A Better BCS

Rahul Agrawal, Sonia Bhaskar, Mark Stefanski

I. Introduction

Most NCAA football teams never play each other in a given season, which has made ranking them a long-running and much-debated problem. The official Bowl Championship Series (BCS) standings are determined in equal parts by input from coaches, sports writers, and an average of algorithmic rankings. These standing determine which teams play for the national championship, and, in part, which teams appear in each bowl (playoff) game.

We seek an algorithm that ranks teams better than the BCS standings do. In particular, since the BCS standings serve to match teams in bowl games, we want better bowl game prediction accuracy. This is a uniquely difficult prediction problem because, by design, teams squaring off in a bowl game are generally evenly-matched and come from different conferences, which means they are unlikely to have many common opponents, if any.

II. The Model

We model the outcome of a game between two teams as the difference between their (noisy) competitiveness levels. We assume a team has a certain fixed average ability, μ_i, and that for each game a multitude of independent random factors collectively determines to what extent that team's performance falls short of or exceeds its average ability. The Central Limit Theorem suggests that we model these random factors as $\epsilon \sim N(0, \sigma^2)$. Therefore, team i's competitiveness in any given game is distributed as $x_i = \mu_i + \epsilon \sim N(\mu_i, \sigma^2)$. So when team i plays team j, each team *independently* samples from its competitiveness distribution, and the resulting score difference is distributed as $x_j - x_i \sim N(\mu_j - \mu_i, 2\sigma^2)$.

We denote by $y_{i,j}^{(k)}$ the *actual* score difference of the kth contest between team i and team j. And to avoid redundancy, we adopt the convention that the score difference between team i and team j where $j > i$ is team j's score less team i's score. So our model says that for the kth meeting between teams i and j,

$$y_{i,j}^{(k)} = x_j - x_i = \mu_j - \mu_i + \delta_{i,j}^{(k)},$$

where the $\delta_{i,j}^{(k)} = \sqrt{2}\epsilon \sim N(0, 2\sigma^2)$ are mutually independent.

This model easily extends to m games played among a common pool of n teams. For instance, if each of the n teams plays every other exactly once:

$$\underbrace{\begin{bmatrix} y_{1,2} \\ \vdots \\ y_{1,n} \\ y_{2,3} \\ \vdots \\ y_{2,n} \\ \vdots \\ y_{n-1,n} \end{bmatrix}}_{\mathbf{y}_{m\times 1}} = \begin{bmatrix} x_2 - x_1 \\ \vdots \\ x_n - x_1 \\ x_3 - x_2 \\ \vdots \\ x_n - x_2 \\ \vdots \\ x_n - x_{n-1} \end{bmatrix} = \underbrace{\begin{bmatrix} -1 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & 0 & 0 & \dots & 0 & 1 \\ 0 & -1 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & -1 & 0 & \dots & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & -1 & 1 \end{bmatrix}}_{\mathbf{A}_{m\times n}} \underbrace{\begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \vdots \\ \mu_{n-1} \\ \mu_n \end{bmatrix}}_{(\mu_{\mathbf{x}})_{n\times 1}} + \underbrace{\begin{bmatrix} \delta_{1,2} \\ \vdots \\ \delta_{1,n} \\ \delta_{2,3} \\ \vdots \\ \delta_{2,n} \\ \vdots \\ \delta_{n-1,n} \end{bmatrix}}_{\Delta_{m\times 1}}$$

If teams i and j do play each other multiple ($K > 1$) times, then there will be K (duplicate) rows of $\mathbf{A}$ corresponding to the $y_{i,j}^{(1)}, \dots, y_{i,j}^{(K)}$ entries of $\mathbf{y}$. If teams i and j do not play at all, then $y_{i,j}$ will be absent from $\mathbf{y}$, as will the corresponding row of $\mathbf{A}$. In reality, many teams might not have played each other in the current season – or, in some cases, ever – so many of the $y_{i,j}$ could be absent from $\mathbf{y}$. In any case, our model reduces to $\mathbf{y} = \mathbf{A}\mu_{\mathbf{x}} + \Delta$.

