CHAPTER 4

THE CHANGE DETECTION AND ESTIMATION SYSTEM

In the previous chapters it was accounted for how the changes in the engine condition are sensed by sensors and transformed into feature signals expressing the engine condition changes. At this point feature signals are provided and from these it might be possible to detect and segment the engine condition changes. This chapter will investigate the next step in the automatic condition monitoring system, which is segmentation. As mentioned in chapter 2 the segmentation task is divided into three sub-tasks, which are:

- On-line change detection algorithm.
- Off-line hypothesis testing.
- Off-line change point estimation.

These sub-tasks will be described, and in the end of the chapter, a short discussion on a term called *panel of experts* is provided.

4.1 Deviation of mean and deviation – on-line change detection algorithm

The main objective with this work is to develop a system that detects changes in the scalar parameters of a Gaussian distributed signal without knowing the parameters of the new distribution, i.e. unknown change detection. During the first period of the work focus was on several *known* change detection algorithms and few *unknown* change detection algorithms suggested by [1]. One could wonder why the *known* change detection algorithms were not skipped, since *unknown* change detection plays the most important role in this thesis. However, it was decided that this could act as a good point of development to understand the *unknown* change detection issues. This decision caused both good and bad experiences.

Among the good things, quite a lot of change detection definitions and problems were experienced. On the other hand it turned out that only a couple of the known change detection algorithms could be modified to *unknown* change detection algorithms. There is a problem in that the change detection problem is split in three cases:

- Change in the mean value of the Gaussian distributed signal. The variance is constant.
- Change in the variance. The mean value is constant.
- Change in both mean value and variance.

Only the first case (change in the mean value) is considered, but it is possible to derive the decision functions of the other cases too. The problem for these unknown change detection algorithms is that they must know which case the change is, and this is not feasible since the new distribution is unknown. If you know which case is present, you also know that a change

has occurred, and why then run an algorithm, which decides whether a change has occurred or not?

Therefore, these algorithms were abandoned and a new unknown change detection algorithm was developed. The algorithm is called the Deviation of Mean and Deviation (DMD) algorithm. During the investigations on the before mentioned algorithms a very interesting property was observed, which turned out to be the first key point in the new algorithm,

Key point 1:

The empirical deviation of a Gaussian distributed signal shows significantly peaks when the signal changes from one distribution to another.

This property is also applied in the Filtered Derivative Algorithm (FDA) [1], but only for a change in the mean value. The discrete deviation of a Gaussian distributed signal is visualized in figure 4.1-3 for the three cases. Good ratios[1] are chosen in order to give a better understanding of the idea. In the case of a change in the mean value (figure 4.1) it is clear that a peak appears in the neighborhood of the change point. This is also the case in figure 4.3 – change in both mean value and deviation.

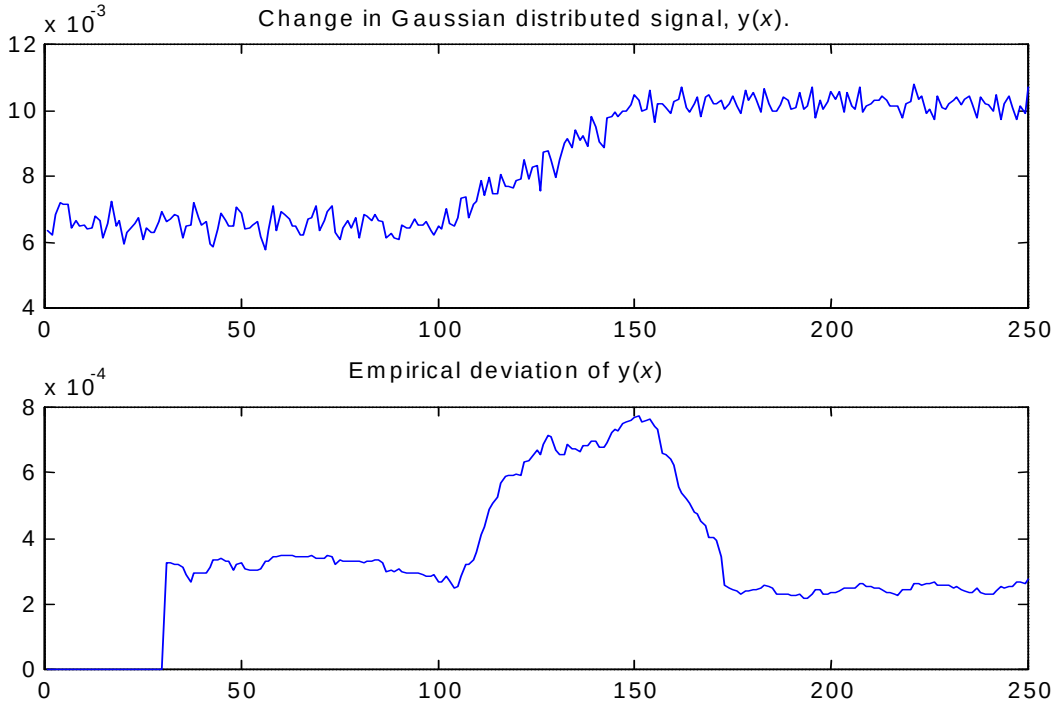

Figure 4.1: Change in the mean value of a Gaussian distributed signal and the empirical deviation of the Gaussian distributed signal.

[1] I.e. very clear difference between the conditions.

