Fitting Models to Data

M. M. Taylor

Martin Taylor Consulting

mmt@mmtaylor.net

Problem

Given data from many tracking runs from each of many subjects, run under several conditions …

What is the best model structure?

What are the best-fitting parameters of the best model?

How are these parameters affected by experimental variables?

Experiment

One of several different experiments repeated on a four-hour cycle using subjects who worked continuously for 28 hours.

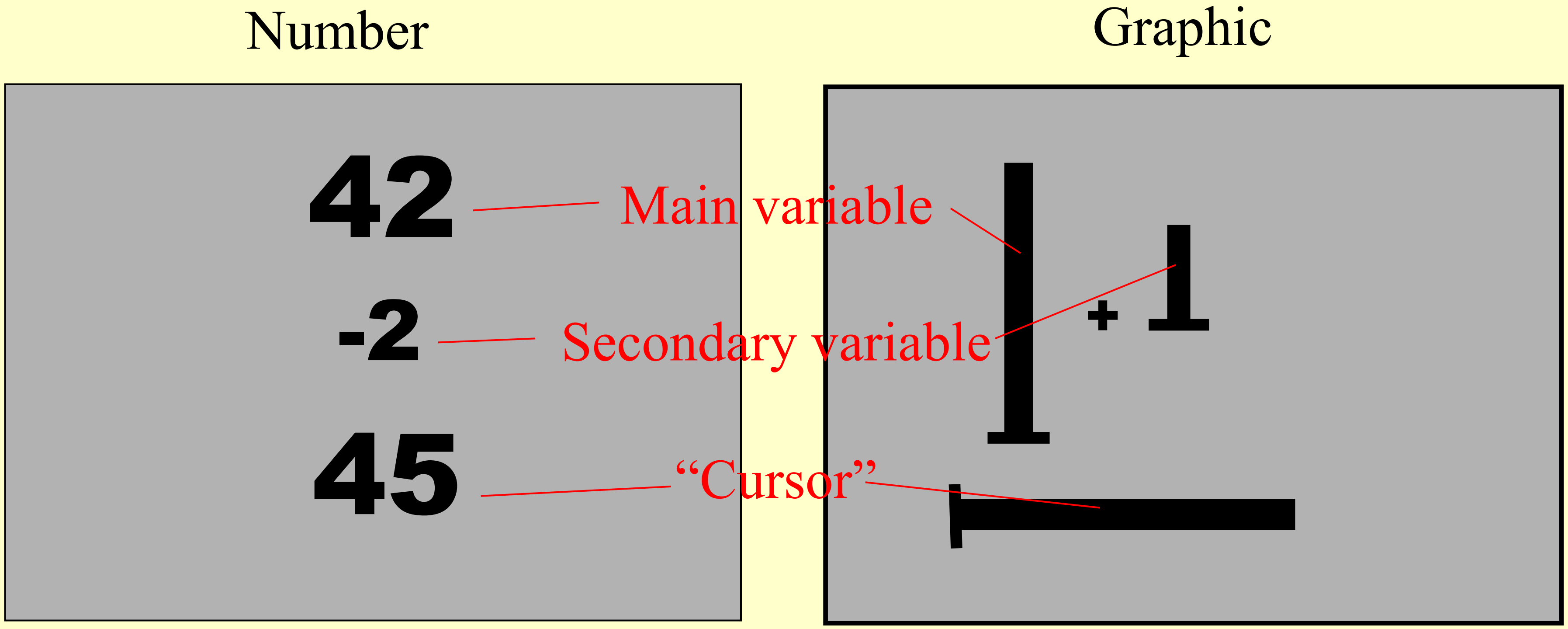
The experiment was intended to compare the effects of sleep loss on tracking different kinds of perception at different levels of complexity. Each of 32 subjects did 42 50-second tracking runs, 6 per experimental cycle — 7 tracking runs on each of three complexity levels, with two different kinds of perception: the value of a (usually) two-digit number and the length of a line — giving a total of 1344 tracks to be fitted.

