Online Question Asking Algorithms For Measuring Skill

Jack Stahl

December 14, 2007

1 Abstract

We wish to discover the best way to design an online algorithm for measuring hidden qualities. In particular, we study the problem of designing a computer-adaptive test in a GRE-like setting. We first define a model for the probability that a test taker of skill θ will correctly answer a test question of difficulty d_q. We then discuss basic properties of the setting. We present an multiplicative-weights algorithm for selecting questions, and present an incomplete argument for its optimality. We then present experimental results to supplement that argument. Finally, we discuss related issues in the setting of computer-adaptive testing.

2 The Model

We model the situation in the following way: each test taker has some skill θ, and each question has difficulty d_q. The probability that the test taker correctly answers the question (an event we denote X) is given by

$$P_{d_q}(X;\theta) = \mathbf{sigmoid}(\theta - q) = \frac{1}{1+e^{-(\theta-d_q)}}$$

That is, if $\theta >> q$, then the test-taker will probably answer the question correctly; if $\theta << q$, then probably not. This model is common with modern item-response theory; derivatives of it are used in practice (see e.g [8]). Common extensions include allowing a constant term that can account for that one should have a lower bound on the probability of an incorrect answer on a multiple choice question, but this could be done equally well by choosing an alternative logistic function.

3 Basic Properties

We now consider our first basic question. Given the model above for how a test-taker performs, is it "better" to mess up easy questions but get hard ones right, or is it better to ace the easy questions but mess up the hard ones? Formally, we fix a set of difficulty levels for n questions, $Q = \{q_1, \ldots, q_n\}$. We now fix some number w of incorrect answers, and we consider all possible sets of incorrect answers $W \subseteq Q, |W| = w$, and let the correctly-answered questions be $R = Q \setminus W$. Letting $f(Y;\theta)$ be the likelihood of these answer results Y as a function of θ, we have:

$$\begin{aligned} f(Y;\theta) &= \prod_{w\in W} \frac{e^{-(\theta-d_w)}}{1+e^{-(\theta-d_w)}} * \prod_{r\in R} \frac{1}{1+e^{-(\theta-d_r)}} \\ &= \prod_{w\in W} e^{-(\theta-d_w)} * \prod_{q\in Q} \frac{1}{(1+e^{-(\theta-d_q)})} \\ &= \prod_{w\in W} e^{d_w} * e^{-\theta|W|} * \prod_{q\in Q} \frac{1}{(1+e^{-(\theta-d_q)})} \end{aligned}$$

Indeed this is a function of the set W; we observe that the data is more likely for harder questions, which is intuitively correct (it is unusual for someone to be able to answer hard questions but not easy ones). However, consider the relative likelihood of two different values θ and θ', i.e. $f(Y;\theta)/f(Y;\theta')$. This is not a function of W explicitly but only of Q and $|W|$. In other words, *which* questions were answered correctly does nothing to inform us about how skilled the test-taker might be! The answer to our original question is that it simply does not matter which questions a test-taker misses in terms of estimating skill.

Another question we might ask is, what is the right question to ask a test-taker of a certain skill level? Put another way, if we thought we knew θ, how could we prove it most efficiently. Formally, we ask how to maximize, over difficulty d_q, the Fisher Information of X, the random variable of whether or not the test taker correctly answers question q

$$\mathcal{I}_{d_q}(\theta) = -\mathrm{E}\left[\frac{\partial^2}{\partial\theta^2}\log f(y;\theta)\right]_y$$

$$= -\Big(\frac{1}{1+e^{-(\theta-d_q)}}\frac{\partial^2}{\partial\theta^2}\big(\log\frac{1}{1+e^{-(\theta-\partial_q)}}\big) + \frac{e^{-(\theta-d_q)}}{1+e^{-(\theta-d_q)}}\frac{\partial^2}{\partial\theta^2}\big(\log\frac{e^{-(\theta-d_q)}}{1+e^{-(\theta-d_q)}}\big)\Big)$$

$$= -\Big(-\frac{1}{1+e^{-(\theta-d_q)}}\frac{\partial^2}{\partial\theta^2}\log(1+e^{-(\theta-d_q)}) + \frac{e^{-(\theta-d_q)}}{1+e^{-(\theta-d_q)}} * \frac{\partial^2}{\partial\theta^2}\big(\log(e^{-(\theta-d_q)}) - \log(1+e^{-(\theta-d_q)})\big)\Big)$$

$$= \frac{\partial^2}{\partial\theta^2}\ \log(1+e^{-(\theta-d_q)})$$

$$= \frac{e^{-(\theta-d_q)}}{1+e^{-(\theta-d_q)}}\big(1-\frac{e^{-(\theta-d_q)}}{1+e^{-(\theta-d_q)}}\big)$$