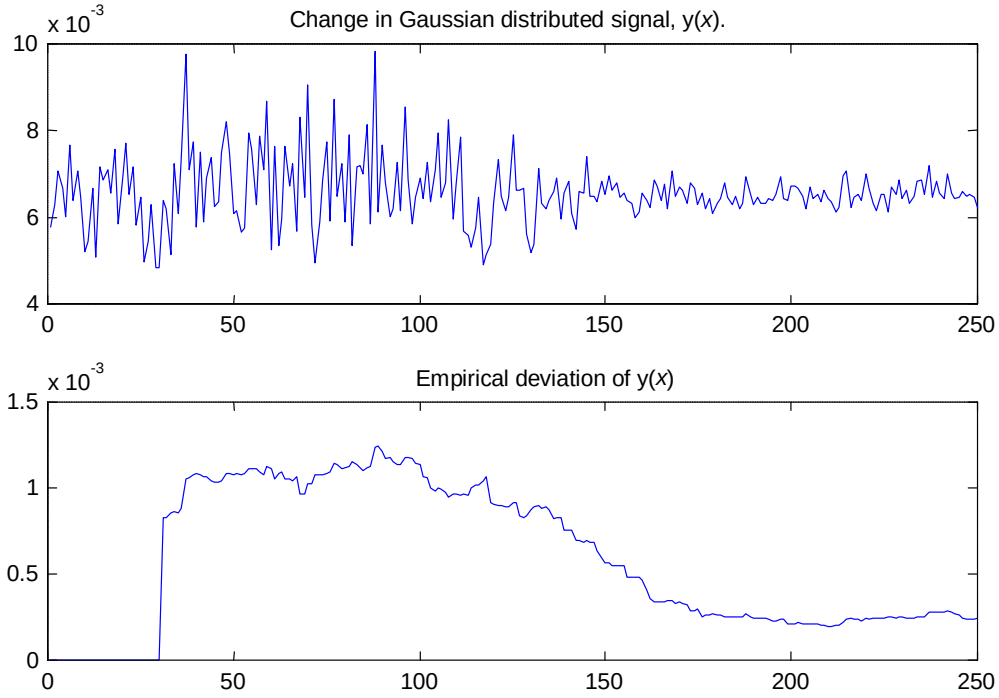

Figure 4.2: Change in the standard deviation of a Gaussian distributed signal and the empirical deviation of the Gaussian distributed signal.

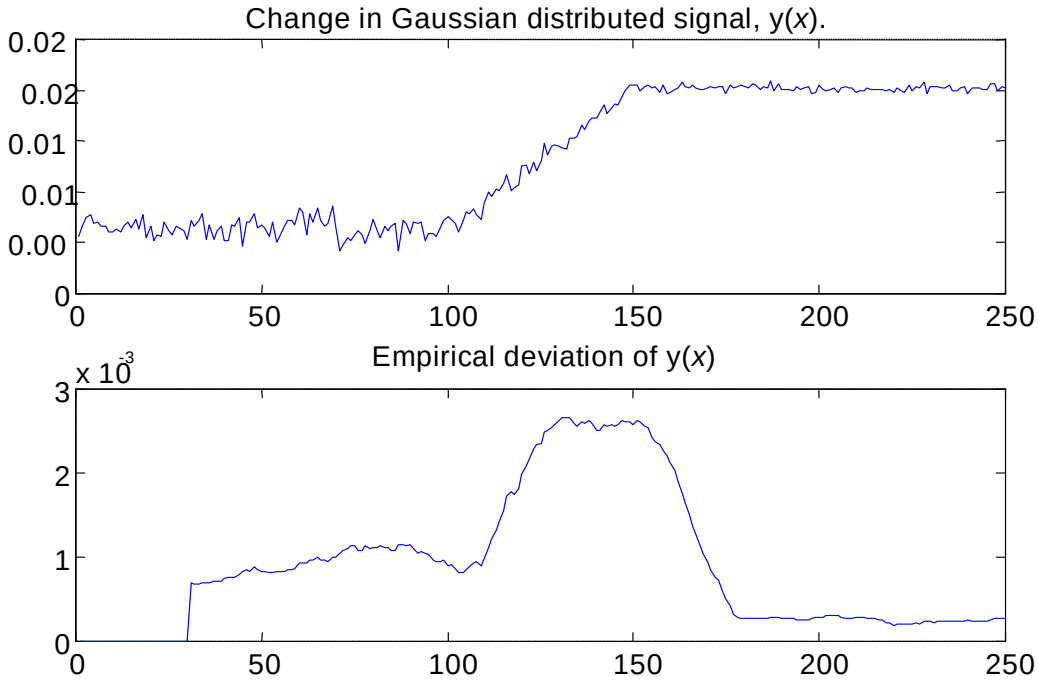

Figure 4.3: Change in the mean value and the standard deviation of a Gaussian distributed signal and the discrete deviation of the Gaussian distributed signal.

If we look at the second case of changes, figure 4.2, we are faced with a new situation since the change does not correspond to a peak but a change in level from high deviation to low deviation. This is in fact a problem if the system only searches for peaks in the decision function (in this case the empirical derivative of the Gaussian distributed signal). Figure 4.4 is the same as figure 4.2[2], except that a threshold is inserted above the high deviation with the aim of detecting a peak. This mission will never succeed since the deviation changes to a lower level beyond the threshold. At this point the second key point in the DMD change detection algorithm emerges,

Key point 2:

The feature signal $y(x)$, i.e. a Gaussian distributed signal, is split in two signals: The mean value $\mu(x)$ and the standard deviation $\sigma(x)$. Next, the standard deviation of $\mu(x)$ and $\sigma(x)$ is calculated giving two new signals $\sigma_{\mu(x)}$ and $\sigma_{\sigma(x)}$. Whenever a change in the mean value happens in $y(x)$, a peak occurs in $\sigma_{\mu(x)}$, and whenever a change happens in the deviation of $y(x)$, a peak occurs in $\sigma_{\sigma(x)}$.

From this, the name of the algorithm is given, since it searches for changes in the mean value and the deviation of a feature signal by means of the *deviation of the mean value and the deviation* of the feature signal.