(Actually, one subject got sick near the end of the period, losing 18 tracks and rendering several more of dubious value.)

A previous experiment had suggested the possibility that more complex tasks were less affected by 60+ hours of sleep loss than were simpler tasks. Most of the effect happened in the second sleepless night. This time the subjects had only one sleepless night, but we thought it worth trying to see whether complexity had a detectable effect.

Today I intend to talk only about the issues involved in fitting models, not about experimental results, which are not complete.

Tracking Displays



In the "Easy" condition, the secondary variable was omitted. In the "Medium" condition, it was fixed for the duration of a track, sometimes +, sometimes -. In the "Hard" condition, it varied, but more slowly than the main variable.

Tracks and Models

This track was done about 10 am after a sleepless working night.

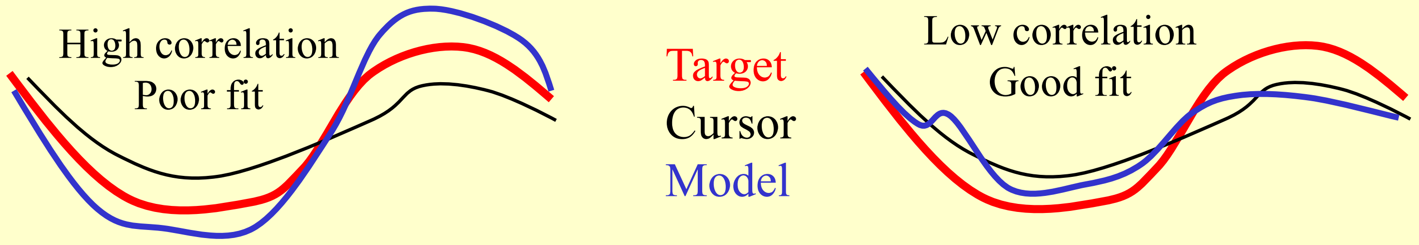
What constitutes a well-fitted model?

•Should the model attempt to fit the areas around sample 950 and 1150-1300 where the subject's cursor movements look unrelated to the target variations?

•Should it attempt to fit the subject's overshoots that are evident over the first three cycles of target variation?

Fitting criteria

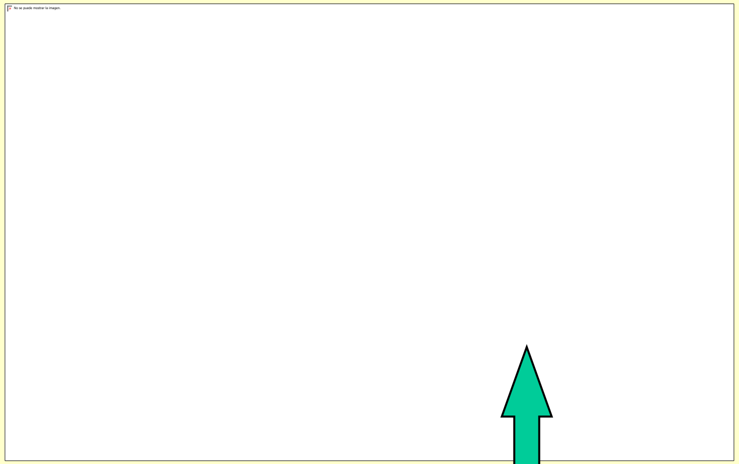
Problem: Any model that actually controls will produce a track that correlates highly with the track produced by a subject who also actually controls.

