Scoring Algorithm for HMMT Individual Exams

Harvard-MIT Math Tournament

Evan Chen

Last updated February 21, 2017

Contents

1 Synopsis

1.1 Input

Each exam contains at least one **problem**. The exams are taken by several **contestants**.

This algorithm takes several **observations**: i.e. it takes as inputs pairs (c, p) of contestants and problems, and for each such pair is told whether or not c **solved** p. During the actual tournament, every problem is attempted by every contestant; however, the algorithm will not use this assumption.

1.2 Output

Based on this, the algorithm outputs:

- An **strength** $\alpha_c \in \mathbb{R}_{\geq 0}$ for each competitor c, and
- A **weight** $\beta_p \in [3, 8]$ for each problem p, which represents its difficulty.

Note that the strength is *not* the same as the score of a contestant. In particular, as described below, each contestant is given a single strength value across all exams.

1.3 Usage in contest

The calculation at the actual HMMT is done in one pass. For example, at HMMT February 2016, there were three exams with 10 problems each and 700 contestants. Thus there were $700 \cdot 10 \cdot 3 = 21000$ observations as input. Note that the algorithm is applied only once! For example the observations on the Algebra test affect the weights of the Geometry test.

For each particular 10-problem **exam**, the **score** of a contestant on that exam is the sum of the difficulties β_p for each problem that they solve. The score of a contestant is used to determine their rankings within each individual test. Scores across different exams are not comparable.

For individual sweepstakes, the strength α_c is used instead (in order to ensure that no particular exam dominates in determining the aggregate individual score). Contestants are ranked based on the value of α_c assigned to them. When determining team sweepstakes, the individual contribution (out of 800) is proportional to the sum of α_c across the team members c, scaled so that the strongest teams earns the maximal 800 points.

2 Design

2.1 Advantages

The scoring algorithm is designed to meet the following criterion:

- The system provides a careful way to **compare scores across tests**, leading to less noise in the aggregate individual and team rankings.
- The system **factors all possible inputs**, rather than for example merely the top 10 contestants or otherwise, as occurred in previous HMMT tournaments.
- The system is **resistant to preconceived difficulty**. Originally, problem czars were forced to estimate the difficulty of every problem by assigning it a weight; this leads to possible noise in the results. The algorithmic system determines the difficulty based on the actual performance.
- The system is **hard to exploit**, in part because it takes inputs from all contestants, and in part because it is so complicated that a contestant is probably better off thinking about the exam problems.

2.2 Qualitative criteria

The following additional criteria are satisfied:

- The strength of each contestant is nonnegative,
- The strength of a contestant is unbounded, but not infinite even if they solve every problem.
- The strength of a contestant is zero if they solve no problems.
- Problem weights lie in the interval $(3, 8)$.
- The distribution functions are smooth.

2.3 Choice of priors

In light of this, we select the following parameters, which describe how the α_c and β_p should be interpreted.

- The strength $\alpha = \alpha_c$ of each contestant c is *a priori* distributed in $\mathbb{R}_{\geq 0}$ according to
$$\mathbf{P}(\alpha) = \exp(-\alpha). \tag{1}$$
- The weight (difficulty) $\beta = \beta_p$ of each problem p is *a priori* distributed in $[3, 8]$ according to
$$\mathbf{P}(\beta) = \exp\left(-\frac{5}{(\beta-3)(8-\beta)}\right). \tag{2}$$

Now we relate the strength $\alpha = \alpha_c$ of contestant c to the difficulty $\beta = \beta_p$ of problem p by postulating that the probability that c solves p is

$$\mathbf{P}\left(c \text{ solves } p \mid \alpha_c = \alpha \text{ and } \beta_p = \beta\right) = \frac{1}{1+\exp(\beta/\alpha)}. \tag{3}$$

Assuming (1), (2), (3) and holding fixed the set of observations of the tournament:

> *The algorithm outputs the unique set of α_c and β_p for which the outcome of the tournament was most likely.*

The rest of the document describes how to actually compute the maximum.

3 Algorithm Description

The algorithm is a slightly modified version of the Rasch model.

3.1 Pseudocode

Let us for brevity denote the function in (3) by

$$w(\alpha, \beta) \stackrel{\text{def}}{=} \frac{1}{1+\exp(\beta/\alpha)}.$$

Note that $w(\alpha, \beta) \in (0, 1)$.

