Learning to Sail: A Data Driven Approach

Abstract:
This report presents a data driven approach to the development of a sail trim controller, with an eye towards applications in both autonomous and high performance sailing. Designing a control algorithm based on the underlying physical model of sailing is difficult due to the complex fluid dynamics involved; instead, we can analyze data collected while the boat is sailed under manual control to determine optimal sail position and export this knowledge to an autonomous stack in a condensed manner. For uniformity, and because various sailboats behave differently under the same external conditions, I explicitly define the underlying speed model and generate noisy pseudo-data on which to train the controllers. Three controllers, based off of random forests, linear regression with basis expansion, and a modified K Nearest Neighbors (KNN) algorithm, were developed for a simplified, two control variable version of this problem, and all perform reasonably well in this task. As a check for robustness, these algorithms were then applied to data generated from a qualitatively different speed model and compared for accuracy. The results suggest that some forms of data driven development may be robust to such changes and are a viable approach to sail control across a variety of sailboats.

Background:
Contemporary sailboats harness the wind using wing-like sails and underwater foils in balance with each other. Lateral “lift” from the sails generates sideways force with a small forward component, and the sideways motion through the water generates a counteracting lift force on the foils, resulting in cancellation of the majority of the lateral force and a net forward motion (fig. 1). Varying wing (i.e., sail and foil) shape enormously impacts the force balance of the boat; it is not simply a matter of putting the sail in the way of the wind and being pushed along. High performance sailboats often have upwards of 10 controls to change various aspects of the wing shape. Because CFD simulations of sail- and foil-plans are computationally expensive and largely non-transferable across different makes of boat, and because high performance sailing is a fairly niche field that has only just begun to attract the attention of major aerodynamics companies like Airbus and Boeing, there are relatively limited quantitative descriptions of the effects of sail adjustments. Without this quantitative understanding, physics-driven control models are difficult to develop.
Figure 1: A birds-eye force diagram of a sailboat. Air (brown lines) bends around the sails, generating lift towards the top right corner of the image (see $F_{\text{sail}}$ in subdiagram A). The resulting relative water flow (blue lines) bends around the foils (in this case, the keel), generating a resultant force $F_{\text{keel}}$ towards the left of the page. In aggregate, the two lift forces produce force in a mostly-forward direction.¹

One of the most critical sail controls is the angle of attack, measured as the angle from the sail to the centerline of the boat. For two-sailed boats, which I will consider for the remainder of this paper, the two angles of attack are controlled independently. Although high performance sailors adjust these angles based on multiple environmental factors including current, other boats, and sea-state, the predominant factors by far are wind speed and wind angle relative to the sailboat. The simplified version of our sail control problem then becomes determining the two angles of attack to generate the maximum forward boat speed given wind speed and wind angle.

Specific Applications:

Recreational sailors use simple heuristics, collectively known as the Points of Sail (fig. 2), to determine sail angle. However, there are times when human control is not desirable, such as:

¹ Image Source: [https://www.comsol.com/blogs/physics-of-sailing-cfd-analysis/]
1) In Autonomous Sailboat Design:

Although still somewhat of a novelty at this point, autonomous sailboats offer the potential for low-energy oceanic observation and search and rescue. The Yale Undergraduate Intelligent Vehicles team has spent the past two years developing a two meter long robotic sailboat designed to compete in the International Robotic Sailing Regatta. The boat, “Ratchet,” (fig. 3) is focused on task completion and basic sailing ability over performance, so angle of attack is the only control variable for each sail. Ratchet boasts a full on-board computational stack, accurate wind and speed collection, and options for manual control, so it is an ideal platform for which our simplified sail control problem can be considered.

---

2 Image source: https://knotalotsailing.wordpress.com/sailing-101/points-of-sail/
2) In High Performance Sailing Analytics:

As high performance sailing pushes the limits of materials and design, subjective human control is increasingly seen as fallible. Grand prix boats, including TP52's like Spookie (fig. 4) collect and analyze close to ninety points of data every second in an attempt quantify the entire state of the boat. As data collection becomes more comprehensive, analysis of what works and what doesn't becomes more feasible. Control algorithms and data-driven velocity prediction algorithms are two ways to summarize this data and leverage it for improved performance.