III. Unweighted Linear Regression

A. Finding The Maximum Likelihood Abilities

Since we can rank teams by their abilities, our goal here is to find the maximum likelihood estimate of ability vector $\mu_{\mathbf{x}}$. To do so, we must set a reference value for the μ_i because otherwise there is no way of distinguishing between $\mu_{\mathbf{x}}$ and $\mu'_{\mathbf{x}}$ that differ by a constant offset $k\mathbf{1}_n$:

$$\mathbf{A}\mu'_{\mathbf{x}} = \mathbf{A}(\mu_{\mathbf{x}} + k\mathbf{1}_n) = \mathbf{A}\mu_{\mathbf{x}} + k\mathbf{A}\mathbf{1}_n = \mathbf{A}\mu_{\mathbf{x}}.$$

That is, $k\mathbf{1}_n \in \mathcal{N}(\mathbf{A})$ by construction of $\mathbf{A}$. So we introduce the condition $\mathbf{1}_n^T \mu_{\mathbf{x}} = \mu_1 + \ldots + \mu_n = 0$, which sets the average ability of the pool of teams to zero.

We would like to find the maximum likelihood estimate of $\mu_{\mathbf{x}}$ by the least-squares approximation

$$\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{y}$$

and then adjust the result so that $\mathbf{1}_n^T\mu_{\mathbf{x}} = 0$. But $\mathbf{A}$ is never full rank because its nullspace is always nontrivial, and therefore $\mathbf{A}^T\mathbf{A}$ cannot be full rank (or, consequently, invertible). However, under certain conditions on the games played (teams are "competitively linked"),

$$(\mathbf{A}_c)_{(m+1)\times n} = \begin{bmatrix}\mathbf{1}_n^T \\ \mathbf{A}\end{bmatrix}$$

is "skinny" ($m+1 \geq n$) and full rank (see Proof 1 in the Appendix). Note that

$$\underbrace{\begin{bmatrix}0 \\ \mathbf{y}\end{bmatrix}}_{(\mathbf{y}_c)_{(m+1)\times 1}} = \underbrace{\begin{bmatrix}\mathbf{1}_n^T \\ \mathbf{A}\end{bmatrix}}_{\mathbf{A}_c}\mu_{\mathbf{x}} + \underbrace{\begin{bmatrix}0 \\ \Delta\end{bmatrix}}_{\Delta_c}.$$

So we can compute

$$\mu_{\mathbf{x}}^* = \left(\mathbf{A}_c^T\mathbf{A}_c\right)^{-1}\mathbf{A}_c^T\mathbf{y}_c.$$

Importantly, $\mu_{\mathbf{x}}^*$ is the maximum likelihood estimate of $\mu_{\mathbf{x}}$, not just the minimizer of $||\mathbf{A}_c\mathbf{x} - \mathbf{y}_c||^2$ because $\mu_{\mathbf{x}}^*$ must also minimize $||\mathbf{A}\mathbf{x} - \mathbf{y}||^2$ (see Proof 2 in the Appendix), the minimizer of which we know is the least maximum likelihood estimate of $\mu_{\mathbf{x}}$.

B. Finding The Maximum Likelihood Noise Parameter

With the maximum likelihood abilities $\mu_{\mathbf{x}}^*$, we can predict whether team i will beat team j *on average* by determining if $\mu_i^* > \mu_j^*$. But to determine the *probability* with which team i beats team j, we need to estimate σ. To this end, note that $\mathbf{y}$ is an m-dimensional Gaussian vector distributed as $\mathbf{y} \sim N(\mu_{\mathbf{x}}, 2\sigma^2 I_{m\times m})$. Then