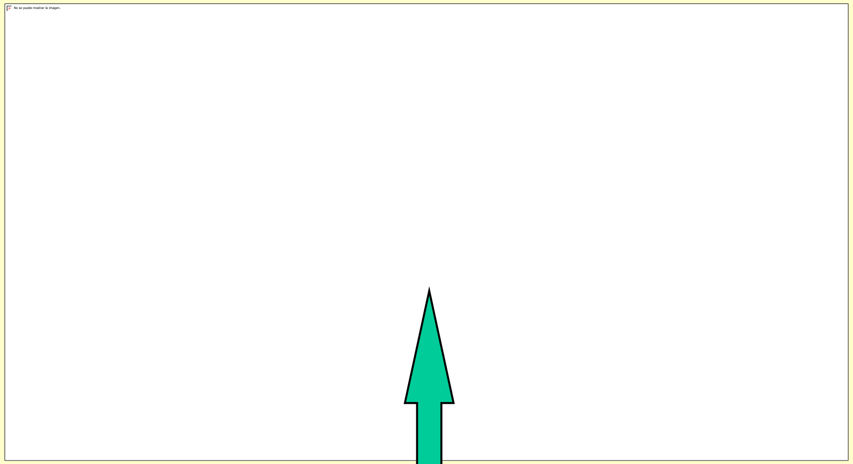


Properties of a good model fit:

1. The model's track should follow close to the subject's track (the cursor).
2. The model's track, where it deviates from the target, should deviate in the same sense as the cursor.

Fitting criteria (continued)





More properties of a good model fit:

1. Regions of the track where the subject appears not to be tracking normally should carry little weight.
2. The model should not attempt to account for variations that are much faster than the variations in the target, so the fitting criterion should not involve those variations. (One can't filter the raw data because doing so ordinarily causes phase shifts that would affect the model parameters).

Fitting criteria (continued)

Several criteria could be used, such as RMS track deviation, cross-correlation, and the like. These are linear, and don't satisfy the conditions.

Criteria used:

•Proportion of samples for which the model is closer to the cursor than to the target.
But it's no good to have the model closer to the cursor if it's far from both, so …
•Weighted track length: the closer one track is to another the greater the weight assigned to that sample, up to 1.0 for identity. The sum is the weighted track length.
We want the model-to-target weighted track length to be high, but the model-to-cursor track length should be higher, which leads to …
•Sample proportion scaled by the weighted track length of model-to-cursor.

•Five fitness parameters were computed, including cursor-to-target weighted track length, which indicates how well the subject tracked, overall. The four that involve the model were combined to make a scalar fitness value to be minimized by the optimization procedures.

What Model?

For any tracking task, there are many models that would control adequately. We need to discover which of these most closely represents what a human does.

Plausible models may be devised from intuition, previous experience, careful examination of typical classes of human control errors, …

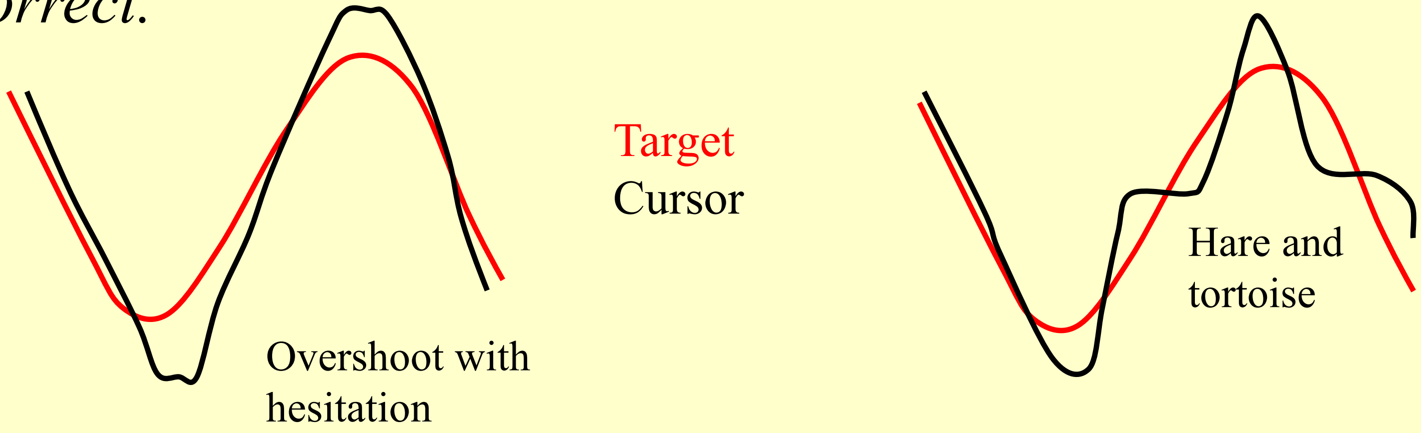
Models are defined by their structure, but their performance is determined by the values assigned to their operational parameters (e.g. output gain, transport lag, threshold, etc.).