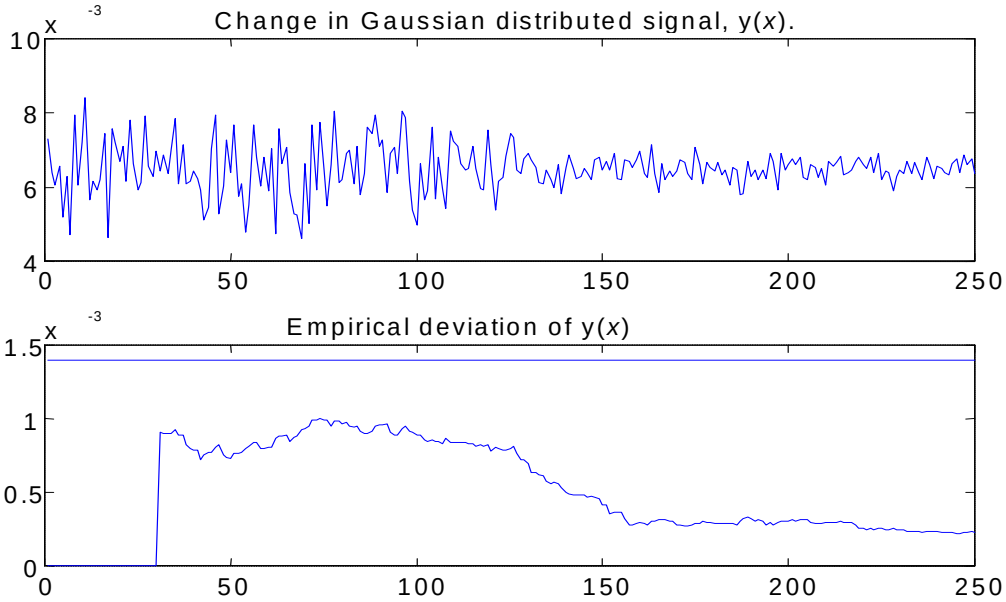

Figure 4.4: Same situation as in figure 4.2, but here a threshold is inserted in order to detect a peak corresponding to a change. This fails.

[2] Read: The log-likelihood ratio is the same. The sequences of pseudo-random numbers are not equal, but this is irrelevant in this context.

4.1.1 DMD parameters

The DMD algorithm has four main parameters, which are,

- *mwl*: Main Window Length: The empirical mean value and the empirical standard deviation of the feature signal are calculated at each cycle from the *mwl* recent cycles of the feature signal.

- *cal_wl*: CALibration Window Length: The DMD algorithm uses *cal_wl* cycles for calibrating or training to the normal condition.

- *swl*: Small Window Length: The empirical deviation of the empirical mean value and the empirical deviation is calculated as the numerical difference between the mean values of the cycles in two small windows of length *swl* separated from each other.

- *sw_dist*: The distance between the two small windows.

Figure 4.5 shows the DMD parameters. One could argue that a single window can replace the two small windows and the deviation in this window can be estimated. However, this has been investigated, and the result was not adequate, since the peak in the decision functions was too poor to use as detection point.

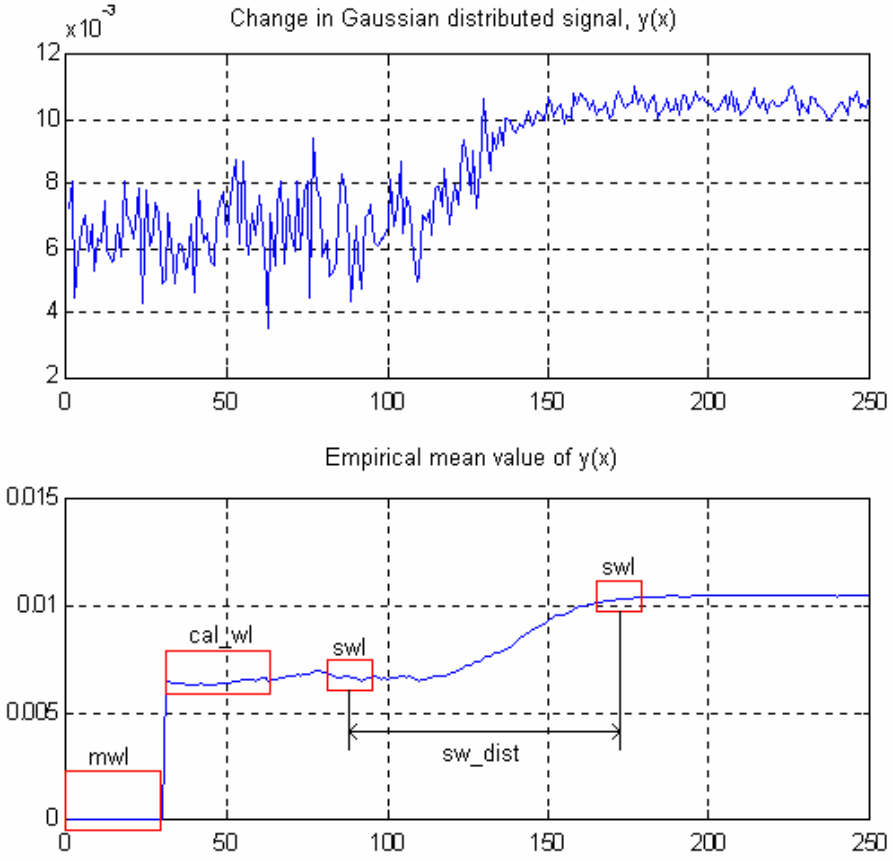

Figure 4.5: The four main DMD parameters.

4.1.2 Estimating the beginning of the new condition

For this work a major central problem is to pick out a window of cycles and use it for off-line investigations[1]. It is not enough just to detect the first change point (the cycle where the engine leaves the normal condition) and then assume when the new engine condition begins. This is because both abrupt and non-abrupt signal changes are present in the feature signals, see figure 4.6. If any assumption on the second change point is made, one have to know more about how this point is related to the first change point, a priori. This is not realistic when only a single test is run on the test engine, which is the case in the work.

Therefore, it was decided to estimate the second change point. In the following the first change point is denoted cp_1, and the second change point cp_2. The key idea in change point 2 estimation is to use the property that, when a sub-feature signal changes from one condition to another, it will give rise to a peak in the empirical deviation of the sub-feature signal. If we look at figure 4.7 it is clear that cp_1 corresponds to the start of the peak, and that cp_2 corresponds to the point where the peak has reached its maximum value.