Now consider the following systems of equations, whose origin we explain below. First, for every $\alpha = \alpha_c$ we have the equation

$$0 = -1 + \sum_{p \text{ solved by } c} \beta_p \alpha^{-2} - \sum_{p \text{ took by } c} \beta_p \alpha^{-2} w(\alpha, \beta_p). \tag{4}$$

Moreover, for every $\beta = \beta_p$ we have the equation

$$0 = \frac{1}{(\beta-3)^2} - \frac{1}{(8-\beta)^2} + \sum_{c \text{ taken by } p} \alpha_c^{-1} w(\alpha_c, \beta) - \sum_{c \text{ solved by } p} \alpha_c^{-1}. \tag{5}$$

The algorithm binary searches for a set of $\vec{\alpha}$ and $\vec{\beta}$ which simultaneously satisfy (4) and (5). It is proved below that this set of parameters is unique; this is the output of the algorithm.

3.2 Derivation

We show the derivation of equations (4) and (5).

Fix a set of observations for the tournament. Now consider a choice of $\vec{\alpha}$ and $\vec{\beta}$ is known. Then the probability of the observations is distributed according to

$$F(\vec{\alpha}, \vec{\beta}) = \prod_c \mathbf{P}(\alpha_c) \prod_p \mathbf{P}(\beta_p) \prod_{c \text{ solved } p} w(\alpha_c, \beta_p) \prod_{c \text{ missed } p} (1 - w(\alpha_c, \beta_p)) \tag{6}$$

assuming the events are independent.

We now claim that:

Proposition 1. *If a choice of $\vec{\alpha}$ and $\vec{\beta}$ optimizes F, then it is a solution to (4) and (5). In other words, the outputted $\vec{\alpha}$ and $\vec{\beta}$ are those for which the observed outcome was most probable.*

Proof. Note that

$$\prod_{c \text{ solved } p} w(\alpha_c, \beta_p) \prod_{c \text{ missed } p} (1 - w(\alpha_c, \beta_p)) = \prod_{c \text{ solved } p} e^{-\beta_p/\alpha_c} \prod_{c \text{ took } p} \frac{1}{1 + e^{-\beta_p/\alpha_c}}.$$

From this, and plugging in (1) and (2), we get

$$F(\vec{\alpha}, \vec{\beta}) = \prod_c e^{-\alpha_c} \prod_p e^{-\frac{5}{(8-\beta_p)(\beta_p-3)}} \prod_{c \text{ solved } p} e^{-\beta_p/\alpha_c} \prod_{c \text{ took } p} \frac{1}{1 + e^{-\beta_p/\alpha_c}}.$$

It is equivalent to maximize $\log F$, which is

$$\log F = -\sum_c \alpha_c - \sum_p \frac{5}{(8-\beta_p)(\beta_p-3)} - \sum_{c \text{ solved } p} \frac{\beta_p}{\alpha_c} + \sum_{c \text{ took } p} \log \frac{1}{1 + e^{-\beta_p/\alpha_c}}.$$

Then right-hand sides of (4) and (5) are the *partial derivatives* of $\log F$ with respect to α_c and β_p for each c and p. The partial derivatives equal zero at any local maximum of $\log F$, as desired. □

In fact, one can actually check that the right-hand sides of (4) and (5) are monotonic in α and β respectively. Consequently, F is convex. This implies:

Proposition 2. *The function $\log F$ has a unique point at which all partial derivatives vanish, and that point is a maximum for F.*

This also implies that one can numerically solve the equations by the following binary search procedure. We let $\vec{\alpha}_0$ be an arbitrary point. Then we let $\vec{\beta}_n$ be the solution to (5) given $\vec{\alpha}_{n-1}$. Notice that one can actually solve for each component β_p using binary search separately, as each equation of the form (5) involves only a single β_p. Similarly, we can let $\vec{\alpha}_n$ be the solution to (4) given $\vec{\beta}_{n-1}$ in the same fashion. The pairs $(\vec{\alpha}_n, \vec{\beta}_n)$ will converge to the maximum since we are looking at partial derivatives of a concave function, so we now have an iterative procedure for computing the output.