ACM 11: Homework 6

Assigned Wednesday, Nov 12 2008. Due on Wednesday, Nov 19 2008 at noon. 50 pts.

Submission instructions: follow the format of the Mathematica problem set template in the handout section, and submit the notebook file in the format `Firstname_Lastname_2.nb` to `ftp.its.caltech.edu/pub/srbecker/incoming`. Follow the standard instructions for resubmissions.

1. **`Sequence` and `Apply`** (25 points) To complete the next exercise, we need to learn the `Sequence` and `Apply` commands. The latter is pretty straightforward: `Apply[f, expr]` replaces the head of the expression `expr` with `f`; `f @@ expr` is the same as `Apply[f,expr]`. Here are examples using `Apply`:

   i. `Apply[Cos, Sin[x]]` has the value `Cos[x]`

   ii. `Apply[List, f[x,y]]` has the value `{x,y}`

   iii. `f @@ (x+y)` has the value `f[x,y]`

   The `Sequence` command is a bit more esoteric, but can be quite handy. Many Mathematica operators take a sequence of arguments, as opposed to a list of arguments: they are called with the syntax `f[x,y,...]` as opposed to `f[{x,y,...}]`. We already know how to generate lists; `Sequence` is a tool which allows us to generate sequences. The usefulness of `Sequence` lies in the fact that the expressions `f[x,y,z,...]`, `f[Sequence[x,y,z,...]]`, and `f[x, y, Sequence[z, ...]]` and others of the same ilk all have the same value. Basically `Sequence` splices its arguments into the argument sequence of f.

   Here's an example application of `Sequence`: `Plus[Sequence @@ Range[6]]`. As you'll find if you evaluate this in Mathematica , this returns 21, the sum of the numbers between 1 and 6. What happens here is that Mathematica evaluates `Range[6]`, which has the value $\{1, 2, 3, 4, 5, 6\}$, then the head (`List`) of this expression is replaced with `Sequence`, so the entire expression has the value `Plus[Sequence[1,2,3,5,6]] = Plus[1,2,3,4,5,6] = 1+2+3+4+5+6 = 21`. Note that you couldn't simply say `Plus[Range[6]]` (what happens if you do this, and why?) and that the above expression can also be written `Plus[Range[6] /. List -> Sequence]` or simply `Plus @@ Range[6]`.

   It's important to note that some functions have the attribute `HoldAll` which means they don't evaluate their arguments until they are needed in the calculation. This causes an error if you attempt to use a `Sequence` as an argument– since the arguments aren't evaluated immediately, the `Sequence` is not spliced into the arguments, instead the entire `Sequence` appears to be one argument. To check for this attribute, use `??f` (in the same way we have checked for the `Listable` attribute before), and to avoid this problem, wrap your `Sequence` object in `Evaluate`. For more details, see the help on `Evaluate` and `HoldAll`.

   As an example, note that `Product` has the attribute `HoldAll`, so the command
   `Product[Cos[(k[1] + k[2])π], Sequence @@ Table[{k[i],1,10}, {i,1,2}]]`
   will not do what you think it should, but
   `Product[Cos[(k[1] + k[2])π], Evaluate[Sequence @@ Table[{k[i],1,10}, {i,1,2}]]]`
   will. The bottom line is, if `Sequence` isn't working as you think it should, try wrapping it in `Evaluate`.

   (a) i. (5 points) From the above explanation of `Sequence` and `Apply`, explain in words why the value of this expression is what it is (you may need to look in the help for `D`):
   `D[ Sum[x[i]^2, {i,1,5}] // Sqrt, Sequence @@ Table[x[i], {i,1,5}]]`

   ii. (5 points) Use `NSum`, `Sequence` to calculate

$$\sum_{i_1=1}^{10} \sum_{i_2=1}^{i_1} \cdots \sum_{i_7=1}^{i_6} \frac{1}{\sum_{k=1}^{7} i_k^2}.$$

(b) We're going to use `Apply` to create a `ListPlot` of a perturbed Lissajoux curve $f(t,\epsilon) = (\sin(5(t+\epsilon)), \cos(7(t+\epsilon))$ where ϵ is random noise in the range [-.1,.1].

i. (5 points) (Feel free to do all the parts of this problem in one cell. Also, don't display these *long* lists.) Write `f`, a function of t and ϵ which returns a sample of the curve as a list pair, $\{\sin(5(t+\epsilon)), \cos(7(t+\epsilon))\}$.