$$\begin{aligned}
\max_{\mu_{\mathbf{x}},\sigma} \log p(\mathbf{y}|\mu_{\mathbf{x}},\sigma) &= \max_{\sigma} \log p(\mathbf{y}|\mu_{\mathbf{x}}^*,\sigma) \\
&= \max_{\sigma} \log\left(\frac{1}{(2\pi)^{m/2}|2\sigma^2 I_{m\times m}|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*)^T(2\sigma^2 I_{m\times m})^{-1}(\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*)\right)\right) \\
&= \max_{\sigma} -\log\left((2\pi)^{m/2}2\sigma^m\right) - \frac{1}{4\sigma^2}(\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*)^T(\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*) \\
&= \min_{\sigma} m\log(\sigma) + \frac{1}{4\sigma^2}||\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*||^2.
\end{aligned}$$

Taking the derivative of the above with respect to σ, setting it equal to 0, and solving gives the maximum likelihood estimate

$$\sigma^* = \frac{||\mathbf{y}-\mathbf{A}\mu_{\mathbf{x}}^*||}{\sqrt{2m}} = \frac{||\mathbf{y}-\mathbf{A}\left(\mathbf{A}_c^T\mathbf{A}_c\right)^{-1}\mathbf{A}_c^T\mathbf{y}_c||}{\sqrt{2m}},$$

which is precisely the root mean squared error of the observed score differences (each of whose variance is $2\sigma^2$).

Since Gaussians are completely determined by their mean and variance, the maximum likelihood predicted outcome of a game between team i and team j is distributed as $N(\mu_j^* - \mu_i^*, 2\sigma^{*2})$. It follows that we would expect the score difference between team i and team j to be, on average, $\mu_j^* - \mu_i^*$, and that team j defeats team i with probability

$$P\{j \text{ defeats } i\} = \Phi\left(\frac{\mu_j^* - \mu_i^*}{\sqrt{2}\sigma^*}\right),$$

where Φ is the standard normal gaussian CDF function. More generally, we have a complete description of the predicted outcome between any two teams.

IV. Locally Weighted Linear Regression

We look to improve our predictions by giving increased importance to the most relevant observed outcomes. Consider a motivating example: Team A defeats team B by 10 points; team B defeats team C by 10 points; and team C defeats team A by 10 points. We are then asked to predict the outcome of a second meeting between team A and team B. Unweighted linear regression would assign each team an equal ability and thus would predict a draw in team A and team B's second meeting. This is indeed the most likely explanation of the data given the model. But one would think that team A's defeat of team B matters more than team B's transitive defeat of team A.

In predicting the outcome between team i and team j, we consider the most relevant observed outcomes to be the ones "closest" to the match-up between teams i and j, (i, j). To capture the notion of distance between games, we construct a graph in which each vertex represents a team and edges join teams that have played. Then to quantify the distance between match-ups (i, j) and (k, l), we compute the length of the smallest cycle containing i, j, k, and l (adding an edge between i and j's vertices and between k and l's vertices, if necessary). We denote this measure of distance $d_G((i, j), (k, l))$.

From these graph distances we can specify any number of weight schemes. In trying to predict the outcome of team i and team j, we choose to weight each observed outcome (k, l) as

$$w_{i,j}(k, l) = d_G((i, j), (k, l))^{-\alpha},$$

where we set $\alpha = 1$ because this value minimized win-loss bowl game prediction error over the past three seasons.

To predict $y_{i,j}$, we construct a diagonal matrix $\mathbf{H}^{(i,j)}_{m\times m}$ where the diagonal element corresponding to $y_{k,l}$ is set to $1/w_{i,j}(k, l)$. To enforce the $\mathbf{1}_n^T \mu_{\mathbf{x}} = 0$ constraint, we construct

$$(\mathbf{H}_c^{(i,j)})_{(m+1)\times(m+1)} = \begin{bmatrix} c & \mathbf{0}_m^T \\ \mathbf{0}_m & \mathbf{H}^{(i,j)} \end{bmatrix}$$

where $c \in \mathbb{R}$ is a very large constant. Then the maximum likelihood estimate of $\mu_{\mathbf{x}}^{(\mathbf{i},\mathbf{j})}$ for predicting $y_{i,j}$ in the weighted case is

$$\mu_{\mathbf{x}}^{(\mathbf{i},\mathbf{j})} = \left(\mathbf{A}_c^T \mathbf{H}_c^{(i,j)} \mathbf{A}_c\right)^{-1} \mathbf{A}_c^T \mathbf{H}_c^{(i,j)} \mathbf{y}_c.$$

