output of `convolution` is a function, since `unapply(E,x)` yields the function `x -> eval(E)`.
First: execute the following Maple code, which produces two plots:

```
g := x -> 1/sqrt(2*Pi) * exp(-x^2 /2);    # normal density (mu=0, sigma=1)
G := x -> int( g(t), t=-infinity..x );    # normal distribution

plot( [g, G], -4..4, color=[blue,red] );  # ("normal" == "Gaussian")

u := x -> Heaviside(x) - Heaviside(x-1);  # uniform density
U := x -> int( u(t), t=-infinity..x );    # Uniform distribution

plot( [u, U], -1..2, color=[blue,red] );
```

An important result of probability theory is this:

> if x_1 and x_2 are independent random values with probability densities f_1 and f_2, respectively, then the sum $x = x_1 + x_2$ has density $f = f_1 \star f_2$.

Examples: if x and y are uniform random values (with probability density u), then $x + y$ is a random value with probability density $(u \star u)$. Also, if x and y and z are normal random values (with probability density g), then $x + y + z$ is a random value with probability density $((g \star g) \star g)$.

Second: make the following plots:

- For $k = 2$, plot the density of the sum of k *uniform random values* (uniform on $[0, 1]$). That is, plot $(u \star u)$.
- For $k = 2, 3$, plot the density of the sum of k *normal random values* (each having mean 0 and variance 1). That is, plot $(g \star g)$ and $((g \star g) \star g)$.

Third: using something like 10,000 random samples:

- For $k = 1, 2, 3, 4$: make a histogram of the sum of k *uniform random values*.
- For $k = 1, 2, 3, 4$: make a histogram of the sum of k *normal random values*.

These histograms should match the plots you made earlier for $k = 2$ and $k = 3$.

3. **Least Squares**
Using either Maple or Matlab, you are supposed to find a least squares fit for specific models on a particular dataset.

In the class directory `~cs170ata/www/` (`http://www.seas.ucla.edu/cs170a/`) you can find a dataset giving 9 attributes of 391 cars called `autos.m` (or `autos.maple` — it is implemented both in Matlab and in Maple for your convenience). For example, the start of `autos.m` looks like this:

```
% Columns of the autos matrix:
% 1 -- MPG
% 2 -- Cylinders
% 3 -- Displacement
% 4 -- Horsepower
% 5 -- Weight
% 6 -- Acceleration
% 7 -- Year
% 8 -- Origin
% 9 -- Make

% MPG Cyl Disp   Hpwr   Wt    Accel Yr Org Make
autos = [ ...
13.0, 8, 360.0, 175.0, 3821., 11.0, 73, 1, 1; ...
14.0, 8, 304.0, 150.0, 3672., 11.5, 73, 1, 1; ...
14.0, 8, 304.0, 150.0, 4257., 15.5, 74, 1, 1; ...
15.0, 6, 258.0, 110.0, 3730., 19.0, 75, 1, 1; ...
15.0, 8, 304.0, 150.0, 3892., 12.5, 72, 1, 1; ...
```

All values in the files are numeric (the last two columns use integer codes).

- Use least squares to find coefficients α and β that yield the best model

$$1/\text{MPG} \;=\; \alpha \cdot \text{horsepower} \;+\; \beta$$

  — that is, find the instance of this formula with minimal squared error.
- Use least squares to find the coefficients α, β, γ, δ that yield the best model

$$1/\text{MPG} \;=\; \alpha \cdot \text{horsepower} \;+\; \beta \cdot \text{displacement} \;+\; \gamma \cdot \text{weight} \;+\; \delta.$$

- Compare the error values of each of the two models above: which model is better? (i.e., which has the lower error?)
- For extra credit, find a model that relates MPG, horsepower, displacement, weight, and acceleration — and that also fits this dataset with lower error than the two models above.