Models can be compared fairly only when each is tested against the data with its own optimum set of parameter values.

I tried a couple of dozen different model structures before settling on two that seemed good enough to be worth trying on the whole set of 1300+ trials.

Common track Features

Problem: There are some tracking characteristics that occur often enough that a good model should at least be able to display them, and preferably fit them when the parameters are correct.

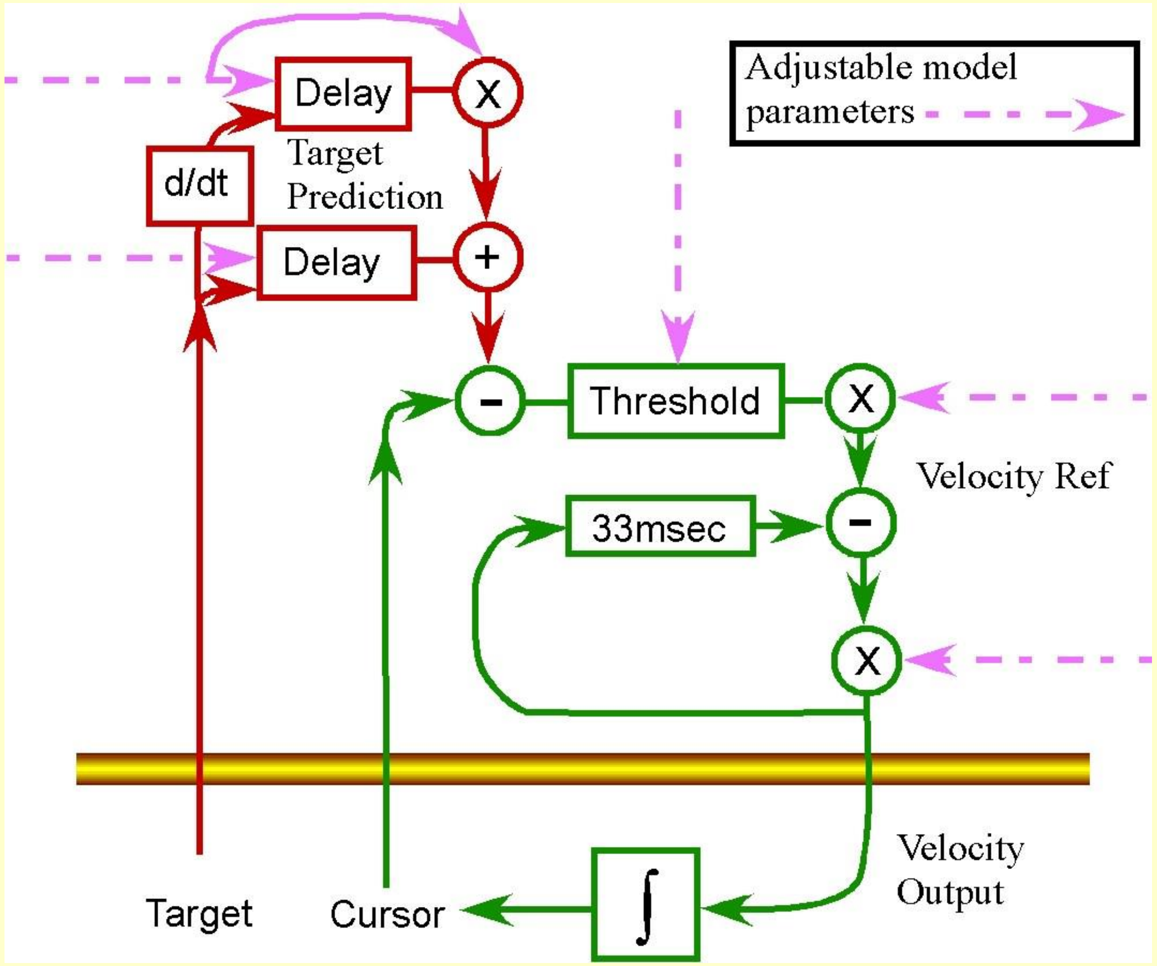


Two problematic characteristic tracking errors that should be modelled:

1. Overshoot with hesitation. The subject keeps going when the target turns around, and then hesitates before chasing the target in the new direction.
2. Hare and Tortoise. The subject chases too fast and overshoots, waits for the target to catch up, and then zooms after it again.

One 5-Parameter Model

This is a model that provides good fits to the data



Assumptions:

It takes time to perceive the target value and rate of change, but the subject uses the rate of change to predict the current value of the target.

The subject's output affects the cursor velocity, which the subject perceives kinaesthetically, with minimal delay (33 sec is one sample period).

The subject will ignore any error smaller than some threshold.

Another 5-Parameter Model

This model also provides good fits to the data

No se puede mostrar la imagen.

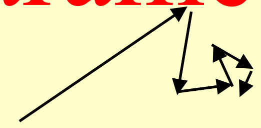
Assumptions:

It takes time to perceive the target value and rate of change, but not the cursor position and rate of change.

The subject will ignore any position error smaller than some threshold.

Subject controls the difference between the cursor velocity and the target velocity (the “chase rate”) using reference provided by the position error.

Finding optimum parameter values



Initial approach: modified “E-coli”

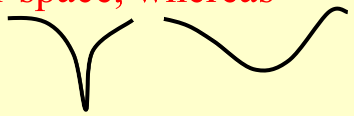
