Abstract

A nomogram is a graphical representation of a mathematical relation in multiple variables, where each variable is represented by a curve and relations between the variables are derived from straight lines intersecting all curves. They were once used in many fields as simple calculating tools before the advent of computers, because they made useful calculations available to laypeople who didn’t know the mathematics behind the equation or didn’t even know what equation was being used. In this paper, we present an modern implementation of nomograms in D3 that allows for interactive exploration of chosen equations. Even though printed nomograms have been rendered obsolete by the computer and calculator, interactive nomograms can still be useful to explore mathematical relationships and visualize equations. With a simple, clear interface to visualize many different equations, the visualization can easily demonstrate the fundamental relationship between multiple variables, giving the user a clearer understanding of the equation involved. As a learning tool in physics, math, and chemistry, a nomogram can be a powerful teaching tool to visualize multivariate equations, an otherwise difficult to visualize concept.

Keywords: nomograms, mathematical visualizations, linear charts, parallel coordinates

1 Introduction

Nomograms were invented in the latter half of the 19th century and were used for over 75 years as a method for simple calculations. A sailor, for example, could use a nomogram to calculate distances to travel for his ship without even knowing the underlying formula. More complicated nomograms could even be used to calculate risk probabilities and diagnose patients.

Since computers have rendered analog computational tools like the nomogram obsolete, nomograms are no longer used as computation tools, but remain valuable as visualization devices for mathematical equations. With a nomogram visualizing an equation, it is easy to see how changes in one variable affect the value of another variable, which facilitates understanding the equation in a deeper way than just looking at the equation.

However, most nomograms are static, and it’s up to the user to use a straightedge to manually figure out how the values relate, as users once did when nomograms were invited. Also, there is currently no easy way to generate nomograms dynamically and intuitively. We developed a tool to generate nomograms easily and interact with the result, so that users can gain a deeper understanding of the equation than with a static nomogram.

2 Related Work

There have been several papers and articles published on the subjects of nomograms, nomogram generation, and real world applications of nomograms. Ron Doerfler has published papers regarding the mathematics behind creating nomograms for equations of various types [1, 2]. Furthermore, Marasco et al. published a paper discussing a useful application for nomograms in helping decide course of action for treatment of rare diseases.

PyNomo [5], developed by Leif Roschier, is a Python library to create static nomograms in PDF form. While this tool can generate nomograms of many different forms other than the simple parallel-scale nomogram, all of the generated nomograms are static images and do not support digital user interaction.

Currently, the only nomogram generator we have found with
a graphical user interface was a Java applet created by Jones et al. [3]. However, the generator is limited to only a few equation types, and has an inefficient and confusing interface. Also, the generator allows users to draw lines interactively, but does not allow the user to move the lines, points, or axes around dynamically.

3 Methods

Our project aims to add interactivity to nomograms. We built an online tool for users to easily generate nomograms from inputted mathematical relations. The user can then interact with the nomograms to explore relationships between variables. We implemented this visualization in JavaScript using D3, which means that anyone with a browser can use our system to explore.

To generate the nomogram, we start with a linear equation in three variables, where the variables could be logarithms. Suppose that the equation given were \( a \cdot x + b \cdot y + c = d \cdot z \) (the logarithmic case is similar). If \( z \) is the variable dependent on \( x \) and \( y \), then we place the \( x \) variable axis on the far left, and the \( y \) variable axis on the far right. Each axis goes either from low to high if the variables coefficient is positive, or is flipped from high to low if its coefficient is negative.

In order to determine the distance between the variables axes, we looked at the ranges of each variable as well as the coefficients in the linear relation. To calculate the range of the \( z \) variable, we calculate the minimum and maximum values \( z \) could be given the ranges of \( x \) and \( y \). Given these constraints, there is only one possible place for the \( z \) variable axis between the \( x \) variable axis and the \( y \) variable axis such that every straight line that intersects all three axes represent a solution to the given relation.

Using similar triangles (see figure), we figured out that the ratio of the distance between the \( z \) and \( y \) axes to the distance between the \( x \) and \( y \) axes is given by

\[
\frac{b \cdot (y_{\text{max}} - y_{\text{min}})}{d \cdot (z_{\text{max}} - z_{\text{min}})}
\]

where \( y_{\text{max}} \) and \( y_{\text{min}} \) are the maximum and minimum of the range of \( y \), and \( z_{\text{max}} \) and \( z_{\text{min}} \) are the maximum and minimum of the calculated range of \( z \).

