Chapter 8

Driving Simulation

The goal of this experiment is to distinguish different people's driving styles. The data was collected from five people using a simulator. The simulator, shown in Figure 8-1, was designed by M.C.Nechyba.

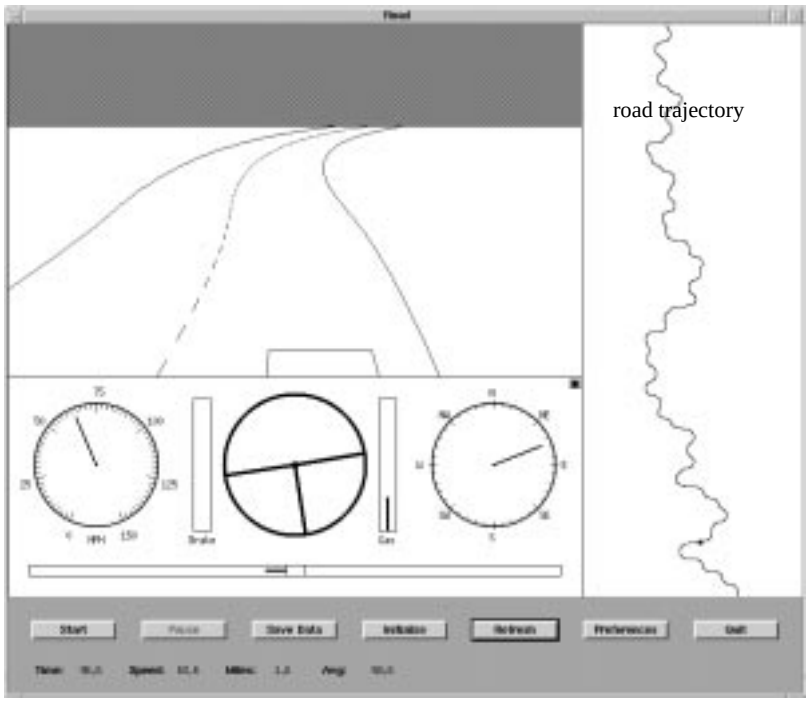

Figure 8-1: Driving simulator interface. (Courtesy M.C.Nechyba)