Yet again, we must solve for the maximum likelihood parameter using $\mathbf{A}_c$ and $\mathbf{y}_c$ instead of $\mathbf{A}$ and $\mathbf{y}$ to ensure the matrix to be inverted is full rank. But this does not result in a worse-fitting $\mu_{\mathbf{x}}^{(\mathbf{i},\mathbf{j})}$ (see Proof 2 in Appendix). Instead of having a single $\mu_{\mathbf{x}}^*$ that gives an immediate ranking of the teams, we have distinct $\mu_{\mathbf{x}}^{(\mathbf{i},\mathbf{j})}$ for each outcome predicted. Returning to the motivating example, it could (and indeed should) be the case that $\mu_A^{(A,B)} > \mu_B^{(A,B)}$ but $\mu_B^{(B,C)} > \mu_C^{(B,C)}$ and $\mu_C^{(A,C)} > \mu_A^{(A,C)}$. So we have traded consistency of predictions for relevance of predictions.

V. Implementation and Results

A. Data Processing

We trained on the regular season data, and tested on the post-season bowl data. We processed this data by assigning each team an index, and removing any duplicate games (if team i plays team j, then team j also plays team i). Since data collection was time-intensive, we collected data for four seasons only.

B. Results and Analysis

The errors in table I show our unweighted and weighted linear regression algorithms have similar performance. Both found the 2009-2010 bowl games particularly difficult to predict.

	Unweighted Linear Regression				Weighted Linear Regression	
Season	Avg. Train	Avg. Test	W/L Train	W/L Test	Avg. Test	W/L Test
2007-2008	10.75	13.08	0.23	0.32	13.33	0.32
2008-2009	10.96	11.01	0.22	0.32	10.82	0.26
2009-2010	9.42	14.09	0.20	0.53	14.22	0.50

TABLE I
Summary of Errors

Win-loss error (W/L): The number of games whose outcome we predicted incorrectly, divided by the total number of games predicted.
Average absolute error (Avg.): The sum of the absolute differences between the actual score margin and our predicted score margin, divided by the total number of games predicted.

For each of the past three seasons, we trained our algorithm on the regular season data and tested on the bowl game data. We compared our algorithms' prediction success rate to the official BCS rankings, the official BCS computer rankings, and ESPN's power rankings. These rankings only has predictions for about half of the bowl games each season (which is why our algorithms' win-loss test error here differs from that of Table I).

Season	BCS Overall	BCS Comp. Avg.	ESPN Power Ranking	Us (Unweighted)	Us (Weighted)
2007-2008	0.41	0.47	N/A	0.53	0.53
2008-2009	0.27	0.33	0.27	0.53	0.67
2009-2010	0.40	0.40	0.40	0.40	0.47

TABLE II
COMPARISON OF RANKING SYSTEM WIN-LOSS PREDICTION SUCCESS RATE

Surprisingly, over the past three seasons, the experts' bowl predictions have been wrong more often than not. Also, our algorithms performed best – in terms of average training and test error, as well as win-loss training and test error – on the 2008-2009 season data, but this was the expert rankings' wost year. Both of our algorithms equalled or outperformed the expert rankings each of the past three seasons.