Given the equation and the ranges of two of the axes, we can use similar triangles to compute the range of the third axis as well as the position of the dependent axis. These have unique solutions because the nomogram must maintain its linear spatial relationship at all times. In other words, no matter where the user drags the axis handles, the equation must still hold. Therefore, there is a unique configuration of the nomogram that will support this constraint. Once solved, our nomogram uses the computed specifications to render the nomogram.

Figure 1: Calculation of dependent variable axis range and position

Our visualization supports the following features:

3.1 Free-form user input

We decided to allow users to input relations through a single input box because it was the simplest option for the user. The other option we considered was to break down relations into parts, and ask the user to input each part of the relation into separate input boxes (coefficients, constants, variable names, etc.). However, this requires the user to manually translate from the relation to its constituent parts before generating the nomogram. Jones’s nomogram applet [1] used this approach for prompting the user for an equation. Instead, we chose to parse the input ourselves to make this step much friendlier for the user.

We first split the equation based on the equals sign. Then we used regular expression matching to parse both sides of the equation, matching it to one of our solvers. Each of our solvers handles equations of a different type, and generates the appropriate ranges, axes, and positions. For this project, we implemented a linear solver and a logarithmic solver, which were the most common equations types.

3.2 Variety of equation types

3.2.1 Linear equations

We supported linear equations of the form \( a \cdot x + b \cdot y + c = d \cdot z \), where \( x \), \( y \), and \( z \) are variables, and \( a \), \( b \), \( c \), and \( d \) are known, non-zero constants (which could be negative). Each constant may be optionally omitted from the equation.
3.2.2 Logarithmic equations

We also supported non-linear equations of the form \( c \cdot x^a \cdot y^b = z^d \). In order to display equations of this form on a parallel-scale nomogram, we transformed this equation to a linear equation in \( \log x \), \( \log y \), and \( \log z \) to become
\[ a \cdot \log x + b \cdot \log y + \log c = d \cdot \log z. \]
As above, \( a \), \( b \), \( c \), and \( d \) are non-zero constants which may optionally be provided.

3.3 Suggested equations to explore

Although there are only two general types of equations we support, there are many equations that fit into these categories. We took some useful equations from physics and added them as suggested equations to explore on our nomogram generator. Some examples are
\[ E = 0.5 \cdot m \cdot v^2 \] (the equation for kinetic energy),
\[ d = r \cdot t, \] (the equation for distance as a function of rate and time), and
\[ a = \frac{v^2}{r} \] (the equation for radial acceleration as a function of velocity and radius).

3.4 Draggable points for each variable

The main way we added interactivity to nomograms was by allowing users to drag points along each variable axis. As a point is dragged along an axis of a three-variable nomogram, one of the other points is fixed while the other one is changed according to the dragged point, such that all three points lie on a line. As the user drags a handle, the values for all three axes are continuously updated, so that the user can observe the change resulting from their actions.

We prevent the user from moving any point past either endpoint on its axis. In particular, if moving one variable too far would cause another variables point to move past the endpoints of its axis, we did not allow the user to drag the first variable past that sub-interval on the scale.

3.5 Fixed point selection

We also allow the user to choose which variable should be fixed when other points are dragged. If a variable is fixed, then dragging other lines will pivot the line around that point. We used a lock icon to indicate that a variable is fixed, and only one variable can be fixed at a time. If the user does not explicitly fix a variable, we automatically choose one of the endpoints of the line to keep fixed.

This feature is helpful for computations involve meeting constraints, because the user can choose the constrained variable and then explore how the other two variables change to accommodate this constraint. For visualizing changes in variables for multidimensional functions like these, a nomogram can be used for understanding how each variable has an impact on the final result, and vice versa.

Figure 3: The same equation \( p = i^2 r \) depicted before and after changing the scale. Note the movement and change in range of the dependent axis.
3.6 Customizable variable ranges

Finally, although on initial nomogram generation we pick default ranges for all variables, we included the interactive feature of allowing the user to change the range of each variable. We put clickable arrows next to each axis which will expand the range to include more numbers on either side of the range through a smooth animation. We also allow the user to drag each axis up or down to translate the range of the variable. While the user drags, the nomogram line representing the relation moves accordingly.

For every move, we recalculate the range and position of the dependent variables axis according to the formula explained earlier. This maintains the linear nature of the nomogram regardless of the range of each axis, so that the nomogram will continue to be valid. Should the user drag an axis so that the axis handles would fall off the valid range, we clamp the handles to stay with the visualization, so that the user can continue to interact with the visualization.

Some ranges are invalid for certain axis scales – for example, a range containing any nonpositive values will be invalid for a logarithmic scale. For these, we clamp the dragging and scaling so that the user cannot go past the valid edge of the scale’s domain.