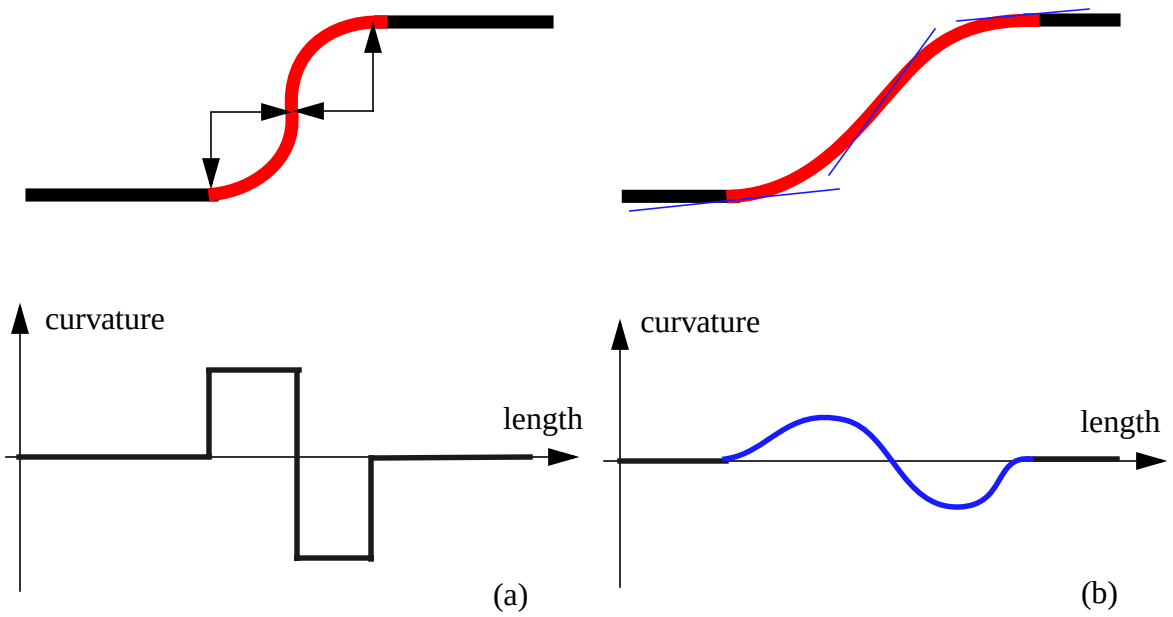

Figure 8-2: The simulator's road trajectory is generated in a way illustrated by (a), in which the curvature of the road changes abruptly. However, a high way in the real world is actually designed in the style of (b), in which the curvature changes smoothly.

8.1 Experimental data

The human operator has the full control over steering (horizontal mouse position), the brake (left mouse button) and the accelerator (right mouse button). Although the dynamics of the simulator strictly follows the form of some real vehicles [Nechyba et al, 98, (a) and (b)], the human drivers' behavior is quite different from the real one on the real roads. One reason is that the road trajectory of the simulator is generated as a sequence of straight-line segments and circular arcs, which differs from the real roads in the real world, illustrated by Figure 8-2.

We generated three road trajectories, each of them is around 20km. Five people were invited to operate on these three different roads after they had warmed up. The simulator took the record of the state of the vehicle and the environmental variables (described in details later) five times per second, while the simulator itself runs 50 Hz. Thus, we collected fifteen datasets, O_{ij}, i =

1, 2, ..., 5, j = 1, 2, 3, i represents the operators, and *j* corresponds to the different road trajectories.

The state and environmental variables are listed in the following table:

Table 8-1: State of vehicle and the environmental variables

	Description	*Time Delay (0.42 Seconds)*
v_ξ	The lateral velocity	6
v_η	The longitudinal velocity	6
ω	The angular velocity	6
(x, y)	The car-body-relative coordinates of the road median	10
δ	The user-applied steering angle	6
α	The user-applied longitudinal force on the front tires	6

If a human driver is viewed as a system, the input consists of the following information: (1) the current and recent vehicle states, $\{v_\xi(t-n_\xi), ..., v_\xi(t-1), v_\xi(t)\}$, $\{v_\eta(t-n_\eta), ..., v_\eta(t-1), v_\eta(t)\}$, $\{\omega(t-n_\omega), ..., \omega(t-1), \omega(t)\}$, where n_ξ, n_η, n_ω are the time delays. (2) previous control actions, $\{\alpha(t-n_\alpha), ..., \alpha(t-1), \alpha(t)\}$, $\{\delta(t-n_\delta), ..., \delta(t-1), \delta(t)\}$. (3) The visible view of the road ahead, $\{x(t+1), y(t+1), ..., x(t+n_r), y(t+n_r)\}$. The outputs should be $\delta(t+1)$ and $\alpha(t+1)$.

Notice that even for the same human driver, very similar inputs may lead to radically different outputs $\delta(t+1)$ and $\alpha(t+1)$, referring to [Nechyba, 98 (b)].

The time delays of the inputs (including n_r of the road median ahead) were decided based on our empirical experiments. Because of the time delays, the input dimensionality of a dynamic system tends to be very high, in this case, it is 50. The high dimensionality may have strong negative impact on the efficiency of both the information retrieval from memory and the clas-

sification process afterwards. For kernel regression, the computational cost is *O(Nd)*, where *N* is the memory size and *d* is the input space dimensionality. Even though we used kd-trees to re-organize the memory in order to speed up the information retrieval process, kd-tree performance is not satisfactory when the input dimensionality is too high.