Two different models, each with 5 parameters (Position Gain, Velocity Gain, Position Lag, Velocity Lag, Position Threshold). The “e-coli” process was modified by analogy with “simulated annealing” to reduce the problem of getting stuck in “canyons” in the fitness space. This considerably improved the fit of the models tested.

Limits: one optimization limited to 30 minutes of CPU time (to ensure that a complete 42-track optimization set took less than 24 hours). This time was partitioned among 10 independent e-coli runs, and the best fit among the 10 was assumed to represent the best for that model.

Observations:

•Across the 10 e-coli runs the values of some pairs of “best-fit” parameters were highly correlated — one could be traded off against another, for example one might get as good a fit with low gain and small lag as with high gain and longer lag.

•The best run for Model “A” usually gave a better fit than the best run for Model “B”, whereas most e-coli runs for Model “B” were better than most runs for Model “A”. The implication seemed to be that Model “A” had a narrow, deep optimum in the parameter space, whereas Model “B” had a broad range of reasonably acceptable parameter values.



MTC

Finding optimum parameter values

Second approach: multidimensional parabolic fit

On the assumption that the space of parameter fitting is reasonably smooth near the optimum, most such spaces have hyper-ellipsoidal surfaces of equal fitness, and cross sections along straight lines through or near the optimum are parabolic.

Observations:

•The fitting values gave rather worse results than the e-coli results for the same models and data, and the variations from one optimizing run to another for the same track were more dramatic.

•Plotting the fitness values along each cross-section suggested that the fitness space was not smooth, but was often quite irregular, violating the conditions for the parabolic fits.

•The fitness values along many cross-sections were not only irregular, they were often non-monotonic, meaning that even the e-coli process could easily get caught in a sufficiently deep local minimum, despite the simulated annealing modification.