It is far easier to detect cp_1 since we know more about the normal condition than the new condition. E.g. one could investigate the decision function in the initialization period and from this set a threshold for cp_1. This is unfortunately not true for cp_2 since we do not know from the normal condition where the peak in the decision function is placed. To find a maximum value, one classical approach is to calculate the gradient and solve when it equals zero. This was the first idea, but it was realized that there was a potentially risk in this method, because the peak very seldom is smooth enough. This will result in estimated change points that are to early with relation to the true change point, i.e. false alarms occur. It was chosen to use the

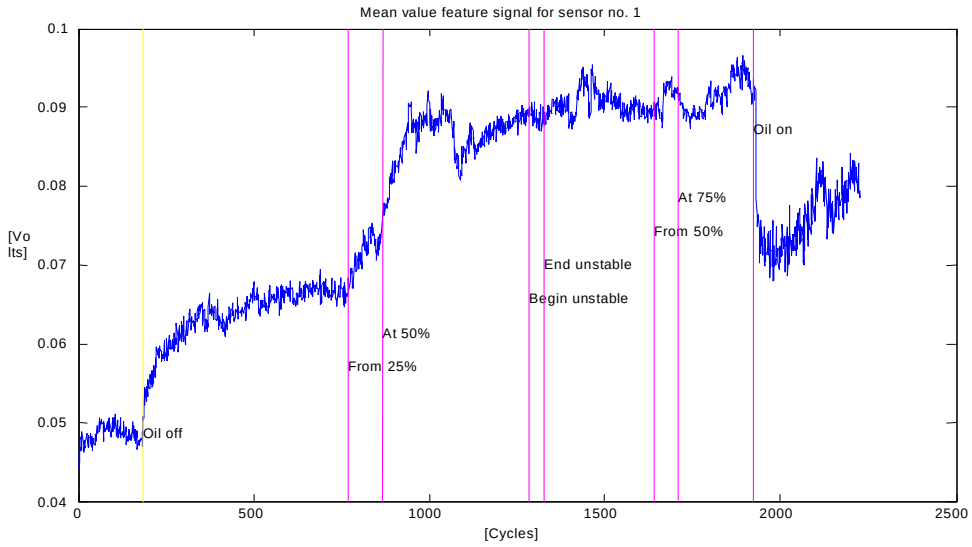

Figure 4.6: Full feature signal from sensor no. 1. The mean values of the cycles are used. Notice the long change in experiment no. 1 (oil off) and the abrupt change in experiment no. 6 (oil on).

[1] Off-line hypothesis testing and off-line change point estimation.

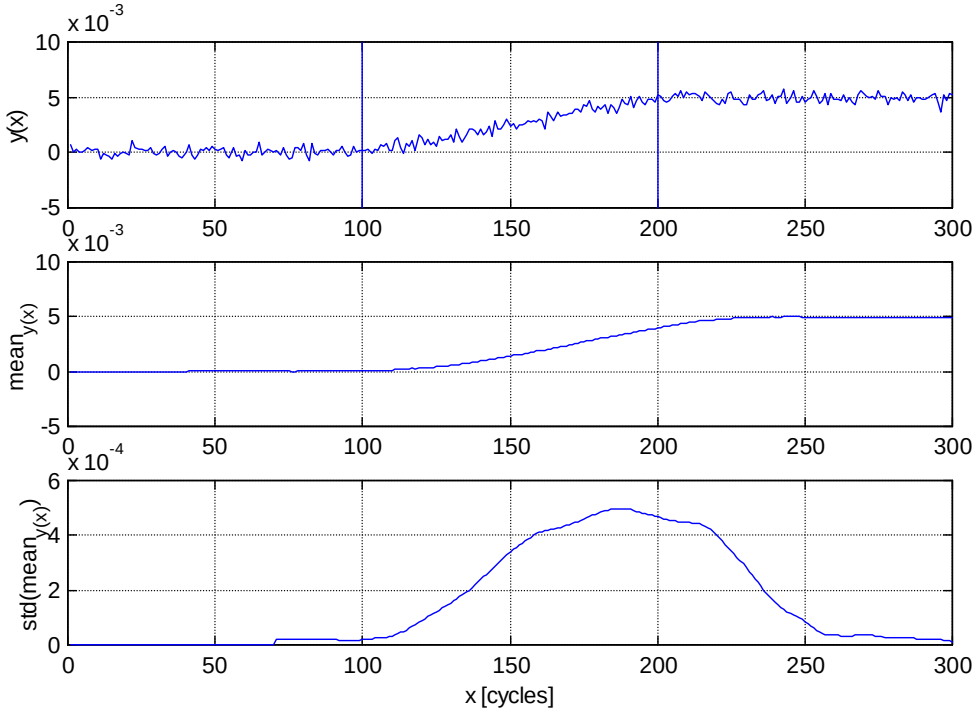

Figure 4.7: The two change points can be estimated by means of the peak in the empirical deviation of the sub-feature signals.

same threshold for cp_2 as for cp_1, since there at this stage in the work was no need of precise detection of cp_2.

Using the same threshold for both change points has the direct consequence that cp_2 is detected later in most cases. But this is no major problem at the moment, because we have to use a number of samples after cp_2 in order to gain good performance of the off-line parts of the change detection system. However, in the future it is preferred that cp_2 is estimated as good as possible, since this will make it easier for the off-line parts to do their job.

4.1.3 The Boost function

In change detection, one of the most central problems is how to place a proper threshold on the decision function so that the probability of false and unseen alarms is low and the probability of true alarms is high. This problem is often solved by means of a ROC curve, where different thresholds are applied on the decision functions and the response on the true, false and unseen alarm probabilities is shown.

However, in some cases there is a small clearance for a threshold, i.e. it should be within an interval that is very narrow. In figure 4.8 a feature signal is plotted together with its sub-feature signal, the discrete mean value of the feature signal, and the corresponding decision function, the discrete deviation of the sub-feature signal. The feature signal is generated by means of the randn() function in Matlab, and consists of a normal condition which is a $N(\ 0,\ (0.38\text{m})^2\ )$ distribution, a new condition which is a $N(\ 1,\ (0.38\text{m})^2\ )$ distribution and a linear drift from the normal condition to the new condition. All three regions are 100 cycles long.

From figure 4.8 it is clear that one can place a threshold properly, but the ratio between the peak corresponding to the true alarm versus the peaks corresponding the false alarms is not very good, i.e. the probability of false alarms is likely to be too high. The boost function improves this ratio in the way that sub-feature signal samples lying within a specific interval is diminished and the samples lying outside the interval are enhanced.

