The Blue Alliance Blog

Analysis, stories, tips, and updates related to FIRST Robotics and The Blue Alliance

OCTOBER 5, 2017
EUGENE FANG

The Math Behind OPR — An Introduction

OPR, or Offensive Power Rating, is a stat that's often used in the FIRST community to compare the performance of teams on the field. It's also common to hear things like "this game is a bad game for OPR" or "OPR is a pretty accurate indicator of performance this year." In order to understand OPR's strengths and limitations, learning the math behind it is key. This blog post will provide an introduction to the math behind OPR, and a future post will delve into how game design affects OPR's usefulness as a statistic.

Note: While this post is mostly targeted at people who have at least taken Algebra 1, having experience with matrices will help a lot. Regardless, hopefully this post sheds some light on the fact that matrices are much more than just a clever way to arrange numbers. Pay attention in your future linear algebra classes!

Introduction

OPR is now the more widely used term for what Karthik from Team 1114 once called "calculated contribution (https://youtu.be/l8syuYnXfJg?t=6m49s)" — OPR attempts to calculate a team's contribution to their alliance's final score. If there were no alliances in FRC, a team's calculated contribution would simply be their average match score. This is an easy and straightforward metric to calculate. However, using average score with alliances, while giving you a positive predictor, isn't the most useful since the average score is a combination of the performance of all three teams on the alliance. 1114 starting using OPR in 2004 because they wanted a more useful metric to evaluate how teams had been playing than average score. They wanted a way to figure out, on average, what each team was contributing to their alliance's score. Here, OPR makes a key assumption: an alliance's final score is a linear combination (https://en.wikipedia.org/wiki/Linear_combination) of each alliance member's individual contribution. In other words, adding together the points earned by each member on a 3-team alliance results in the alliance's final score. As an equation, this looks like:

$$A + B + C = score$$

Here, A, B, and C are each team's contribution to their alliance score. But how do we calculate A, B, and C? The answer is this: solve a system of equations.

A basic example

To simplify things, let's imagine that alliances are made up of 2 teams instead of 3. Also, let's only focus on one alliance in a match. Let's say that teams A and B scored 10 points together. This can be described by the equation:

$$A + B = 10$$

At this point, we still can't calculate how many points each team scored. Did A score 8 and B score 2? Did A score 10 and B score 0? Did both teams score 5 points? Currently, we have an underdetermined system (https://en.wikipedia.org/wiki/Underdetermined_system) with more variables (teams) than equations (match results). We need more information to find values for all the variables. Luckily, as the competition goes on, we get more matches to work with. Let's say more matches are played with the following results:

$$A + C = 13$$
$$B + C = 7$$

Now we have 3 equations and 3 unknowns — we should be able to solve for A, B, and C! With some basic algebra (or having a computer solve it (http://www.wolframalpha.com/input/?i=a%2Bb%3D10,+b%2Bc%3D7,+a%2Bc%3D13)), we find that team A's calculated contribution (OPR) is 8 points, B's is 2, and C's is 5. Congratulations you know now how to compute OPR! But wait, there are a few problems with this.

An overdetermined system

Let's try another example with more matches:

$$A + B = 10$$
$$A + C = 13$$
$$B + C = 7$$
$$A + D = 15$$

With 4 equations and 4 unknowns, we can solve this like before. The solution (http://www.wolframalpha.com/input/?i=a%2Bb%3D10,+b%2Bc%3D7,+a%2Bc%3D13,+a%2Bd%3D15) yields A=8, B=2, C=5, and D=7. However, this will only continue to work as more equations are added if all teams are perfectly consistent and each team scored the same number of points in their respective matches. As we know in FRC, robots break, teams are inconsistent, and sometimes the game itself may be the problem (but that's a topic for another blog post). Let's add the following match result:

$$B + D = 10$$

Trying to solve (http://www.wolframalpha.com/input/?i=a%2Bb%3D10,+b%2Bc%3D7,+a%2Bc%3D13,+a%2Bd%3D15,+b%2Bd%3D10) this system of equations yields "no solution." Now we have an overdetermined system (https://en.wikipedia.org/wiki/Overdetermined_system) with 5 equations and 4 unknowns — no values of A, B, C, and D can simultaneously satisfy all equations (in general). Instead of finding values that perfectly satisfy all equations, we now can only find an approximate solution that minimizes the error across all the equations. To do that, we turn to matrix algebra.

