The Problem:

In a cycling individual time trial (ITT), pacing plays a large role in the efficiency and speed of a cyclist throughout the race. The optimal pacing strategy, however, is very hard to find, as it can vary greatly based on the course, putting riders with limited experience or coaching resources at a disadvantage to others.

My Solution:

To try and generate an optimized pacing strategy for a given course, I am building models for rider fatigue, using a formula that estimates the W' balance, which acts as a sort of "battery" for a rider at a given time, and the speed at which a rider travels at an inputted power output and road gradation. These models, as well as a segmented version of an ITT course, which will vary between training generations, are then used to train an AI agent to complete the course as fast as possible. This is achieved through the use of Neural Evolution of Augmenting Topologies, or NEAT, in which a neural network is evolved using a set of nodes and connections, and is granted a fitness score to evaluate and reproduce the networks. This AI should, once trained, be able to take in information about the course and rider, and output a pacing strategy. Because rider fatigue may vary based on environmental and other factors, such as heat or how a rider feels, the pacing strategy generated may be too easy or too hard when implemented. I plan to combat this by comparing a rider's expected heart rate over the last few minutes to their actual heart rate. Then, while the rider rides, if their actual heart rate is considerably above or below this expected heart rate, the remainder of the pacing strategy can be scaled accordingly.

Progress:

Since submitting my proposal, I have built the models for fatigue and speed, as well as coded a function to read a GPX file and turn it into a set of segments, each with a value for segment length and grade, which the AI can read. I have also begun to implement the NEAT-py library, which implements a version of NEAT in python. So far, I have trained a simplified version of the final product, which only used one course.

Next Steps:

Next, I plan to run the model on a variety of courses to build a model that can handle any course. Once I have a neural network, I will make a GUI of some sorts, likely a website using HTML, so that

users can input a course and receive a pacing plan. I will then make it so that this website can receive data from an external heart rate monitor, to adjust the pacing during a ride as outlined above. At some point during this process, once easily usable, I will also compare the results of a rider in an ITT when self-paced and when using the pacing plan generated for them.

Expected Results:

I expect to end up with a model which, hopefully, looks at a course and paces such that it outputs higher power on uphills, then recovers on descents. It should also hopefully save energy when a large hill is coming up, and expend more when approaching a significant descent. The real-time adjustments should make it so that these pacing plans are achievable, but near the limit for a human rider. This should all hopefully culminate in pacing that outperforms self-paced efforts in testing.

Citations:

https://gssns.io/posts/w-prime-balance-algorithms/
https://www.gribble.org/cycling/power_v_speed.html
https://markliversedge.blogspot.com/2014/10/wbal-optimisation-by-mathematician.html
https://chatgpt.com/g/g-p-68fbe55a7c5481918e238407350512b0-supercomputing-25-26/project
https://neat-python.readthedocs.io/en/latest/index.html