Finding optimum parameter values

Third approach: Genetic Algorithm

The GA approach is quite different from the first two approaches, since there is no assumption about the form of the surface other than that the fitness doesn't change too abruptly at too many points in the space.

The GA Process:

•An "actor" (a model with specific parameter values) is defined by a set of "genes" and an algorithm for producing the model structure and its parameter values (its phenotype) from the values of the genes (its genotype). In this case, the model structure is predefined, and the genes affect only the parameter values.

•Each actor is a model that creates a specific track when confronted with a target track. It therefore has a unique fitness value for that track, and a different fitness value for other tracks.

•To create an initial population of actors for a specific track, random values are assigned to the genes of a large number of actors. Those that have sufficient fitness for the track in question are allowed to survive, until a large enough population has been produced.

•At this point, actors are scattered randomly over the parameter space. All of the actors control tolerably well, and their tracks are not too different from the track to be simulated.

Finding optimum parameter values

Third approach: Genetic Algorithm (continued)

Genes and Phenotype

•Five parameters define the phenotype of an actor, but in the e-coli studies I found that these parameters were often highly correlated — variation in one could be traded for variation in another.

•In an attempt to avoid this problem in the GA process, I used five "basic genes" and five five-dimensional rotations. The rotations were expected to evolve so as to reduce correlations among the effects of the basic genes. The five rotated results were scaled nonlinearly. To produce the five parameters took 35 genes, all of which might contribute to the value of any one parameter.

Finding optimum parameter values

Third approach: Genetic Algorithm (continued)

The GA Process:

•To create a new population from the old, actors are allowed to “mate” to create “children” whose genes are derived from those of the parents. To reduce the effect of inbreeding, newly created actors are added to the population at every generation.

•From the larger population (parents, children, and newcomers), those with the greatest fitness are allowed to survive — the same number in each generation.

•After sufficient generations, the parameter values (the “phenotype”) of the fittest actor are taken to be the optimum model parameters. The values of the genes are not recorded.

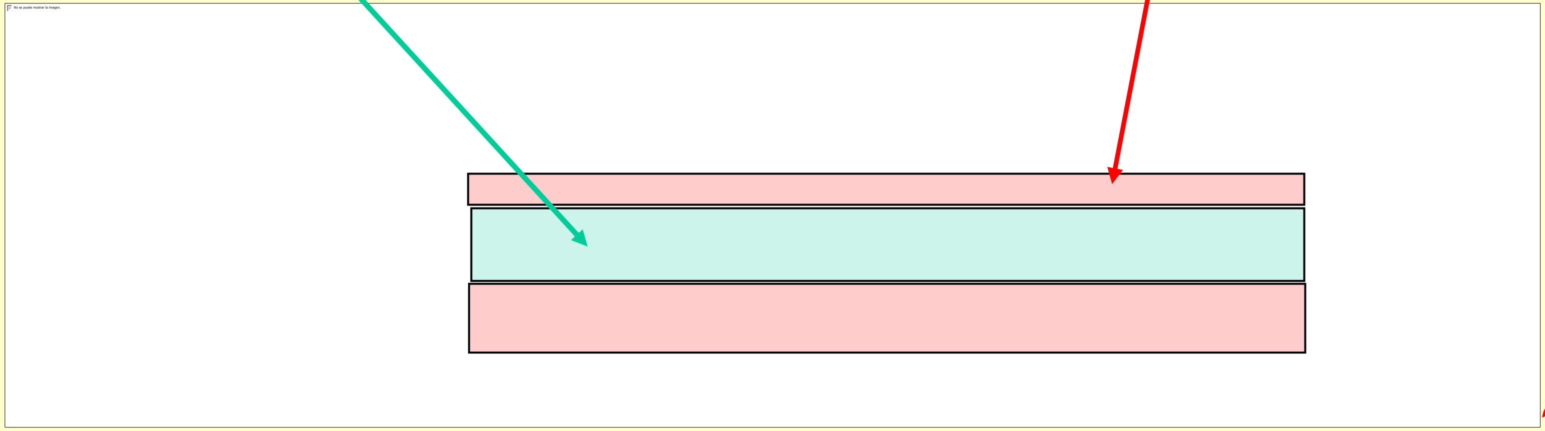
Finding optimum parameter values

Third approach: Genetic Algorithm (continued)

•The genes are real numbers between zero and unity. The mating process for a gene generates a gene whose value is a new real number that is a proportion p of one parental gene and 1-p of the other, where -0.5< p <1.5. A small mutation (a real number) may, with low probability, be added.