Figure 4.9 shows the effect of using the boost function on the sub-feature signal. The clearance for the threshold is increased significantly, which can be measured directly by calculating the ratio between the peaks from the false alarms and the true alarm. Unfortunately, the improved clearance causes a later detection of the first change point if one want to exploit the full clearance. However, this is not a major problem if the whole change detection system is considered, and not only the on-line change detection part – the important thing here is to detect the two change points, when the engine leaves the normal condition and when it enters the new condition. Then the off-line change point estimation part will improve the alarm times.

A passing remark to figure 4.9 is that the decision function has a smoother peak, which intuitively should make it more possible to use a gradient method to find the second change point faster, than using the same threshold for the two change points.

Before proceeding with the description of the boost function it is necessary to explain that the sub-feature signal undergoes a kind of normalization before the boost function is applied on it. The mean value of the first cycles are subtracted from the sub-feature signal. In this way the upper and lower thresholds are equal in numerical value, and this makes it far easier to solve the succeeding steps in the boost function, as will soon be shown.

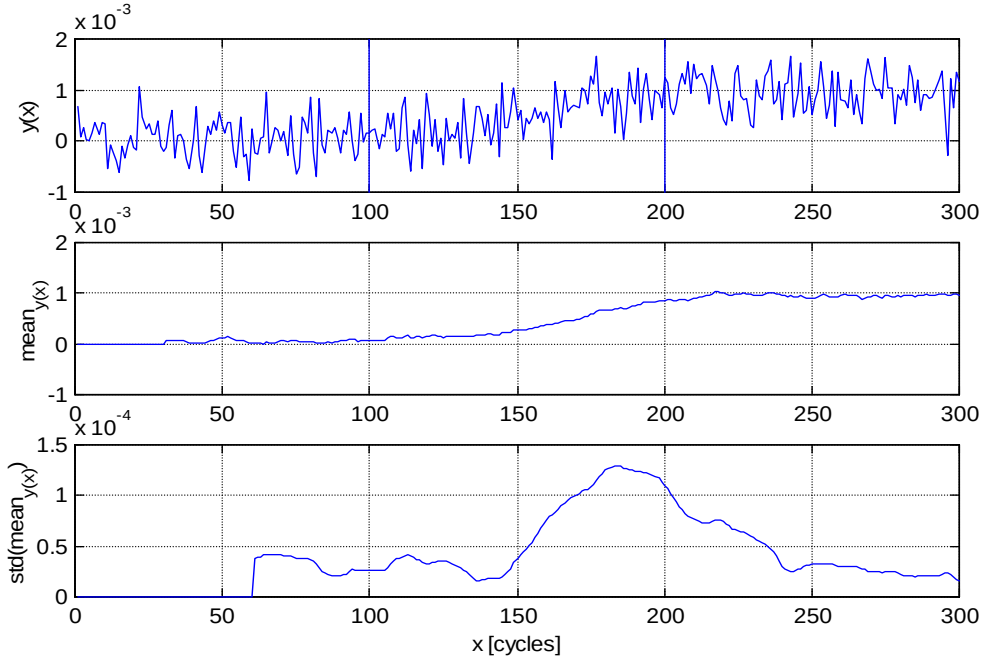

Figure 4.8: The clearance of the threshold for the decision function is relatively small.

4.1.3 The Boost function

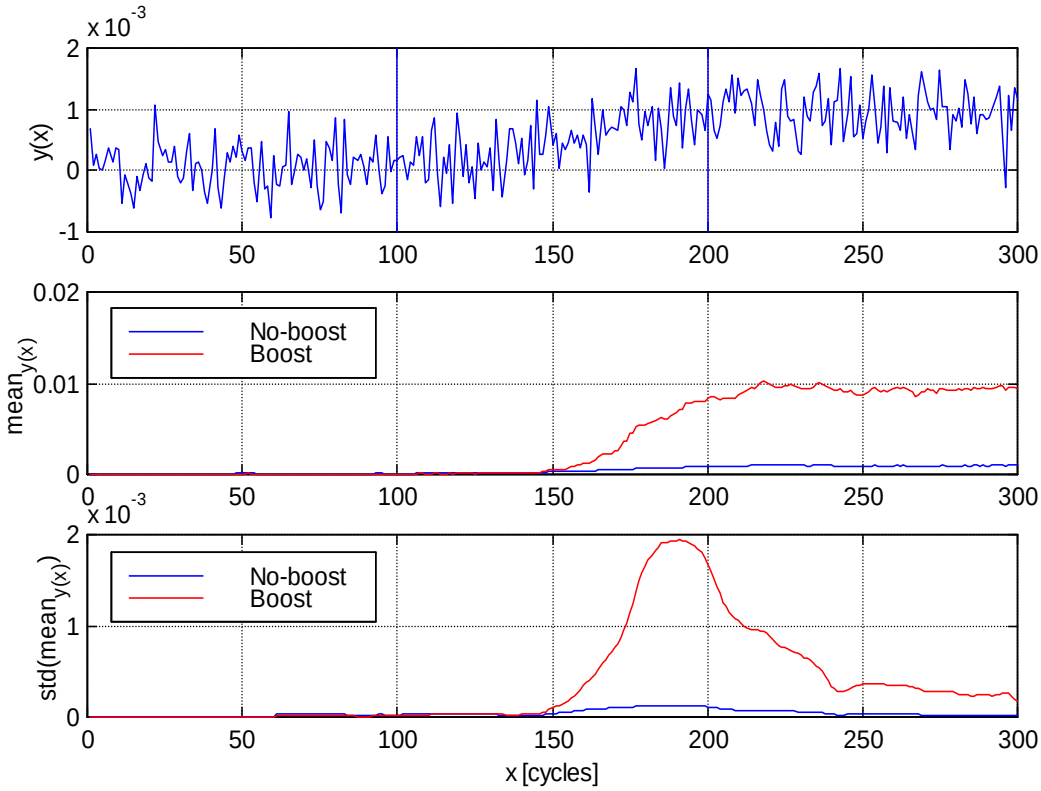

Figure 4.9: The clearance of the threshold for the decision function is increased when the boost function is applied on the sub-feature signal.