Principal Component Analysis (PCA) [Jolliffe, 86] can be used to compress the input space if some of the inputs are linearly correlated. Notice that, theoretically there is no guarantee that PCA can shrink the dimensionality of the dataset in all cases especially when the input attributes are not linearly correlated; however in practice, PCA is a very popular method. In the simulation driving experiment, we used PCA to compress the input space from 50 dimensions to 3 dimensions, with only 7.2% loss of information.

8.2 Experimental results

As mentioned above, we collected fifteen datasets from five people driving on three road trajectories. We assigned one dataset to be a testing dataset; say, O_{21}, which is actually the dataset generated by the second driver along the first road. We did not tell OMEGA who was the real driver, and asked OMEGA to figure it out. To do so, OMEGA needed some *labeled* training datasets. In our experiments, we let those datasets collected from the other roads be the training datasets, i.e. O_{ik}, $i = 1, ..., 5, k = 2, 3$. By "labeled" we mean for each training dataset, OMEGA knew exactly who was the operator.

Using the OMEGA technique described in Chapter 2, we calculated the average of the negative log likelihood of each testing dataset with respect to all five human operators. Hence, for each testing dataset, we got five likelihood curves corresponding to the five possible drivers. OMEGA detected the hidden driver according to the tails of the likelihood curves: the lowest one indicates the most likely operator.

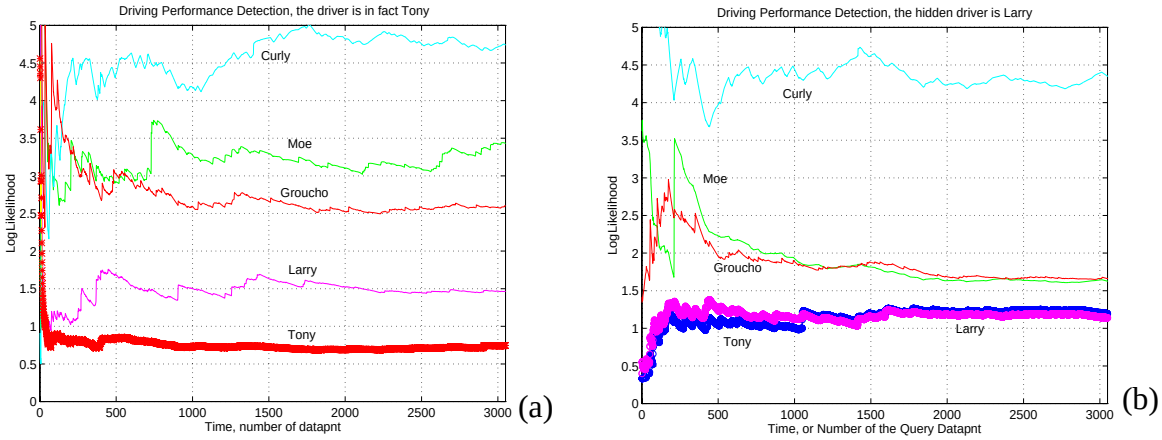

Figure 8-3: Simulation driving style OMEGA detection. (a) A correct case. (b) A sample of the confused cases. There are two confused cases out of the fifteen experiments, all others are correct.

There are in total fifteen testing datasets, OMEGA succeeded in detecting the hidden drivers correctly thirteen times. A typical correct case is demonstrated in Figure 8-3(a), which shows how OMEGA detected the underlying operator of a testing dataset, O_{11}. The horizontal axis is the number of data points in the testing dataset OMEGA has processed. The vertical axis is the average of the negative log likelihood. Tony's negative log likelihood curve is closest to the horizon, and it is remote from all other drivers' curves. Hence, Tony is the most likely operator of the testing dataset, O_{11}. At the early stage when only a few testing data points have been processed, the curves are not stable, but afterwards they become smoother and more stable.

Although OMEGA did not make any mistakes in the fifteen experiments, it was confused in two cases[1]. One of them is shown in Figure 8-3(b), in which the lowest curve does correspond the real driver, Larry; however, Tony's curve is too close to Larry's, so that OMEGA can hardly tell who is more likely to be the hidden driver between Larry and Tony.