•In mating complete actors, two values of the proportion are used, p and q. We call each column of the table below a “chromosome”. In the mating, part of the chromosome uses proportion p, the other part uses proportion q.

•All the chromosomes are split in exactly the same way, so that the rows in the table always stay together for the selection of whether to use p or q, but mutations are applied independently to individual genes of the children.



MTC

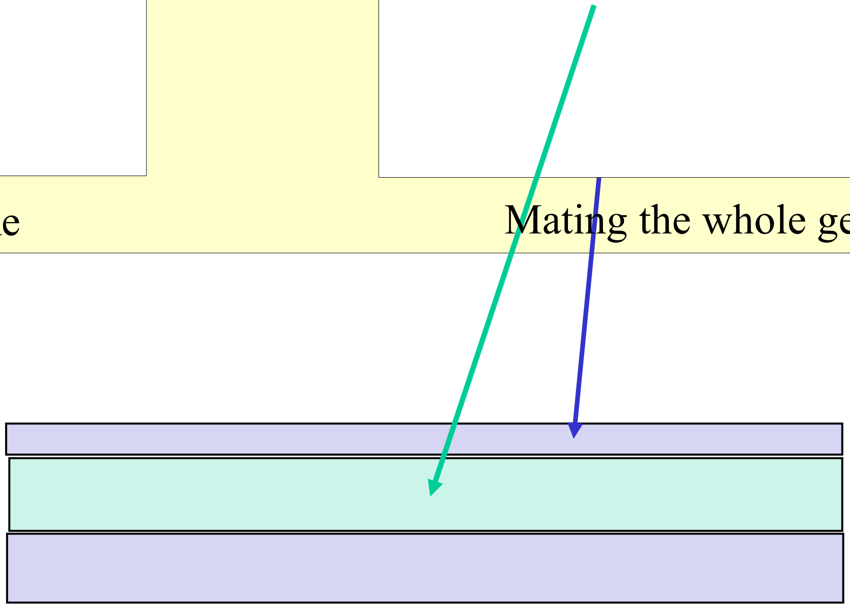
Finding optimum parameter values

Third approach: Genetic Algorithm (continued)

•All the chromosomes are split in exactly the same way, so that the rows in the table always stay together for the selection of whether to use p or q.