The main criteria for the boost function are as follows:

- When a sample x_i lies *inside* a decided interval, it must be multiplied by a factor between 0 and 1.
- When a sample x_i lies *at the boundary* of the decided interval, its value does not change.
- When a sample x_i lies *outside* a decided interval, it must be multiplied by a factor larger than 1.

Several functions can be used, but the hyperbolic tangent of x, $\tanh(x)$, is chosen from intuitive arguments that it moves towards a fixed value when $x \to \infty$ and $x \to -\infty$ and that it changes smoothly from the low level to the high level. In order to fulfill the main criteria previously mentioned, some modifications have to be made with $\tanh(x)$. Figure 4.10 shows the changes. The first plot in the figure is the original hyperbolic tangent of x. Here it is clear that $\tanh(x) \to 1$ and $\tanh(x) \to -1$ when $x \to \infty$ and $x \to -\infty$, respectively. This is not enough to fulfill the criteria so two things remain. First of all 1 is added to $\tanh(x)$, thereby moving the function upwards, so that $\tanh(x)+1 \to 2$ and $\tanh(x)+1 \to 0$ when $x \to \infty$ and $x \to -\infty$, respectively. Now we have a function with minimum value equal to 0. The next thing is to shift this function to the right, since x must be positive. It seems reasonably to shift the function by 3, because then $\tanh(x-3)+1$ is close to zero when x is close to zero.

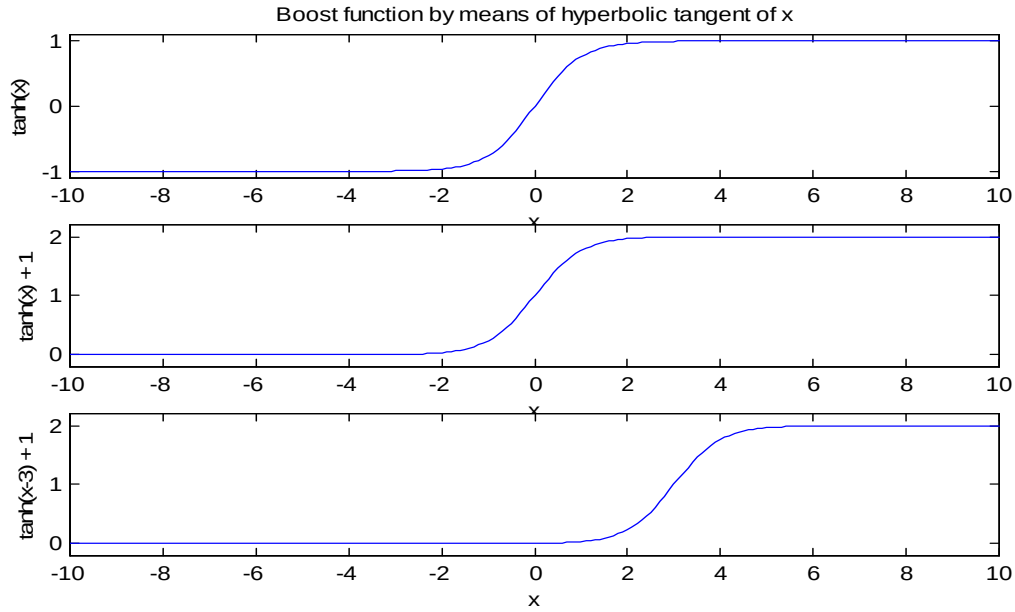

Figure 4.10: Modification of the tanh(x) function into a useful boost function.

Next, the boost function parameters A, h_0, and $h_{A/2}$ are introduced:

- A: The maximum value of the boost function. At the moment $A = 2$.
- h_0: The threshold for the interval in which samples are likely to be, if they belong to the normal condition.
- $h_{A/2}$: The threshold where the sample is multiplied by A/2.

Figure 4.11 shows the three boost function parameters. To avoid future confusion about the axis, x is replaced by $y_{con}(x)$, since the sub-feature signals magnitude has to be boosted, and this is given on the y-axis, not the x-axis. The index *con* is used to distinguish between $y(x)$ and the argument to the boost function, since some modification on $y(x)$ has to be done before it is transferred to the boost function.

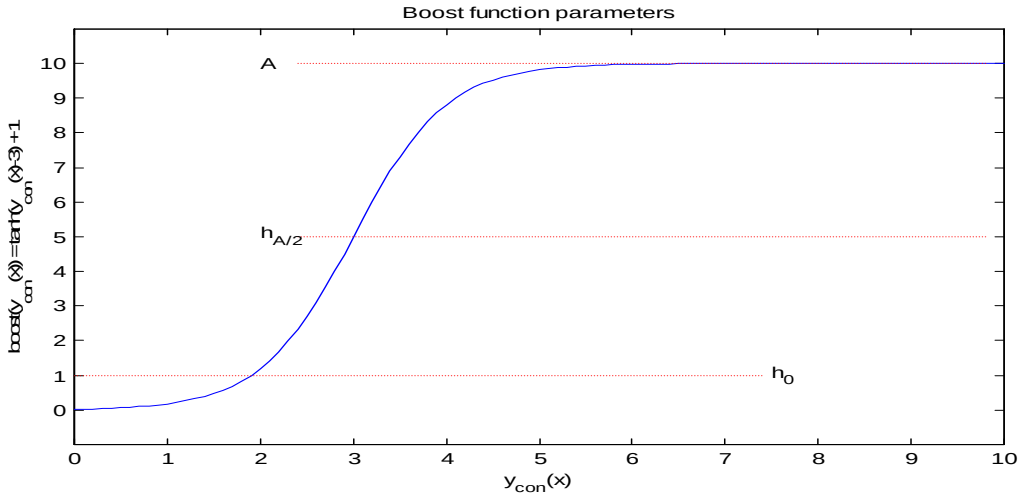

Figure 4.11: The boost function and its three main parameters.