---

3  Private images graciously provided by Yale Undergraduate Intelligent Vehicles.
4  Private image graciously provided by Steve Benjamin and the Spookie team.
**Methodology:**

Originally, this project intended to explore the viability of data driven development of sail controllers using data exclusively collected on Ratchet, Yale Undergraduate Intelligent Vehicle's autonomous sailboat. Five variables, wind speed, relative wind angle, mainsail (rear sail) angle, jib (forward sail) angle, and boat speed, would be collected. By sailing the boat in a variety of different wind conditions and exploring the full range of control variable input, the resultant data should be sufficient for developing a sail controller.

A pseudo-data generator was developed order to parallelize Ratchet's manufacturing process and the development of these control algorithms. The data from this code uses simple heuristics that are qualitatively correct in nature, even if quantitatively different than the data that would be collected on Ratchet. A cross section of the data can be viewed in Figure 5. When Ratchet's engineering schedule was delayed to the point that data collection became infeasible, this heuristic model became the central data source for this investigation.

**Figure 5:** (Top) A visualization of a cross-section of the generated pseudo data. $\Theta$ is the relative angle of the wind, while $r$ shows the resultant boat speed. The slider at the bottom controls wind speed and can be varied interactively. Clicking on a data point displays the other two variables, main and jib angle, in the inspection pane on the right hand side. (Bottom) The optimal controller given the underlying speed model used for data generation. All the axes are the same, except that there is no inspection pane and the sail position is printed on the plot every 20 degrees. Because the underlying speed model was written using basic sailing heuristics, it is not surprising that this control strategy is qualitatively similar to the “Points of Sail” rules discussed in Figure 2. All visualizations were developed in-house as part of this project.
Although machine learning algorithms designed for classification and regression have been developed extensively, there is less literature discussing applications to control problems. Because I expressly try to avoid understanding the underlying physics of the system, there isn't an obvious way to obtain a derivative of boat speed with respect to any of the input or control variables, meaning that it cannot easily be incorporated into a loss or cost function. This problem is addressed in two different ways. The modified K-nearest neighbors algorithm uses a distance function based solely on the input variables but weights the nearby points based on their resultant boat speed, thus favoring fast nearby points. The other two algorithms (based off of random forests and linear regression with basis expansion) re-frame the question as a regression problem; models are trained to predict boat speed from the input and control variables, and the learned function is then subjected to constrained optimization.

\[\theta\] is the relative angle of the wind, while \(r\) shows the resultant boat speed. The slider at the bottom controls wind speed and can be varied interactively. Sail trim corresponding to these strategies are displayed periodically on the plot.

We see that the learned strategies (red) tend to perform close to the optimal strategy (black), with the exception of some boundary-related inaccuracy in the KNN controller. All visualizations were developed in-house as part of this project.

**Figure 6:** Optimal versus learned control strategies for KNN (Top Left), Linear Regression (Bottom Left) and Random Forest (Top Right).
Having access to the underlying data generator affords us the benefit of being able to compare our learned results to an optimal solution, something that is impossible in real life where the optimal solution isn't known or identifiable. I used the mean squared difference in boat speed between the learned controller and the optimal controller as a performance metric, which works well except for the fact that it is scaled to the somewhat arbitrary pseudo-data values. All three algorithms performed fairly well: KNN had a fit of 0.069, Random Forests had a fit of 0.109, and Linear Regression had a fit of 0.072. These values were obtained after testing across various algorithm parameters and settling on 1000 neighbors, 100 trees in the random forest, and seven degrees of basis expansion, respectively.

Visualizations of the three control strategies are presented in Figure 6. The only major feature of interest occurs in the KNN controller near 180 degrees from the wind. There is a steep performance dropoff caused by the natural 180 degree boundary, which I believe could be resolved with more training data.