Since this is $X(1-X)$ (for $0 \leq X \leq 1$), we see this is maximized by $X = \frac{1}{2}$, or in other words $d_q = \theta$. This is again no surprise – if we asked something too easy, it would hard to differentiate θ from anything else above d_q (and likewise for too hard). What is interesting that a binomial random variable parameterized by θ has F.I. $\frac{1}{\theta(1-\theta)}$, and so it is generally binomials with extreme probabilities that carry the most information. Here we depend very strongly on the steepness of the sigmoid function at 0.

4 Potential Directions

We now turn our attention to the main question of how to choose an online algorithm which adapts to the results of the test-taker in choosing further questions to ask. One potential way to look at this problem is as an instance of noisy binary search, A naive such algorithm would repeatedly ask questions of some difficulty d_q until it was confident within ε of either $\theta > d_q$ or $\theta < d_q$, then proceed as in binary search. The problem with this approach is that the number of questions asked before the binary search has a factor ε leading constant. Since in a situation such as the GRE questions are extremely limited, we can not afford to take, e.g., 10 questions at one difficulty level. It is however conceivable that backtracking ala [4] could overcome this difficulty. One alternative algorithm is to greedily maximize the fisher information $\mathcal{I}_{d_q}(\hat{\theta})$, where $\hat{\theta}$ is the current MLE estimate of θ; this is more or less what is used by the GRE (see [6, 7] for details), and corresponds (via the above lemma) to always picking a question whose difficulty matches the current estimate of the test takers skill. One problem with this is that the MLE given only correct answers is the highest possible value of θ, which means that the second question will either be the easiest or hardest possible question, intuitively unusual. To combat this, we suggest the following:

5 A Multiplicative Weights Algorithm

1. Initialize weights $w(\theta)$ to a prior (if available / MAP learning) or else the uniform distribution (for MLE).

2. For each $1, \ldots, n$

   - Find $\theta_q = \arg\max_{\theta_q} \sum_\theta w(\theta) * \mathcal{I}_{\theta_q}(\theta)$
   - Ask question q with difficulty θ_q
   - Given result y, update $w(\theta) = p(y|\theta) * w(\theta)$ and normalize

3. Output $\arg\max_{\hat{\theta}} w(\hat{\theta})$

It is straight forward to see that because we are doing Bayesian updating at each question, our output will either be the MLE or the MAP estimate (depending on whether or not we used a non-uniform prior). The key difference between this and the greedy algorithm is that (we will speak in Bayesian semantics but the reasoning holds for an MLE setting as well) we maximize the expected Fisher Information given our current distribution for θ instead of merely maximizing Fisher Information at a single (possibly extreme) point. To illustrate this, consider the case where $\theta \in [-5, 5]$ (discretized into tenths), and the user correctly answers one question of difficulty 0. Then difficulty of the next question asked would be 2.7, not 5.0 as it would be in the greedy situation. While the MLE may be 5.0, it is also relatively likely that θ is any other value greater than 0, and a question of difficulty 5.0 would not be an efficient question for most other positive values of θ. Another advantage over the greedy algorithm is that one can choose to use a prior. In tests such as the GREs, a prior such as a normal distribution is usually already a modeling assumption, and it can now be explicitly incorporated into the model.