4.1.3 The Boost function

The next task is to connect the boost function to the sub-feature signal, i.e. determine the connection between $y(x)$ and $y_{con}(x)$, and this is not easy. First of all only positive values of the sub-feature signal is considered, but since the mean value of the sub-feature signal from the normal condition is subtracted from the sub-feature signal, the same can be done with negative values, if the sign of the sub-feature signal sample is remembered. Three points are defined for the boost function, a_{boo}, b_{boo}, c_{boo}, and three similar points are also defined for the sub-feature signal, a_{sub}, b_{sub}, c_{sub} - figure 4.12. These two set of points are related to each other by the following,

$$\begin{aligned} y(x) = a_{sub} &\Rightarrow y_{boo}(x) = a_{sub} \cdot boost(a_{boo}) = a_{sub}, \\ y(x) = b_{sub} &\Rightarrow y_{boo}(x) = b_{sub} \cdot boost(b_{boo}) = b_{sub} \cdot A/2, \\ y(x) = c_{sub} &\Rightarrow y_{boo}(x) = c_{sub} \cdot boost(c_{boo}) = c_{sub} \cdot A, \end{aligned} \tag{4.1}$$

where the new boosted sub-feature $y_{boo}(x)$ is defined as,

$$y_{boo}(x) = y(x) \cdot boost(\, y_{con}(x)\,) \tag{4.2}$$

where y_{con}(x) denotes the connected sub-feature signal value and where the boost points are related to the boost function by,

$$\begin{aligned} boost(a_{boo}) &= 1, \\ boost(b_{boo}) &= A/2, \\ boost(c_{boo}) &= A. \end{aligned} \tag{4.3}$$

The main idea of boosting the sub-feature signal is that the further away $y(x)$ is from a_{sub}, the higher the $boost(\, y_{con}(x)\,)$ should be. In some cases, $y(x)$ can be so large that $y(x)$ always will be multiplied with A. This will not improve the ratio between the false alarm peaks and the true alarm peak in the decision function, since they will all be multiplied by A.

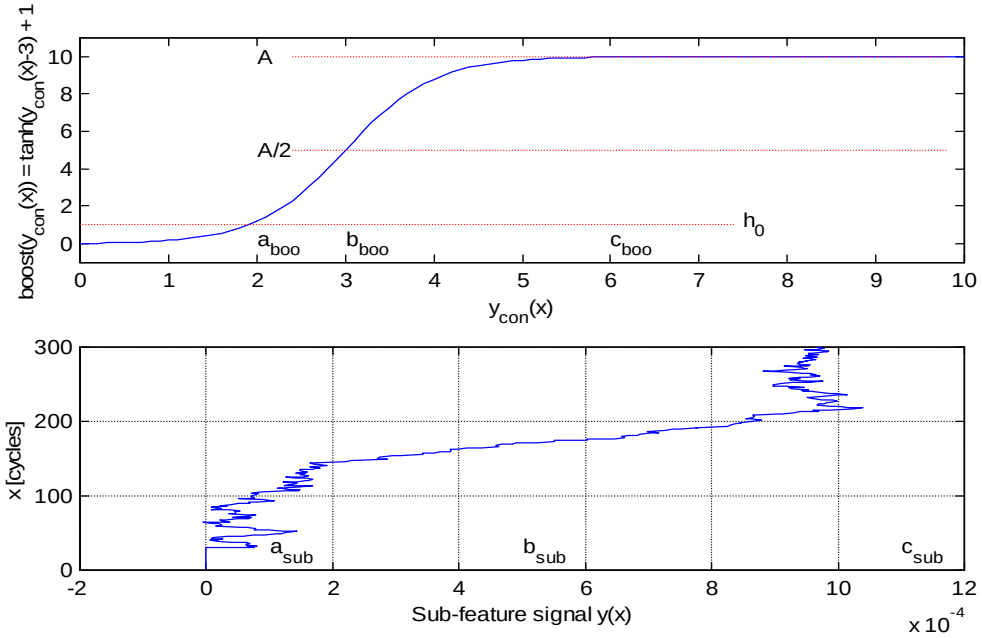

Figure 4.12: The boost function versus the sub-feature signal.

One solution to avoid this is first of all to skip the *c*-parameters and only connect the boost function and the sub-feature signal with each other by using the *a*- and *b*-parameters. Next, a_{sub} is determined by setting it equal to the 100th percentile[3] of the sub-feature signal distribution in the calibration period. For determining b_{sub} it is important to realize that the closer b_{sub} is to a_{sub}, the higher is the risk of "saturation", i.e. $boost(\ y_{con}(x)\ ) \approx A$. On the other hand, if b_{sub} is far away from a_{sub}, no boost will occur because $boost(\ y_{con}(x)\ ) \approx 1$ in this case. There is no satisfactory answer to this problem, but b_{sub} was set equal to 1.5 times a_{sub} in this section. In the future work the determination of b_{sub} must be investigated, but for now it works fine though the risk of "saturation" is high since b_{sub} is close to a_{sub}.

The remaining problem now is to find $y_{con}(x)$ when $y(x) \neq a_{sub}$ and b_{sub}. This is illustrated in figure 4.13. Two axis are shown, the upper with the $y(x)$ values and the lower with the $y_{con}(x)$ values. The *a* and *b* parameters are plotted on their respective axis, and a specific value of $y(x)$, i.e. $y\ (x_i)$, is also plotted on the $y(x)$-axis. Then the first of four lengths, *L1*, is calculated as the distance from a_{sub} to b_{sub}. The second length, *L2*, is calculated as the distance from a_{sub} to $y(x_i)$. The third length, *L3*, is calculated as the distance from a_{boo} to b_{boo}. Then it has to be found out which value $y_{con}(x_i)$ corresponds to $y(x_i)$, which is easy, since the ratio between *L2* and *L1* must be equal to the ratio between *L4* and *L3*. Thus $y_{con}(x_i)$ is equal to $L4 + a_{boo}$.

At this point one question arises: Why is it necessary to modify the sub-feature signal before passing it to the boost function? Will it be enough to say that $y_{boo}(x) = boost(\ y(x)\ )$? If one chooses to do so, it is clear that $y_{boo}(x)$ will be in the range 0 to A, and when the range of $y(x)$ in the new condition is unknown, there is a risk of saturation. In this case the second alarm in the on-line change detection algorithm will be set too soon, i.e. in the drift. This causes that no proper window is transferred to the off-line algorithms, and thereby no reliable change detection and change point estimation can be done. If the boost function is used as a "factor" function, no true saturation will occur, since the range of $y_{boo}(x)$ will be in the range from 0 to $A \cdot y(x)$. The only thing that happens is that the ratio between the peaks in the decision function from the false alarms and the true alarm wont be improved more.