1. To distinguish the confusing cases, we assign the significance level α to be 5%, referring to Chapter 2.

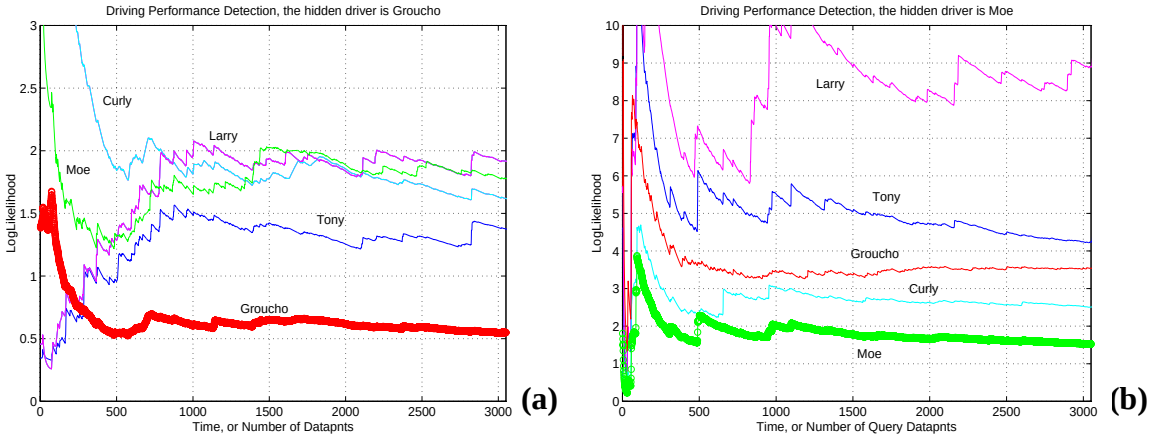

Figure 8-4: When some data points in the testing dataset are not consistent with a certain training dataset, the corresponding likelihood curve may look bumpy. If the data points are so unusual that there is no similar scenario in all the training datasets, then all the curves are bumpy, and roughly paralleling to each other, referring to (b).

As an on-line detection tool, OMEGA is capable of starting its job with very few data points. As expected, the precision is very bad. Thus, the likelihood curves look chaotic at first. But with more and more data come, the curves converge to be stable.

Sometimes the likelihood curves are bumpy, because the driver did something unusual compared with his behavior in the training datasets. After studying the datasets carefully, we notice that the abnormal behavior usually occurs when the curvatures of the road change rapidly, referring to Figure 8-2(a). If the human operator does not pay sufficient attention, he may drive off the road when the abrupt change of the curvature happens. Therefore, a careful driver's curve is smoother and more stable than others, illustrated by Figure 8-4(a). However, sometime the curvature changes so much and so suddenly that no one was able to keep his operation in a consistent manner. In those cases, all the curves are bumpy and roughly parallel to each other, referring to Figure 8-4(b).

Another interesting observation is that some people's curves tend to be close to each other, for example, Moe's and Groucho's. The short distances between their curves implies that their

driving behaviors are close to each other in the experiments. But does it give any hint to the similarity of their personalities? This is an open question, but it is interesting to observe that Moe and Groucho do spend a lot of time together during weekends.

8.3 Comparison with other methods

Although OMEGA works well for detecting the hidden drivers in these simulation experiments, some legitimate questions are still opened, such as: is there any simpler method which can work as well or better?