Mating one chromosome

Mating the whole genome



Control Systems Group, Crieff Hills, 2005

MTC

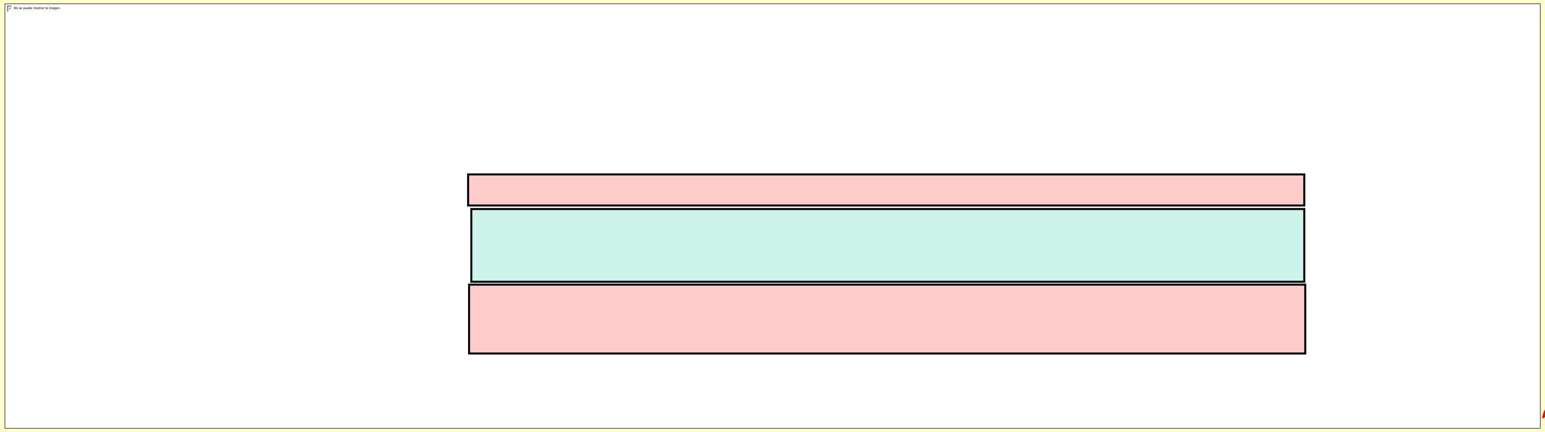
Finding optimum parameter values

Third approach: Genetic Algorithm (continued)

•A population size of 100 is maintained through many generations.

•An initial population is created by choosing all the genes randomly and keeping those “actors” whose fitness is better than a threshold value.

•The next generation is chosen by allowing 200 matings and generating 20 new sufficiently fit actors (temporarily creating a population of 320). The parents for each mating are chosen randomly, with higher probability for the fitter members of the parental generation. The fittest 100 of the 320 form the next generation.



Finding optimum parameter values

Conclusions

Comparison with e-coli

I ran all 42 tracks by one subject using many variants of the GA approach, using the same model and the same limit on computing time as I used for e-coli.

All of the GA variants found a better optimum for all or almost all the trials than did the e-coli method using the same amount of computation. The improvement was often dramatic.

The GA approach seemed to work much better using 35 genes to generate 5 parameters than it did using 5 genes, one per parameter. Allowing the evolution of the rotation matrix to decorrelate the effects of the individual genes may be responsible. I did not try the e-coli method in such a high-dimensional space, but prior analyses have suggested that its effectiveness is sharply reduced as the dimensionality increases.

Finding optimum parameter values

Other conclusions

The fitness criterion I used was evolved by trial and error, starting from the simple use of model-cursor correlation. Particular examples showed inadequacies of successively modified fitness criteria, and there is a certain ad-hoc quality about the criterion used in these experiments.

Although usually a "by eye" comparison of two fits agrees with the criterion as to which is better, this is not always true. In particular, there are cases in which a parameter set leading to a non-controlling static model is found to be a better fit than any sets of parameters for which the model actually controls.

If a model is non-trivial, a Genetic Algorithm approach seems well suited to finding the best parameter values. It might be worthwhile to use it to find the best model structure, but this would require appreciable software development.

Furthermore …

Reorganization

This experiment is only concerned with finding an optimum fitness for a set of parameters in a relatively low-dimensional space. It found that using a genetic algorithm in a much higher-dimensional space was a more efficient way to discover good parameter sets than was the "e-coli" method first tried.

The GA approach is modelled on biological evolution (very much simplified). It seems reasonable to try it further, to optimize not only the model parameters, but also the model structure, within design constraints appropriate to the experimental situation.

Speculating further, it also seems not unreasonable to think that the GA approach might lend itself to implementation of reorganization within a structure of elementary control systems such as the canonical HPCT hierarchy. To do this would require that the structure be multiply redundant, using several copies of control systems at each level rather than relying on the optimal performance of a single system for any particular perceptual or output function. I have not thought about how such a notion might be implemented, and it might not be practical, but it might be worth some consideration.

Fitting Models to Data

M. M. Taylor

Martin Taylor Consulting

mmt@mmtaylor.net