6 A Failed Proof

We wish to say something substantial of this algorithm. Ideally, we would be able to say that it is minimizes the mean-squared error over all possible tests, or that it is a minimum-variance unbiased estimator of θ. Here we sketch the idea of a proof which we long attempted but failed to turn into a rigorous statement. The idea is the following: Fundamentally, the Fisher Information of any test bounds the variance of any estimator, and in particular, MLEs are asymptotically unbiased and asymptotically achieve a variance of $1/\mathcal{I}(\theta)$. Therefore, if we can maximize the Fisher Information of the the total test we can minimize the variance of the MLE of the whole test. Since Fisher Information is additive, this reduces to maximizing the Fisher Information of each individual question. By induction, we can assume we have maximized Fisher Information up till any particular question q. Since we have maximized Fisher Information, our current distribution on $w(\theta)$ is as accurate as possible. Therefore, our computation of the expected Fisher Information (with respect to the true θ) is as accurate as possible, and so indeed at question q we maximize the possible Fisher Information from the question. This proof has three major difficulties. One, we wish to drawn both on Bayesian expectation semantics when maximizing Fisher Information as well as non-Bayesian MLE semantics in discussing the relationship between variance of an estimator and Fisher Information. Two, the direct relationship between Fisher Information and the variance / unbiasedness of MLEs is only asymptotic over many i.i.d. samples; here we do not have any i.i.d. samples. Three, it is unclear that maximizing Fisher Information corresponding to a "maximally accurate" *a posteriori* or likelihood distribution. While attempts can be made to rectify any of these problems, we were unable to glue all of them together simultaneously.

7 Experimental Data

In the absence of theoretical guarantees, we wrote a Python script in order to test the performance of our multiplicative weights algorithm against the similar greedy algorithm. In general, with sufficiently many questions, the algorithms performed remarkably similarly. However, when questions are very limited ($n \leq 5$), the variance of the multiplicative-weights algorithm is significantly lower. This agrees with our earlier discussion that, with very very few questions, the MLE is not yet accurate, and so maximizing Fisher Information with respect to it is not necessarily very helpful. However, once the greedy algorithm has enough data for a reasonable MLE estimate the algorithms become more or less equivalent. It does also appear that the multiplicative-weights algorithm is slightly more efficient at detecting extreme θ, perhaps because it does not "punish" them with extreme questions too early. Both estimators appear to have a bias towards the middle of the domain of θ that tends towards 0 with increasing n; such a bias might contribute to the difficulty of our failed proof. Finally, we originally expected that a multiplicative-weights algorithm would minimize the variance of our estimator. From the data, we can see that, even for sufficiently large questions, in fact the multiplicative weights estimator does have a smaller sample variance, along with a more accurate sample mean, but only noticeably so when the discretization of the θ-space is small enough; otherwise preditions are "easy" enough that the algorithms seem to be identical.[1]

8 Assorted Discussion

From the data it appears that one major advantage of the multiplicative weights algorithm is its resistance to bad early estimates Another way to stave this off would be to always use a prior, and another is to use KL Information instead of Fisher Information [3]. This however yields poor results if done exclusively, and so [3] suggests using KL Information only for early questions; one nice property of the multiplicative weights algorithm is that it does not require such tuning.

There are many other multiplicative-weights algorithms for various other domains ([1], [2]). However, most of them try to minimize some loss function in some repeated game. In this situation, we are concerned only with the quality of our final estimate. One can try to construct a loss function such as the $\mathcal{I}_{d_\theta}(\theta) - \mathcal{I}_{d_q}(\theta)$, the difference in Fisher Information between asking question q and the optimal question. However, we are not privy to our incurred per-question losses. Because of this distinction in optimization and knowledge, straight-forward Bayesian updating seems to make sense here where it might not have in other multiplicative-weights algorithms.

We have claimed that the GRE uses the greedy algorithm, but we know that the GRE does not ask extremely hard questions immediately as predicted by our analysis. Why is that? The GRE does in fact use the greedy algorithm, but it also tries to satisfy certain constraints. For example, no question is supposed to be seen by more than 20% of test-takers. More importantly, there is a constraint that each test "look like it were designed by an expert test taker", and so the test biases itself towards certain "normal" distributions of questions. For more, see [7, 5].