8.3.1 Bayes classifier

Bayes classifier is a simple method which compares the features. Referring to Table 8-1, the state of the vehicle and the driver's action are the instantaneous velocity (including v_ξ and v_η), angular velocity ω, user applied steering angle δ and acceleration or brake force α. We treated the vehicle's state variables, the environmental variables, in conjunction with the control actions as the feature and applied Bayes classifier, with tuned-up parameters, to distinguish the five human operators. The result is shown in the first row of Table 8-2 :

Table 8-2: Comparison of OMEGA with other alternatives

	Correct	*Wrong*	*Confused*
Bayes classifier	6	6	10
HMM	13	0	2
Global linear	12	1	2
OMEGA	13	0	2

Obviously, feature-based Bayes classifier did not perform well. The reason are that: (1) features-based approach does not consider the mapping between the inputs and the outputs. (2)

some features are also influenced by the road conditions, besides the different human driving styles. (3) different human operators' feature values have a large overlapping region, exhibited in Table 8-3.

Table 8-3: Aggregate features of human simulation data (based on Nechyba's data)

	Velocity (v)	*Angular velocity* (ω)	*Steering angle* (δ)	*Longitudinal force* (α)
Tony	67.2 (12.6)	(0.205)	(0.097)	2.03 (3.86)
Larry	72.2 (7.8)	(0.193)	(0.072)	1.85 (2.37)
Moe	70.5 (7.9)	(0.198)	(0.074)	1.91 (3.25)
Curly	63.3 (10.1)	(0.175)	(0.056)	1.33 (1.88)
Groucho	73.2 (9.3)	(0.259)	(0.100)	2.33 (2.68)

The numbers in parentheses are the standard deviations. Since the mean values of angular velocities and steering angles depend on the specific road trajectories, only their standard deviations are listed in the table.

8.3.2 Hidden Markov Model

With rich mathematical fundamentals, the Hidden Markov Model [Rabiner, 89] is very useful in speech recognition. When we hear the sentence "I love you", in fact, our perception system recognizes the states [ai] [la] [v] [ju:] in sequence. The order is also important. However, due to the difference in emphasis, skipping, and pausing, the transitions among the states are not deterministic. Some states may last longer, others may be skipped. For the same example, it can be expressed in a different way: [ai] [pause] [la] [la] [v] [ju:], or "I, lo-ve you", which sounds more romantic than the plain tone. Therefore, HMM assumes the transitions among the different states are probabilistic instead of deterministic. To recognize a piece of speech, HMM relies on the approximation of those state transition probabilities.

Due to accents and/or personal styles, few people can precisely pronounce every word. Thus, the states (i.e. [ai], [la], [v], and [ju:]) are hidden underneath the stream of the sound signals. The mapping between the sound signals and the hidden states is not so simple as one-to-one; instead their relationship is also probabilistic. HMM is capable of approximating the probabilistic mapping between the sound signal and the states, as well as the transition probabilities.

Although HMM is very successful for speech recognition, one should be careful before using HMM as a general purpose time series recognizer. The reason is that HMM assumes the state transition probabilities are the most fundamental characteristic of a time series. And usually, the transition probabilities are assumed to be time-invariant.

[Nechyba, 98 (a)] applied HMM to distinguish different simulation driving styles. He did not separate the inputs and outputs, instead, he treated the states of the vehicle and the environmental variables equally as parts of observations. He assumed that the observations were stochastically decided by some hidden states. Although the physical meanings of those states were not clear, he conjectured that their transitions probabilities differed with different drivers. Therefore, given a unlabeled driving time series, Nechyba approximated a HMM which fit the time series well. Then he compared the new HMM with those in memory whose underlying drivers were known. Usually one HMM in memory is closer to the new one than the others are. The closest HMM in memory indicated the driver who is most likely to be generator of the unlabeled driving time series.

As Table 8-2 shows, the experimental performance of HMM is as good as that of OMEGA.

Why does HMM approach work in this domain? In our point of view, a hidden state is an abstract scenario of the state of the vehicle in conjunction with the environmental situation, and the human driver's control action. Facing a certain scenario, different drivers may give dissimilar control responses which lead to different new scenarios at the next time step. Thus, different

drivers' diverse responses make the transition probabilities of his HMM distinguishable from those of others.

