ACM116 Problem Set 5, Proposed Problems

Sushant Sundaresh

31 October 2014

In problem set 4, we applied the Chi-squared test to decide whether or not empirically measured frequencies matched the predictions of simple models of genetics and the lottery. There, we answered "yes" or "no" and moved on. Here, let's see how much more can we learn about our models, and hopefully, our system. In the process, we will build comfort with Bayes' rule. Let's computationally infer an optimal model of a system.

The following two problems build on each other. They will require scripting in Matlab or Octave (octave-online.net).

Suppose a microprocessor is being sent packets of data by a master controller. Inputs change on a clock, so arrival times are discrete, and whether or not data is posted at a time n is not deterministic from the processor's point of view. The processor cannot process the data exactly as fast as packets come in, so it has a queue. Your goal in **Problem 1** is to model the data arrival times. Your goal in **Problem 2** is to use your model to estimate the probability that the microprocessor data queue will overflow.

Problem 1

(a)

You first decide to study the arrival timing of a sample input data stream over $M_{sample} = 10,000$ clocks. You treat arrivals as a time-ordered sequence S_k of ones and zeros, where $S_k = 0$ means no packet was recorded at time k and $S_k = 1$ means a packet was recorded. The arrivals (or lack thereof) look like coin tosses, so you consider R_k and N_k, defined below, as variables that indicate whether S_k an a$R_k$rival or $N_k$ot, respectively.

$$R_k = \begin{cases} 1 & S_k = 1 \\ 0 & S_k = 0 \end{cases}$$

$$N_k = \begin{cases} 1 & S_k = 0 \\ 0 & S_k = 1 \end{cases}$$

You decide calculate the following statistics on the off chance there is a glaring time-dependence to the arrivals.

- $1 - p_W = p_R = M_{sample}^{-1} \sum_{k=1}^{M_{sample}} S_k = 0.4968$
- $S_0 = 1$, that is, there is no arrival in the first time point
- $rr = 2515 = \sum R_k R_{k-1}$, the number of packets recorded right after another packet
- $wr = 2453 = \sum N_k R_{k-1}$, the number of lulls recorded right after a packet
- $rw = 2452 = \sum R_k N_{k-1}$, the number of packets recorded right after a lull
- $ww = 2579 = \sum N_k N_{k-1}$, the number of lulls recorded right after a lull

Note that $ww + wr + rw + rr + 1 = M_{sample}$ form a partition of the samples S. The one comes from S_0, which succeeds no other sample. The asymmetry you observe indicates the data cannot be modeled as an independent sequence of coin flips.

Suggest a single-parameter correction, ϵ, to the independent coin flipping model ($\boldsymbol{M_0}(p)$, below) by considering which types of corrections qualitatively introduce the observed dependence between samples. Indicate valid bounds for ϵ and p in your new model $\boldsymbol{M_1}(p, \epsilon)$. Leave things parametric in p, ϵ for now.

$$\boldsymbol{M_0}(p) = \begin{cases} \boldsymbol{P}_{prior}[p] & \sim Uniform(0,1) \\ \boldsymbol{P}_{prior}[R_k] & = p \\ \boldsymbol{P}_{prior}[N_k] & = 1 - p \\ \boldsymbol{P}[R_k | R_{k-1}] & = p \\ \boldsymbol{P}[R_k | N_{k-1}] & = p \\ \boldsymbol{P}[N_k | R_{k-1}] & = 1 - p \\ \boldsymbol{P}[N_k | N_{k-1}] & = 1 - p \end{cases}$$

(b)

Use your model to express $\boldsymbol{P}(S|\boldsymbol{M_1}(p, \epsilon))$, the likelihood that your data were generated by your model, assuming you specify parameters p, ϵ. Make use of rr, rw, wr, ww as calculated above, and don't forget about the first sample S_0!

(c)

As you noted in part (a), we actually have a class of models $\boldsymbol{M_1}(p, \epsilon)$ over a bounded region of (p, e)-space. We're looking for a way to quantitatively select models that do a good job at capturing what the features we've observed in our data.

If we refuse to consider alternatives to this class of models,and we still want to reason about "which model is best," then no matter what we do next, we've effectively decided to restrict ourselves to a perspective where only the space of our models (p, ϵ) are considered. That's the practical reality.