ii. (3 points) Use `Range` and `RandomReal` to generate θ`samples`, a list of 4000 evenly spaced points in the range $[4000^{-1}, 2\pi]$, and `noise`, a list of 4000 samples of random numbers in the range [-.1,.1]. (To get the θ symbol in Mathematica , type ESC-th-ESC)

iii. (5 points) Now we'll use `Apply` to sample the curve. First turn the θ`samples` and `noise` lists into a list of (θ, ϵ) pairs. We discussed how to do this in class using `Riffle` in conjunction with `Partition`. You may, if you like, find an easier way to do the same thing using `Transpose`. Call this list `pointsamples`.
As it stands, we have a list `pointsamples =` $\{\{\theta_1, \epsilon_1\}, \ldots, \{\theta_{4000}, \epsilon_{4000}\}\}$, which we'd like to turn into samples of the curve, $\{f(\theta_1, \epsilon_1), \ldots, f(\theta_{4000}, \epsilon_{4000})\}$. To do this, use `Apply`, and tell it that we only want it to operate on the elements in the first level of the list: `Apply[f, pointsamples, {1}]`. For more details of what it means to use `Apply` on different levels of a list, see the help.

iv. (2 points) Finally, `ListPlot` this list with the option `Joined->True`. Beautiful, isn't it? (If you're so inclined, try changing the scale of the noise; interesting things happen).

2. **Multivariable integration in Mathematica** (25 points) The `Integrate` command in Mathematica allows one to calculate integrals of arbitrary dimensionality, by placing multiple iterators $\{x_i, a_i, b_i\}$ in the call. For example, the call `Integrate[Boole[x^2 + y^2 ≤ 1], {x, −1, 1}, {y, −1, 1}]` computes the volume of the two-dimensional unit ball.

To compute the volume of a ball in dimension 27, we'd have to add 25 other iterators, not to mention wait an abominable amount of time– `Boole` is reasonable for low dimensional integrals, but the speed at which Mathematica computes `Boole` integrals decreases fast as a functions of the dimension.

This exercise will teach you how to create functions which make it easy to investigate multi-dimensional integrals as a function of the dimension. In particular, we'll experimentally verify Robbin's integral identity:

$$\int_{\substack{0\le x_1\le 1\\ x_1\le x_2\le 1\\ \vdots \\ x_{n-1}\le x_n\le 1}} \cdots \int \left| \begin{pmatrix} x_1^{a_1-1} & x_1^{a_2-1} & \cdots & x_1^{a_n-1} \\ x_2^{a_1-1} & x_2^{a_2-1} & \cdots & x_2^{a_n-1} \\ \vdots & \cdots & \ddots & \vdots \\ x_n^{a_1-1} & \cdots & \cdots & x_n^{a_n-1} \end{pmatrix} \right| = \left(\prod_{i=1}^{n} \prod_{j=i+1}^{n} (a_j - a_i) \right) \left(\prod_{i=1}^{n} a_i \right)^{-1} \left(\prod_{i=1}^{n} \prod_{j=1}^{i-1} (a_j + a_i) \right)^{-1};$$

here we take the a_i to be positive numbers. For a discussion of this identity(it is related to the Selberg integrals that arise in random matrix theory), see `http://arxiv.org/pdf/math/9805108`.

(a) (2 points) We want to choose simple coefficients a_i to start with, while retaining generality in our program, so define a function `a` for generating the i-th coefficient a_i according to the rule $a_i = i$. It's interesting to note that with this choice the integrand is a Vandermonde matrix.

(b) (5 points) Write a function `rhs`, on a single line, that takes `n` and the function `a` as arguments and computes the product on the right hand side of the identity.

(c) (5 points) Using `Det`, `Table`, and `a`, write a function `integrand`, on a single line, that takes `a` and `n` as arguments and calculates the integrand symbolically. To represent x_i in Mathematica , use `x[i]` or `x$_i$` (don't mix the notations, choose one and use it consistently). To check your expression, call `integrand` with `n = 3` and the `a` you defined previously; the result should be

$$-x_1^2x_2 + x_1x_2^2 + x_1^2x_3 - x_2^2x_3 - x_1x_3^2 + x_2x_3^2.$$

(d) (10 points) If I gave you a fixed value of `n`, you would now be able to test Robbin's identity by comparing `Integrate[ integrand[a,n], {x[1],0,1}, {x[2], x[1], 1}, ..., {x[n], x[n-1], 1}]` and `rhs[a,n]`. The crux of this exercise is automating the creation of this `Integrate` expression.

Define a function `lhs` which is a function of `a` and `n` so that for any value of `n`, `lhs[a,n]` will return the value of the left hand side of the above identity.

Hint: Use `Table`, `Apply`, and `Sequence` to generate the iterators and splice them into the argument sequence for `Integrate` (i.e. the construct `Sequence @@ Table` from problem 1).

(e) (3 points) Letting `n` range from 1 to 5, use `Table` and `TableForm` to display a table with the values of the integral in the first column and those of the product in the second column. It will help you with error checking to know that when `n` is 3, the value of the integral is $\frac{1}{180}$. Congratulations! You've shown that for this specific choice of `a`, Robbin's identity holds for at least five dimensions.

(f) **Bonus:** (10 points) Recreate the above table with one line of code (that is, show that you can do everything above without relying on intermediary functions)