Therefore, we think the fundamental methodology of [Nechyba, 98(a)] is similar to that of OMEGA. There is no surprise that the accuracies of HMM and OMEGA are close to each other. While Table 8-2 gives a top-level comparison, Table 8-4 and Table 8-5 view the precision in depth. Each number in the tables is a probability of a testing dataset being generated by a certain operator. Each row corresponds to a specific testing data set, and the real operator is in the leftmost column. The other columns represent the five candidate drivers. The number in the (2,3)'th cell is the probability that a testing dataset, which was secretly generated by Larry, would be detected as the performance of Moe. Thus, the sum of the five probability values in each row is always 1.0. The number on the shaded diagonal is expected to be bigger than the others. And the bigger the diagonal number is, the better the detection system performs. Otherwise, the detection fails.

Comparing Table 8-4 and Table 8-5, we claim that HMM and OMEGA have similar accuracy in this simulation domain. No one is significant better than the other.

Table 8-4: Cross validation of OMEGA

	Tony	*Larry*	*Moe*	*Curly*	*Groucho*
Tony	0.677	0.139	0.020	0.031	0.133
Larry	0.243	0.441	0.014	0.129	0.173
Moe	0.037	0.001	0.836	0.114	0.012
Curly	0.060	0.030	0.272	0.570	0.068
Groucho	0.130	0.070	0.199	0.156	0.445

However, OMEGA outperforms HMM in other aspects, such as efficiency, data consumption, flexibility, robustness, etc., referring to Chapter 2.

Table 8-5: Cross validation of HMM (based on Nechyba's data)

	Tony	*Larry*	*Moe*	*Curly*	*Groucho*
Tony	0.425	0.157	0.217	0.154	0.047
Larry	0.202	0.538	0.116	0.101	0.043
Moe	0.212	0.077	0.429	0.172	0.110
Curly	0.154	0.073	0.180	0.413	0.180
Groucho	0.066	0.040	0.163	0.237	0.494

8.3.3 Global linear model

OMEGA is a non-parametric method, which means it does not need any assumption about the function relationship between the input and output. However, if we do know the function form, we have more options to detect the system. For example, linear system is simple and very popular in practice, which assumes the output is a linear function of the inputs. To detect a linear system, we can either follow the residual approach or compare the parameters of the linear functions.

- **Residual approach:** For each training dataset, we approximate the parameters of the linear function between the inputs and the outputs. Then, given a unlabeled testing dataset, we temporarily suppose it was generated by the first system. Through the first system's linear function, we predict the outputs corresponding to the inputs of the testing data points. There usually exist some residuals between the predicted outputs and the real outputs in the testing dataset. The smaller the residuals, the more likely the first system is the underlying system of the testing datasets. We enumerate all the candidate systems, the one with the smallest residuals is most likely to be the underlying system.

- **Parameter approach:** We can approximate the linear function's parameters of the testing dataset, as well as those of each training dataset. By comparing the parameters of the test-

ing dataset with those of each training dataset, one by one, we can tell which training dataset is most similar to the testing dataset, hence, we detect the underlying operator of the unlabeled testing dataset.

It is interesting to find that the simulation driving domain *happens* to be linear. Referring to Table 8-2, the global linear approach performed satisfactorily compared with OMEGA and HMM. It did the correct detection job in most cases.

In our previous work [Deng et al, 97], we compared the driving behaviors of an identical human operator, but under two conditions: sober and intoxicated. We found that ARMA(4,4)[2] was a good model for the behaviors under both conditions. We approximated the ARMA parameters of the datasets under different sobriety conditions, and found the parameters of the intoxicated driving behavior deviated from the sober ones, shown in Figure 8-5. The drunken parameters were more widely scattered due to the fact that the human operator experienced the varying levels of intoxication.

8.4 Summary

In this chapter, we applied OMEGA to detect the driving style using simulation datasets. This domain is more complicated than the tennis one because driving is dynamic with feedback, and there are a large number of variables effecting the driver's control action. Hence, the pre-processing of the datasets is important. We used PCA technique to compress the input space.