How well a model $\boldsymbol{M}$ reproduces key data S features directly impacts how much we believe in the model; however, if a model can represent our data over a very wide range of parameters, we might distrust its predictions. Either we have hit on a reasonable model, and the insensitivity to parameter variation is a robustness feature of our system, or we've introduced way too many parameters and can fit *anything.* Four to fit the elephant, five to wiggle its trunk, after all.

$$\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) = \frac{\boldsymbol{P}(S|\boldsymbol{M_1}(p_1, \epsilon_1))\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1))}{\boldsymbol{P}(S)}$$

$$\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) = \frac{\boldsymbol{P}(S|\boldsymbol{M_1}(p_1, \epsilon_1))\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1))}{\iint_{p,\epsilon} dpd\epsilon \boldsymbol{P}(S|\boldsymbol{M_1}(p, \epsilon))\boldsymbol{P}(\boldsymbol{M_1}(p, \epsilon))} \tag{1}$$

Bayes' rule, restated above, seems to reflect our intuition. The probability of any given model calculated via Bayes' rule increases with the quality of the fit to observed data, as

$$\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) \propto \boldsymbol{P}(S|\boldsymbol{M_1}(p_1, \epsilon_1))$$

Each model probability also decreases with the number of other equally "well-fitting" models, which increase the size of the normalization integral $\boldsymbol{P}(S)$ in the denominator, as

$$\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) \propto \frac{1}{\iint_{p,\epsilon} dpd\epsilon \boldsymbol{P}(S|\boldsymbol{M_1}(p, \epsilon))\boldsymbol{P}(\boldsymbol{M_1}(p, \epsilon))}$$

$\boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1))$ specifies our initial belief that a model $\boldsymbol{M_1}(p_1, \epsilon_1)$ is "the actual model." If the result of our reasoning returns $p = 0.05$ and $\epsilon = 0.5$, and your gut screams "it's the other way around," well, priors here are your intuition written explicitly.

Suppose you choose your priors uniform in some bounded region of (p, ϵ) space. Then equation (1) above reduces to the log likelihood

$$\ln \boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) = \ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_1, \epsilon_1)) + \ln K$$

over valid bounds, where $\ln K$ is constant, as both your priors and your normalization integral are constants. Logarithms here are good practice since probabilities get real small, real fast and computers start reading them all as "0" unless we do something about it.

Since our goal is simply to see which models best represent our data, we only really care about ratios of probabilities $\frac{\boldsymbol{P}(\boldsymbol{M_1}(p_1,\epsilon_1)|S)}{\boldsymbol{P}(\boldsymbol{M_1}(p_2,\epsilon_2)|S)}$, so we can more simply define the log likelihood in this case as

$$\ln \boldsymbol{P}(\boldsymbol{M_1}(p_1, \epsilon_1)|S) \equiv \ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_1, \epsilon_1))$$

ignoring the constant offset. The loss of the constant is immaterial to our calculations as,

$$\begin{aligned}\frac{\boldsymbol{P}(\boldsymbol{M_1}(p_1,\epsilon_1)|S)}{\boldsymbol{P}(\boldsymbol{M_1}(p_2,\epsilon_2)|S)} &= \exp(\ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_1,\epsilon_1)) - \ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_2,\epsilon_2)) - (\ln K - \ln K)) \\ &= \exp(\ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_1,\epsilon_1)) - \ln \boldsymbol{P}(S|\boldsymbol{M_1}(p_2,\epsilon_2)))\end{aligned}$$

Write an Octave script to calculate $\ln \boldsymbol{P}(\boldsymbol{M_1}(p,\epsilon)|S)$, for which you can easily modify the expression you derived in **Problem 1b**. Sample the distribution over a grid of reasonable p, ϵ values. Use your intuition to sample relevant (likely) parameter space. Plot this surface $(p, \epsilon, \ln \boldsymbol{P}(p,\epsilon|S_k, \boldsymbol{M_1}))$ using the function **meshc** and study the probability distribution using the **view** and **axis** commands to rotate and scale your plot.

(d)

Some questions follow. You don't need to answer any except the first three, though we'd love it if you thought about the others.

- Describe what you see - does the likelihood peak anywhere? At what parameter values?
- How distinct is (are) the peak(s), and how does this influence how confident you'd feel using the peak value(s) of p, ϵ to infer future packet arrival behavior?
- Does your intuition agree with values of p, ϵ given what you've seen of the data S_k?
- Some things to keep in mind:
  - if you're seeing complex likelihoods, your bounds for p, ϵ are probably overlapping, and you're computing $\ln x, x < 0$
  - a difference of 1 in log-likelihoods between two points corresponds to an ~ 2.3 fold difference in probability $\frac{\boldsymbol{P}(p,\epsilon|S_k,\boldsymbol{M_1})}{\boldsymbol{P}(p',\epsilon'|S_k,\boldsymbol{M_1})}$
  - there are no right or wrong answers here

Other things to think about...

- Do you feel your results justify the added complexity of your model?
- Do you think having more (consistent) data would sharpen the peak?
- If we had only given you the results of 1000 samples, would your inference have suffered? That is, play with your expression from **Problem 1b** and suggest at a threshold M_{sample} below which the effects of ϵ might start to become negligible.

- Consider the process by which we arrived at our estimate of the plausibility of various parameters for our model; in what ways is it similar to the Chi-squared test?