When designing a test, a lot more comes into play than simple point estimation that we do here. In particular, there are strategic aspects involved. For example, one might want to prove that no test-taker ever has incentive to intentionally get a question wrong (in order to get easier questions that he can then answer correctly). There are also fairness properties, such as that if θ is sufficiently larger θ' then θ is near-guaranteed to outscore θ' (i.e. our estimation of θ should be near-guaranteed to be higher than our estimation of θ'). However, most fairness constraints seem to reduce to questions of bias and variance in our estimator. For example, a perfectly unbiased estimator of variance 0 – an omniscient test – would necessarily mean all fairness was preserved since estimation would correspond exactly to reality.

9 Conclusion

Doing theoretical statistics on an online learning problem is a challenging ordeal. Because the questions themselves as well as their answers are highly interdependent, standard i.i.d.-focused results are hard to apply. There are many applicable ideas floating out there on the issue – current computer adaptive tests, multiplicative-weights

[1]In the table, [-a, a] means that the domain for θ was the integers beginning at $-a$ and ending at a. When we write $[-5.0, 5.0]$, this means we instead discretized by hundredths. n is the number of questions.

algorithms, noisy binary search, point estimation theory – but marrying them together remains a problem which will take more time to solve. However, if that should prove intractable, there are many strategic-type questions that could also shed light on the issue. For example, our first lemma suggested that users do not have incentive to concentration preparation for either the challenging or the easy questions. However, a stronger statement would have to be proved in order to account for the way in which answer correctness actually affects the question set itself. Nonetheless, the algorithm we present here seems like a promising direction for minimizing the variance in an online computer adaptive test.

References

[1] Y. Freund *Predicting a Binary Sequence Almost as Well as the Optimal Biased coin* **Conference on Computational Learning Theory**, 1996.

[2] H. Freund, R. Schapire *A Decision-Theoretic Generalization of Online Learning and an Application to Boosting* **European Conference on Computational Learning Theory**, 1995.

[3] H. Chang, Z. Ying *A Global Information Approach to Computerized Adaptive Testing* **Applied Psychological Measurement**, 1996.

[4] R. Karp, R. Kleinberg, *Noisy Binary Search and Its Applications.* **Symposium on Discrete Algorithms**, 2007.

[5] G. Schaeffer et al. *The Introduction and Comparability of the Computer Adaptive GRE Test.* **ETS Report**, 1995.

[6] M. Stocking, L. Swanson, and M. Pearlman, *Application of an Automated Item Selection Method to Real Data.* **Applied Psychological Measurement**, 1993.

[7] M. Stocking, L. Swanson, *A Method for Severely Constrained Item Selection in Adaptive Testing.* **Applied Psychological Measurement**, 1993.

[8] W.J. van der Lin, E. Boekkoi-Timming, *A Maximin Model for Test Design with Practical Constraints.* **Psychometrica**, 1987.

Table 1: Experimental Results

True θ, n	**MW** $(\hat{\mu}, \hat{\sigma})$	**Gr** $(\hat{\mu}, \hat{\sigma})$
$0, [-5, 5], 3$	-0.06, 1.948	-0.138, 1.965
$-3, [-5, 5], 3$	-2.851, 2.099	-2.9, 2.101
$-5, [-5, 5], 3$	-4.343, 1.181	-3.944, 1.181
$0, [-50, 50], 3$	-5.136, 14.81	-5.01, 117.544
$-25, [-50, 50], 3$	-26.44, 19.76	-22.04, 39.11
$-50, [-50, 50], 3$	-49.989, 0.121	-45.458, 20.65
$0, [-50, 50], 10$	-0.026, 1.021	0.164, 1.058
$-25, [-50, 50], 10$	-24.98, 0.905	-25.05, 0.911
$-50, [-50, 50], 10$	-49.65, 0.407	-45.65, 0.407
$0, [-5.0, 5.0], 10$	-0.073, 0.393	-0.163, 0.402
$-2.5, [-5.0, 5.0], 10$	-2.469, 0.416	-2.388, 0.429
$-5.0, [-5.0, 5.0], 10$	-4.67, 0.199	-4.60, 0.214
$0, [-5.0, 5.0], 50$	0.052, 0.082	-0.172, 0.087
$-2.5, [-5.0, 5.0], 50$	-2.464, 0.066	-2.508, 0.067
$-5.0, [-5.0, 5.0], 50$	-4.950, 0.031	-4.838, 0.040