C. Predictions

Table III shows our predictions for this current season's upcoming major bowl games. For each game, we predicted the winner, the margin of victory, and also computed the probability of the win, derived from σ^*. Our two algorithms have made near-identical predictions, and both predicted a comfortable win (with probability 0.76) for second-ranked Oregon over first-ranked Auburn in the national title game.

Major Bowlgame Matchups		Winner	Point Margin (Unweighted)	Point Margin (Weighted)	Probability of Win
Auburn (1)	Oregon (2)	Oregon	9	9	0.7609
Wiconsin (5)	TCU (3)	TCU	7	7	0.6928
Arkansas (8)	Ohio State (6)	Ohio State	4	4	0.6245
Oklahoma (7)	Connecticut	Oklahoma	21	21	0.9462
Virginia Tech (13)	Stanford (4)	Stanford	11	10	0.7940

TABLE III
OUR PREDICTIONS FOR UPCOMING 2010-2011 MAJOR BOWL GAMES

VI. CONCLUSION

The structure of the NCAA football season poses some challenges to the formation of a "fair" ranking system. Since the vast majority of teams do not play each other, ranking them in a way that is fair and completely assesses their relative abilities is not an easy problem and has resulted in a lot of criticism of the current BCS rankings. Our algorithms predicted bowl game results for the past three seasons with much better accuracy than both chance and expert predictions. We expect our algorithms' predictions for the this season's upcoming bowl games to be similarly successful.

There are plenty of opportunities to extend our work. For one, we could explore different weight schemes for weighted linear regression. More broadly, since there nothing NCAA football-specific about our model or our algorithms, we can apply them to other sports leagues – especially those with more teams than games played per season – and even competitive activities beyond sports.

VII. Appendix

A. Proof 1

Given a vector of results $\mathbf{y}$, we can construct a graph G in which each vertex represents a team. An edge joins two vertices if and only if the two teams represented by those two vertices have played each other (and the outcome is contained in the dataset). We say the pool of teams is "competitively linked" if G is connected.

We claim that $\mathbf{A}_c$ is full rank if and only if G is connected.

If G is connected, then for any two teams i and j we have

$$\begin{aligned} x_i - x_{k_1} &= \pm y_{i,k_1} \\ x_{k_1} - x_{k_2} &= \pm y_{k_1,k_2} \\ &\vdots \\ x_{k_K} - x_j &= \pm y_{k_K,j}. \end{aligned}$$

Summing these equations,

$$y_{i,j} = x_i - x_j = \pm y_{i,k_1} + \pm y_{k_1,k_2} + \ldots + \pm y_{k_K,j}.$$

Now we can show $\mathcal{N}(\mathbf{A}_c) = 0$. So suppose $\mathbf{y} = 0$. Then for every i and every j

$$0 = y_{i,j} = x_i - x_j.$$

Hence $x_1 = x_2 = \ldots = x_n$. Then the constraint $\mathbf{1}_n^T\mathbf{x} = 0$ implies $x_1 = x_2 = \ldots = x_n = 0$. So $\mathbf{y} = 0$ implies $\mathbf{x} = 0$. Hence $\mathcal{N}(\mathbf{A}_c) = 0$ and $\mathbf{A}_c$ is full rank whenever G is connected.

If G is not connected, then we can partition the x_i into I and J where no team in I has played a team in J (and, of course, vice versa). Then define $\mathbf{x}' \in \mathbb{R}^n$ where $x'_i = 1/|I|$ if $i \in I$ and $x'_i = -1/|J|$ if $j \in J$. Clearly, $\mathbf{x}' \neq 0$ and $\mathbf{A}_c\mathbf{x}' = 0$, so in this case $\mathcal{N}(\mathbf{A}_c) \neq 0$. Hence if G is not connected, then $\mathbf{A}_c$ is not full rank. (This is what intuition would tell us: If the pool of teams is not "competitively linked," then there is no way of comparing connected component of teams to another.)