Equations as matrices

Let's go back to our simple example. It can be written out in matrix form as:

Applying matrix multiplication (https://en.wikipedia.org/wiki/Matrix_multiplication#Illustration), we can see that the matrix form is the same as our original set of equations:

$$1A + 1B + 0C = 10$$
$$1A + 0B + 1C = 13$$
$$0A + 1B + 1C = 7$$

If we have n teams and m matches (remember, we're still only looking at one alliance per match), the system of equations takes the form of $Mx=s$. Here, M is a $m \times n$ matrix where each row indicates which teams were in an alliance together in a given match (it should only be 1's and 0's — 1 if a team was in the match and 0 if they weren't), x is a $m \times 1$ vector of OPRs (variables that we are trying to solve for), and s is a $m \times 1$ vector of alliance scores.

One way to solve for x is to left-multiply both sides by M^{-1}, the inverse (http://mathworld.wolfram.com/MatrixInverse.html) of matrix M. In practice, you wouldn't necessarily do this. The inverse of M doesn't always exist, and there are other faster and more numerically stable (robust to computer rounding errors) ways to solve for x, especially as M becomes very big (many teams, many matches). However, we are using it here to demonstrate that we get the same solution as before. Applying this inversion (https://www.wolframalpha.com/input/?i=inverse+%7B%7B1,+1,+0%7D,+%7B1,+0,+1%7D,+%7B0,+1,+1%7D%7D), our matrix equation looks like:

$$M^{-1}Mx = M^{-1}s$$
$$x = M^{-1}s$$

Multiplying it out, we see that the solution is the same as before: A=8, B=2, C=5.

Least squares solution

Great, now we can solve the equations in matrix form. But how does that help when the system is overdetermined? Let’s take a look at our overdetermined system again:

$$A + B = 10$$
$$A + C = 13$$
$$B + C = 7$$
$$A + D = 15$$
$$B + D = 10$$

In matrix form, we have:

$$Mx = s$$

Like before, we cannot solve this system of equations. To remedy this, we can multiply both sides by the transpose (https://en.wikipedia.org/wiki/Transpose) of M and try again.

$$M^T Mx = M^T s$$

When we left-multiply both sides of the overdetermined system ($Mx = s$) by the transpose of the binary design matrix (M^T), we are creating what is called the “Normal Equation (http://mathworld.wolfram.com/NormalEquation.html).” The solution of the Normal Equation is the “least squares solution” of the overdetermined system (more on what “least squares” means later). There’s a proof for this, but that’s out of scope here.

Let’s plug in numbers and see what this looks like:

Now we can solve for x however we like! For simplicity, let’s convert this back into separate equations and solve (http://www.wolframalpha.com/input/?i=3*a%2Bb%2Bc%2Bd%3D38,+a%2B3*b%2Bc%2Bd%3D27,+a%2Bb%2B2*c%3D20,+a%2Bb%2B2*d%3D25).

$$3A + 1B + 1C + 1D = 38$$
$$1A + 3B + 1C + 1D = 27$$
$$1A + 1B + 2C + 0D = 20$$
$$1A + 1B + 0C + 2D = 25$$

And there we have our OPRs! A=7.75, B=2.25, C=5, D=7.5

Okay, so what exactly is the least squares solution?