**Next Steps: Robustness**

Because optimal control strategy differs significantly from boat to boat, it is important that the methods developed here be robust to changes in the speed model. This investigation is somewhat uncommon in that we have the ability to adjust the underlying physical model and observe its effect on the learning algorithms. There are many changes that we might make. In high performance sailing, boats are looking for the last hundredth of a knot of boat speed; getting it 99% right doesn't cut it. A model could be developed that only slightly penalizes close to optimal configurations to see how well algorithms can still truly identify and output optimal behavior. Scales could also be changed; some boats move faster than others overall. Some are particularly sensitive to one sail's tuning and not the other. The list goes on.

As one such experiment, I developed a secondary speed model that reflects the ability of some boats to plane (fig. 7). Planing occurs when a boat's weight is supported by hydrodynamic lift rather than buoyancy and is characterized by a sharp reduction in drag and jump in boat speed. This model highly rewards near optimal control with a large bonus (getting onto a plane), while other configurations are treated similarly to the previous models.
Figure 7: (Left) A 505 planing downwind. The hull is raised partially out of the water by hydrodynamic lift, resulting in reduced wetted surface and increased maximum boat speed. (Right) Data generated from an underlying planing model, where optimal and nearly optimal control result in planing and drastically increased speed compared to other configurations. $\Theta$ is the relative angle of the wind, while $r$ shows the resultant boat speed. The slider at the bottom controls wind speed and can be varied interactively. Clicking on a data point displays the other two variables, main and jib angle, in the inspection pane on the right hand side. All visualizations were developed in-house as part of this project.

All three learning algorithms perform worse than they did on the original speed model, as is to be expected (fig. 8). Even so, visual inspection of the resulting controllers reveals that all three algorithms correctly identify and reproduce some parts of the planing regime. Linear regression with basis expansion performs particularly well, with significant errors only at a wind angle of about 160 degrees. KNN and Random Forests, which produce a more piecewise output than linear regression, struggle to smoothly control sails as wind angle varies and accordingly crash between planing and non-planing states. Because the training data is theoretically generated from human sailors controlling the boat, one potential remedy to this is to extensively collect data in the planing regime. Modifying the data generator in this manner improved performance to a small degree, but linear regression with basis expansion remains the algorithm most robust to changes in the underlying model that was developed for this project.

---

5 Image source: http://www.505tanktalk.com/2015_05_01_archive.html
Figure 8: Optimal versus learned control strategies for KNN (Top Left), Linear Regression (Bottom Left) and Random Forest (Top Right) for data generated using an underlying planing model. $\Theta$ is the relative angle of the wind, while $r$ shows the resultant boat speed. The training data is scattered on the plot to help delineate the planing and non-planing regimes – the outer ring of data corresponds to planing while the inner cluster corresponds to normal (displacement) sailing. The linear regression based controller performs particularly well across the range of wind angles; KNN and to a lesser degree random forests both struggle with flip flopping between regimes.

Further Considerations:

Although this investigation did result in successfully leveraging machine learning algorithms to develop sail controllers, several consideration remain unaddressed. These algorithms were developed for offline use; controllers will not adapt to new physics paradigms in real time. This means that comprehensive training data must be obtained before a controller can be learned, which might be inconvenient. The linear regression and KNN algorithms discussed here are not clearly extended to higher dimensional problems. KNN is notorious for failing in high dimensions, and basis expansion grows polynomially in the input space and exponentially in the degree of expansion. Finally, running the controllers is somewhat costly. Linear regression and random forests both require solving a constrained optimization problem every time control values must be produced, while KNN itself scales poorly with the size of the input data. Because the controllers are intended to be developed offline, however, this problem can be circumvented by precomputing control outputs across the entire input space at some resolution and implementing the actual controller as a lookup table.
Conclusion:

This report details a proof of concept for a data driven approach to sail controller development. We consider a simplified control problem with just two input variables and two control variables and are able to produce reasonably good control solutions using learning techniques including linear regression with basis expansion, random forests, and KNN. Linear regression in particular is demonstrated to be robust to significant change in the underlying physical model; this is important because the true physical model of sailing is not easily reproduced, and response functions vary across types of sailboat. This specific control problem has direct applications to autonomous sailing including Yale Undergraduate Intelligent Vehicles' sailboat Ratchet, but it can also be viewed as a stepping stone toward higher dimensional control solutions and analytics that are increasingly relied upon in the world of high performance sailing.