OMEGA does very job in this domain, but is not significantly better than the other methods. However, OMEGA has other good properties: it is simple, it is easy to update the memory, it is

2. Auto Regression Moving Average (ARMA(p,q)) model [Brockwell et al, 91] is a popular linear time series model. (p,q) refers to the window sizes of its AR part and MA part.

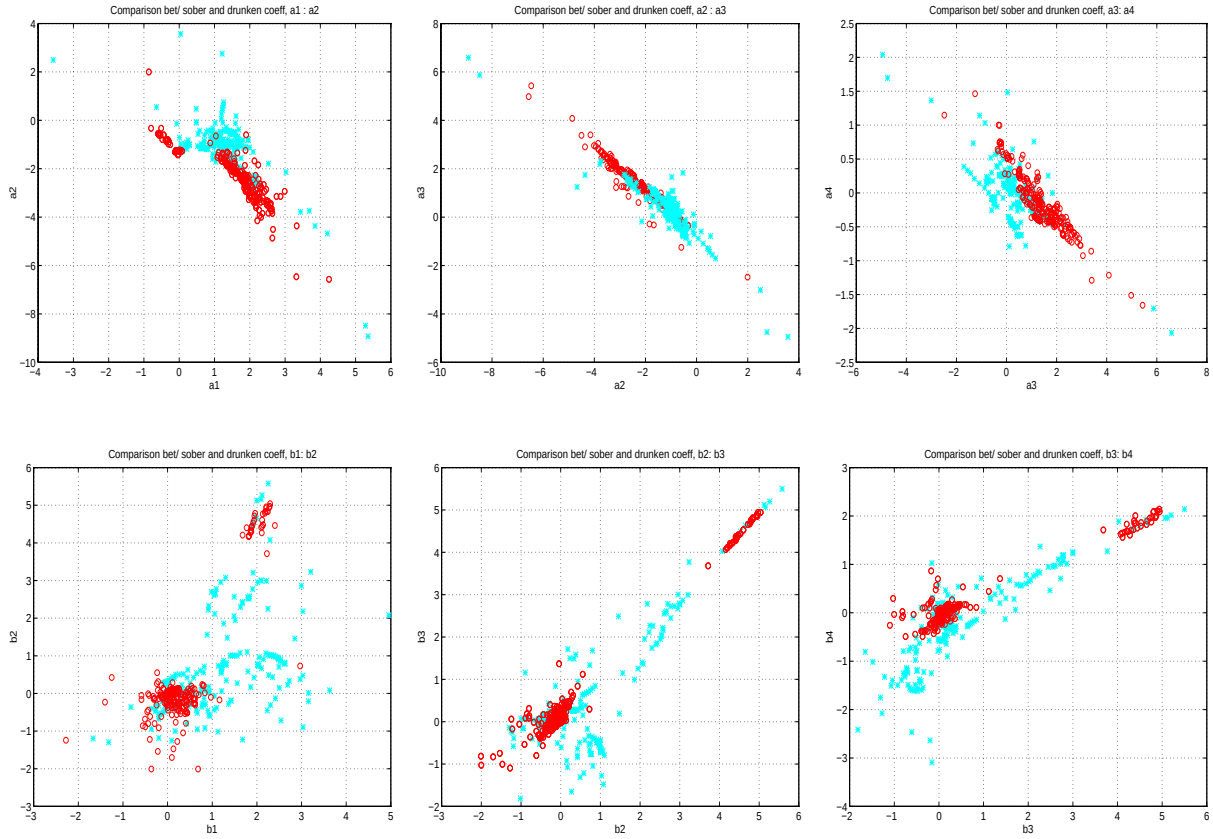

Figure 8-5: ARMA(4,4) parameters of the sober driving behavior are deviated from those of the intoxicated ones.

computational efficient, it consumes fewer data, and finally it is an on-line system, with more data involved in, it becomes more precise.

In next chapter, we will ask OMEGA to handle an even harder problem. We will see OMEGA performs more accurately than the other competing methods.