4 Results

From informal studies, we determined that users quickly became familiar with the interface and understood intuitively how to use the visualization. Even though the locking and dragging features are not emphasized in the interface, users instinctively knew how to use them, which was a good sign that our interface was intuitive.

Our users thought that the ability to drag the variable axes was really useful and fun to play with. Some were interested to know the mathematics behind the nomogram generation process, because they moved the visualization into a strange configuration. In particular, they wanted to know what governed the scales and ranges of each axis and the position of the middle axis in relation to the others. This was an unexpected artifact of the interactive nomogram process, because as users interacted with and moved the nomogram axes around, they became fascinated with the visualization itself, rather than the content of the visualization.

Although the users seemed to enjoy looking at the physics equations our tool suggested, they also often wanted to enter equations not supported by the system. Some of these equations were equations we simply hadn’t implemented a solver for yet, and others were impossible to visualize with a nomogram because they could not be mapped to a simple linear model, which is all that the standard three-line linear nomogram can support.

5 Discussion

We think that users found interactive nomograms to be an easy and intuitive way to see how variables influence each other in equations. Interactive nomograms are a more graphical and intuitive way of representing the same information as traditional multiplication tables (e.g., with the equation $x \cdot y = z$) or even tables for more complicated relations.

Nomograms give a hands-on approach to equation visualization, which evokes a deeper understanding of the equation, rather than just looking at a plot. Allowing the user to manipulate the variables on each axis help reveal the relationships between them. For example, if the middle axis is closer to the right axis than to the left axis, then a user can easily see that moving the point on the right axis by a fixed pixel distance will affect the dependent variable on the middle axis more than moving the point on the left axis by the same pixel amount.

For functions of two or more variables, looking at a plot or graph because infeasible, leaving nomograms as one of the best ways to visualize multivariate equations. Just performing some calculation is trivial to do a calculator, but what a nomogram can explain that a calculator can’t is how those values work together to achieve a final result, and how changing those values perturbs the result.

For example, in the equation $v^2 = 2gh$, which governs the maximum height of a projectile thrown upwards in a gravity field of acceleration $g$ with $v$ velocity and no air resistance, it is very difficult to intuitively grasp how these values work together. But with a nomogram’s visualization, the numbers come together to tell a story about how increasing velocity actually increases the thrown height by the square of the velocity, and how given a fixed height, one can find many combinations of velocity and gravity that will get a projectile to that height. These would require significant mathematical acrobatics to intuit just from looking at the equation, but a nomogram makes it clear instantly.

6 Future Work

There are many possible extensions that we could add to the nomogram generator that we built. Possible new features that we could add are allowing the user to enter in specific values for variables, automatically extending axes when points get dragged to the endpoints of their axes, or allowing users to drag axes around while updating the axes ranges automatically.
Figure 4: Linear-scale nomogram with the equation $a + b = c$

Figure 5: Logarithmic-scale nomogram with the equation $v^2 = 2gh$ (the formula for the apex height of a projectile)
We would also like to be able to generate intelligent explanations for the relations that users enter – for instance, for the relation \( d = r \cdot t \), being able to generate the explanation “If you drive for 1.5 hours at 55 miles an hour, you will go 82.5 miles.” This is useful to integrate the mathematics more tightly with the deeper meaning of the equation, and to make the numbers outputted by the nomogram more relevant.

Additionally, it would be useful to support nomogram generation for equations with more than three variables, as well as more complicated three-variable equations. To accommodate more complex equations, we would like to implement a more sophisticated equation parser instead of using regular expressions.

Finally, we would like to support a more diverse set of nomograms. In addition to parallel-scale nomograms, there are many other forms of nomograms including N-shaped nomograms or nomograms with curved axes. In the future, we would like to be able to automatically generate as well as provide interactivity for these more complicated nomograms given non-linear equations of more than three variables.

7 Conclusion

We began this project knowing very little about nomograms, but after implementing a complete nomogram system, we’ve learned a great deal about how nomograms are generated and how interactivity plays a huge role in making nomograms relevant to a modern user. The mathematics proved to be more complicated than we expected, especially because of our interactivity goals – dragging an axis while a handle was locked proved to be extremely difficult, which is probably why other systems have not introduced the kind of interactivity we were looking for.

We view nomograms as a fascinating tool for mathematical exploration, a great visualization of otherwise cold and remote equations. Often, knowing an equation is only a fraction of true understanding of the big picture – everyone knows \( E = mc^2 \), but understanding how those numbers interact and describe the universe that we live in is significantly trickier.

8 Acknowledgements

We’d like to thank the CS448B staff for running a great class and Jeffrey Heer for teaching it. The concepts we’ve learned will (hopefully) continue to be useful for years in the future.

9 References