B. Proof 2

We claim that

$$\min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^n} (\mathbf{y} - \mathbf{A}\mathbf{x})^T\mathbf{H}(\mathbf{y} - \mathbf{A}\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^n} (\mathbf{y}_c - \mathbf{A}_c\mathbf{x})^T\mathbf{H}_c(\mathbf{y}_c - \mathbf{A}_c\mathbf{x}) = \min_{\mathbf{x}\in\mathbb{R}^n} J_c(\mathbf{x})$$

In the special case $\mathbf{H} = \mathbf{I}_{n\times n}$ and $\mathbf{H}_c = \mathbf{I}_{(n+1)\times(n+1)}$, this claim becomes

$$\min_{\mathbf{x}\in\mathbb{R}^n} ||\mathbf{y} - \mathbf{A}\mathbf{x}||^2 = \min_{\mathbf{x}\in\mathbb{R}^n} ||\mathbf{y}_c - \mathbf{A}_c\mathbf{x}||^2.$$

Note that

$$\min_{\mathbf{x}\in\mathbb{R}^n} J_c(\mathbf{x}) = (\mathbf{y}_c - \mathbf{A}_c\mathbf{x})^T\mathbf{H}_c(\mathbf{y}_c - \mathbf{A}_c\mathbf{x}) = (\mathbf{y} - \mathbf{A}\mathbf{x})^T\mathbf{H}(\mathbf{y} - \mathbf{A}\mathbf{x}) + (H_c)_{1,1}(0 - \mathbf{1}_n^T\mathbf{x})^2 \geq \min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x})$$

for all $\mathbf{x} \in \mathbb{R}^n$ since $(H_c)_{1,1} > 0$. So it suffices to show $\min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x}) \geq \min_{\mathbf{x}\in\mathbb{R}^n} J_c(\mathbf{x})$. To this end, let $\mathbf{x}' = \arg\min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x})$. Then define $a = (\mathbf{1}_n^T\mathbf{x}')/n$ to be the average of the entries of $\mathbf{x}'$. Using a and the fact that $\mathbf{1}_n \in \mathcal{N}(\mathbf{A})$:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} J(\mathbf{x}) &= J(\mathbf{x}') \\ &= (\mathbf{y}_c - \mathbf{A}_c\mathbf{x}')^T\mathbf{H}_c(\mathbf{y}_c - \mathbf{A}_c\mathbf{x}') \\ &= (\mathbf{y}_c - \mathbf{A}_c(\mathbf{x}' + a\mathbf{1}_n))^T\mathbf{H}_c(\mathbf{y}_c - \mathbf{A}_c(\mathbf{x}' + a\mathbf{1}_n)) \\ &= (\mathbf{y} - \mathbf{A}(\mathbf{x}' + a\mathbf{1}_n))^T\mathbf{H}(\mathbf{y} - \mathbf{A}(\mathbf{x}' + a\mathbf{1}_n)) + (H_c)_{1,1}(0 - \mathbf{1}_n^T(\mathbf{x}' + a\mathbf{1}_n))^2 \\ &= (\mathbf{y} - \mathbf{A}\mathbf{x}' + a\mathbf{A}\mathbf{1}_n))^T\mathbf{H}(\mathbf{y} - \mathbf{A}\mathbf{x}' + a\mathbf{A}\mathbf{1}_n) + (H_c)_{1,1}(\mathbf{1}_n^T\mathbf{x}' - a\mathbf{1}_n^T\mathbf{1}_n)^2 \\ &= (\mathbf{y} - \mathbf{A}\mathbf{x}')^T\mathbf{H}(\mathbf{y} - \mathbf{A}\mathbf{x}') + (H_c)_{1,1}(an - an)^2 \\ &= (\mathbf{y} - \mathbf{A}\mathbf{x}')^T\mathbf{H}(\mathbf{y} - \mathbf{A}\mathbf{x}') \\ &= J_c(\mathbf{x}') \\ &\geq \min_{\mathbf{x}\in\mathbb{R}^n} J_c(\mathbf{x}) \end{aligned}$$