We just solved the Normal Equation, which means the result is the least squares solution of the original problem. Since there was no exact solution to the overdetermined system, we instead solved for the “best” approximation of the solution. The “best” solution can be defined in different ways (e.g. least absolute deviation, minimum mean squared error, etc.). For OPR, least squares the de facto definition of "best." In other words, we found the solution (values for A, B, C, and D) that minimize the sum of squared errors across all our equations. Let's look at our original equations again:

$$\begin{gathered} A + B = 10 \\ A + C = 13 \\ B + C = 7 \\ A + D = 15 \\ B + D = 10 \end{gathered}$$

Plugging in our solution, we get:

$$\begin{gathered} 7.75 + 2.25 = 10 + error \\ 7.75 + 5 = 13 + error \\ 2.25 + 5 = 7 + error \\ 7.75 + 7.5 = 15 + error \\ 2.25 + 7.5 = 10 + error \end{gathered}$$

The equations are no longer perfect — they have some error (0, -0.25, 0.25, 0.25, and -0.25 respectively). If we sum the square of these errors, we get:

$$0^2 + (-0.25)^2 + (0.25)^2 + (0.25)^2 + (-0.25)^2 = 0.25$$

0.25 is the minimum this sum of squared errors can be. Pick any other values for A, B, C, and D, and it will be larger. For example, let's choose A=7, B=3, C=5, D=7.5:

$$\begin{gathered} 7 + 3 = 10 + error \\ 7 + 5 = 13 + error \\ 3 + 5 = 7 + error \\ 7 + 7.5 = 15 + error \\ 3 + 7.5 = 10 + error \end{gathered}$$

Now, our errors are: 0, -1, 1, -0.5, and 0.5 respectively. Calculating the sum of the square of these errors, we get:

$$0^2 + (-1)^2 + (1)^2 + (-0.5)^2 + (0.5)^2 = 2.5$$

These new values for A, B, C, and D are worse than our optimal least squares solution; we get a higher sum of squared errors: 2.5 vs. 0.25.

OPR vs. average match score

Looking back to the original goal of finding a metric that more accurately conveys a team's individual contribution than average final score, we see that OPR attempts to mathematically separate a team's contribution from their alliance's overall output. One neat fact about OPR is that if you take the average OPR of all teams at a regional, the result should be equal to the average match score of all teams at the event divided by 3. This makes logical sense since three teams with an average OPR should combine to give you an average match score. Another point to consider, it is possible to have negative OPRs. This means that on average a team's presence on the field actually brings down the alliance's score. This is uncommon but used to pop up from time to time, especially when alliance penalties lowered your own score instead of increasing your opponent's score.

Some Notes

Here, we simplified the problem by only looking at one alliance per match, with two teams on that alliance. For real-world 3v3 matches, this only changes a few things (and none of the math). Instead of M and s having the same number of rows as the number of matches, they would have two times the rows (one for each alliance per match). Also, the M matrix would always have three 1's per row instead of two, since 3 teams are playing.

For very large systems of equations (e.g. calculating "World OPR" using all 15K+ matches from a season), using sparse matrices can dramatically speed up computation.

Conclusion

At its core, OPR is just solving a system of equations — something taught in a basic algebra class. However, when there are more equations than variables (more alliance matches than the number of teams), the system, in general, becomes overdetermined. OPR finds the least squares solution to this overdetermined system by finding the solution to the Normal Equation form of the system.

Many thanks to Karthik and Ether for their input on this post.

Edit: Jaci posted a great follow-up (http://imjac.in/ta/post/2017/10/08/oprs-and-least-squares-linear-algebra.html) that goes into more depth on the linear algebra that was mostly glossed over here. Be sure to check it out!

STATISTICAL ANALYSIS

OPR **STATS**

Published by Eugene Fang

I'm currently pursuing a graduate degree at CMU's Robotics Institute. I'm fascinated by the interplay between hardware and software. View all posts by Eugene Fang

One comment

1. **Seth** says:
   DECEMBER 13, 2017 AT 7:58 AM
   I appreciate the time you've taken to teach us about this stat and how it's calculated. If it's not too much trouble, could you do a post about DPR and how it's calculated? Thanks!

   REPLY