Problem 2

Restrict the models you're considering to the optimal model $\boldsymbol{M_1}(p_{optimal}, \epsilon_{optimal})$ you found in **Problem 1,** and the model $\boldsymbol{M_1}(p_{optimal}, \epsilon = 0)$ (these might be one and the same). Your goal in this problem is to evaluate the probability that the processor data queue, which has been programmed to buffer up to $N - 1 = 14$ consecutive packet arrivals in a time span of $M = 1000$ clocks, will overflow. That is, for each packet arrival model, what is the probability of seeing a run of packet arrivals (1's) of length greater than or equal to $N = 15$, in a sequence of $M = 1000$ time steps?

For this problem, a hint is in order, because direct counting is very painful (though kudos to you if you try). Consider instead the following:

$$\begin{aligned}
\boldsymbol{P}[R_k|R_{k-1}, \boldsymbol{M_0}] &= p \\
\boldsymbol{P}[R_k|N_{k-1}, \boldsymbol{M_0}] &= p \\
\boldsymbol{P}[N_k|R_{k-1}, \boldsymbol{M_0}] &= 1 - p \\
\boldsymbol{P}[N_k|N_{k-1}, \boldsymbol{M_0}] &= 1 - p
\end{aligned}$$

When we defined the probabilities above in specifying our original model for **Problem 1,** we described what's known as a Markov chain. There are two states: R_k and N_k, which are achieved again and again over time in a sequence that isn't deterministic. The probabilities above are transition probabilities from the arrival state to the non-arrival state of the chain as it evolves; they stay constant with time in this example. The output we see is the sequence of states of the chain, S_k. Let's try calculating the probability of being in state R on the fourth time point, if all we know is how states transition between each other, and how we started off in S_0.

$$\begin{aligned}
\boldsymbol{P}[S_4 = R] &= \boldsymbol{P}[S_4 = R|S_3 = R]\boldsymbol{P}[S_3 = R] + \boldsymbol{P}[S_4 = R|S_3 = N]\boldsymbol{P}[S_3 = N] \\
&= \boldsymbol{P}[R_k|R_{k-1}]\boldsymbol{P}[S_3 = R] + \boldsymbol{P}[R_k|N_{k-1}]\boldsymbol{P}[S_3 = N]
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{P}[S_4 = N] &= \boldsymbol{P}[S_4 = N|S_3 = R]\boldsymbol{P}[S_3 = R] + \boldsymbol{P}[S_4 = N|S_3 = N]\boldsymbol{P}[S_3 = N] \\
&= \boldsymbol{P}[N_k|R_{k-1}]\boldsymbol{P}[S_3 = R] + \boldsymbol{P}[N_k|N_{k-1}]\boldsymbol{P}[S_3 = N]
\end{aligned}$$

which can also be written in matrix form as

$$\begin{bmatrix} \boldsymbol{P}[S_4 = R] \\ \boldsymbol{P}[S_4 = N] \end{bmatrix} = \begin{bmatrix} \boldsymbol{P}[R_k|R_{k-1}] & \boldsymbol{P}[R_k|N_{k-1}] \\ \boldsymbol{P}[N_k|R_{k-1}] & \boldsymbol{P}[N_k|N_{k-1}] \end{bmatrix} \begin{bmatrix} \boldsymbol{P}[S_3 = R] \\ \boldsymbol{P}[S_3 = N] \end{bmatrix}$$

We notice that this is exactly the same operation we'd perform to get $\begin{bmatrix} \boldsymbol{P}[S_3 = R] \\ \boldsymbol{P}[S_3 = N] \end{bmatrix}$ and iterate, finding that:

$$\begin{aligned} \begin{bmatrix} \boldsymbol{P}[S_{k'} = R] \\ \boldsymbol{P}[S_{k'} = N] \end{bmatrix} &= \begin{bmatrix} \boldsymbol{P}[R_k|R_{k-1}] & \boldsymbol{P}[R_k|N_{k-1}] \\ \boldsymbol{P}[N_k|R_{k-1}] & \boldsymbol{P}[N_k|N_{k-1}] \end{bmatrix}^{k'} \begin{bmatrix} \boldsymbol{P}[S_0 = R] \\ \boldsymbol{P}[S_0 = N] \end{bmatrix} \\ &= \begin{bmatrix} p & p \\ 1-p & 1-p \end{bmatrix}^{k'} \begin{bmatrix} \boldsymbol{P}[S_0 = R] \\ \boldsymbol{P}[S_0 = N] \end{bmatrix} \end{aligned}$$

That's very useful, that the state transitions can be represented as a matrix. If you were to walk along some hypothetical future sequence of data arrivals, at any point you'd know if you were in a run, and how long it was, wouldn't you? This "being in a run of length n" is also a state, and seeing an R or N next in your hypothetical sequence updates this state - it either lengthens the run or cuts it short. Try drawing out the evolution of this run state as a diagram, defining the states appropiately to match what you're interested in calculating. Do you think you could represent the state transitions as a matrix operation, as we did here? What do the elements of the matrix mean? Does this help you solve the original problem?