Another question is which effect it will have to increase A. Is the clearance for the threshold thereby increased? Unfortunately an increase from A to $F \cdot A$, where F is a factor, will not improve the clearance for the threshold, since both the false alarm peaks and the true alarm peak will be multiplied with F. This means that the ratio between the peaks will be constant, and thus no improvement is feasible. The only way to increase the clearance is to shift b_{sub} away from a_{sub}[4], and at the same time avoid saturation. In fact the only effect of increasing A is that the peaks within the threshold interval h_0 are diminished more, and the peaks outside the interval are enhanced more.

One of the advantages with the boost function is to improve the log-likelihood ratio of the sub-feature signals, but the new condition does not need to be raised to a very high level to provide an acceptable log-likelihood ratio. Therefore, the temporary investigations suggest that $\max(A)$ should be no more than 20. The minimum value must be greater than 1 since $boost(a_{boo})$ is defined to be 1. A is set equal to 10 in the generation of the figures in this section.

[3] Other percentiles have also been investigated, but at this point it seems reasonable to use the 100th percentile.
[4] The result is the same if b_{boo} is shifted towards a_{boo}.

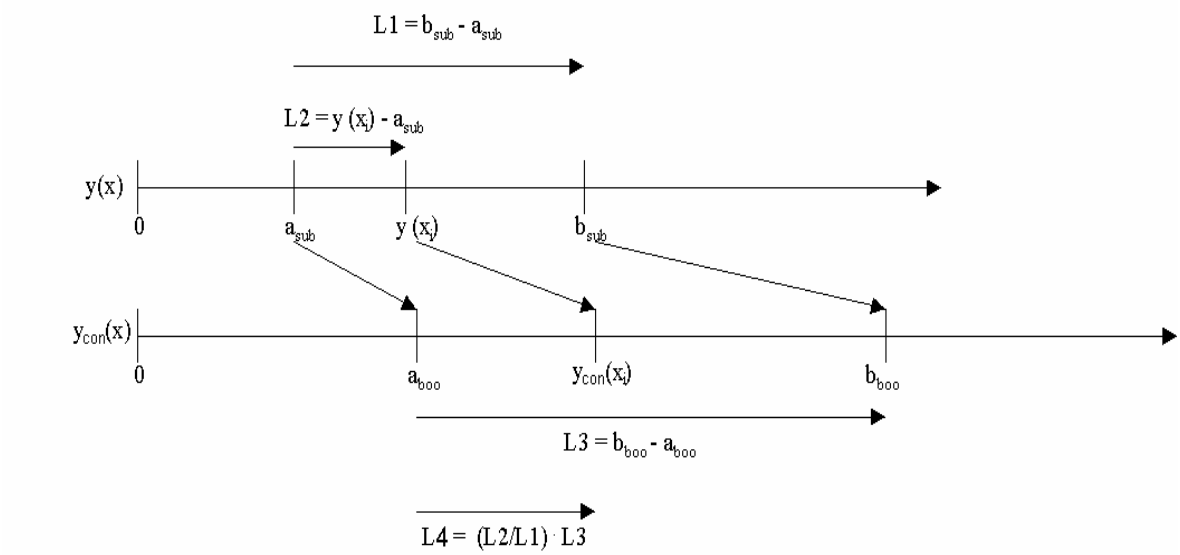

Figure 4.13: How to connect the boost function and the sub-feature signal.

4.1.4 Code of boost.m

```
function [yboo] = boost(a_sub, b_sub, y)

% Define A and the shift on the axis:
A     = 10;
shift = 3;

% Calculate a and b parameters for the boost function:
a_boo = atanh(2/A -1) + shift; % boost( a_boo ) == 1
b_boo = atanh(4/A -1) + shift; % boost( b_boo ) == A/2

% Calculate lengths:
L1 = b_sub  - a_sub;
L2 = abs(y) - a_sub; % Notice the absolute value of y
L3 = b_boo  - a_boo;
L4 = (L2/L1)*L3;

% Find the connected y:
y_con = a_boo + L4;

% Return the boosted sample:
yboo = y*(A/2)*( 1 + tanh(y_con-shift) );
```

4.1.5 The DMD plot

At this point all the principles in the on-line change detection algorithm, developed for the automatic condition monitoring system in this work, have been described. An important issue has not been discussed. How the parameters in the algorithm influence the overall algorithm performance? This has not been highly prioritized, but here, a short discussion on the choices of the parameter values made for the thesis, follows:

- *cal_wl* is set to 30 cycles due to the Central Limit Theorem (CLT), [3]. In the calibration period the boost function is initialized. The longer the calibration period is, the more fluctuations of the feature signal in the normal condition are included. Thus, false alarms are less likely to occur. However, if the calibration period is too long,

there is a risk of not detecting the next engine condition change, since this can be included in the calibration period.

- *mwl* is also set to 30 cycles due to CLT. The longer the main window is, the less abrupt is the drift, since the cycle corresponding to the engine condition change will give a smaller change in the empirical mean value and the empirical deviation (the sub-feature signals) of the feature signal. The smaller the main window is, the less smooth the sub-feature signals will be.

- *swl* is set to 10 cycles. It must be relatively small in order to detect a change in signal level.

- *sw_dist* is strongly related to the drift length. The longer drift, the longer distance. However, because of short amount of time in the normal conditions, the distance must be short, therefore it is set to 30 cycles.

With these parameters the DMD algorithm has been applied on a few feature signals, and the results are given in the following figures. These figures are called DMD plots, since they plot how the DMD algorithm works on the feature signals. The first signal in the DMD-plots is the feature signal. The next two signals are the sub-feature signals, i.e. the empirical mean value and the empirical deviation. Here the red signal is the boosted signal of the original sub-feature signal. A zoomed DMD-plot is given in figure 4.16-17 in order to visualize the difference between the boosted sub-feature signals and the originals. The two decision functions are shown next, in some cases with a threshold. The last signal is the alarm signal. When no alarm has been set, the alarm signal is zero. When the first alarm is set, the alarm signal is set to 1, and when the second alarm is set, the alarm signal is set to 2.

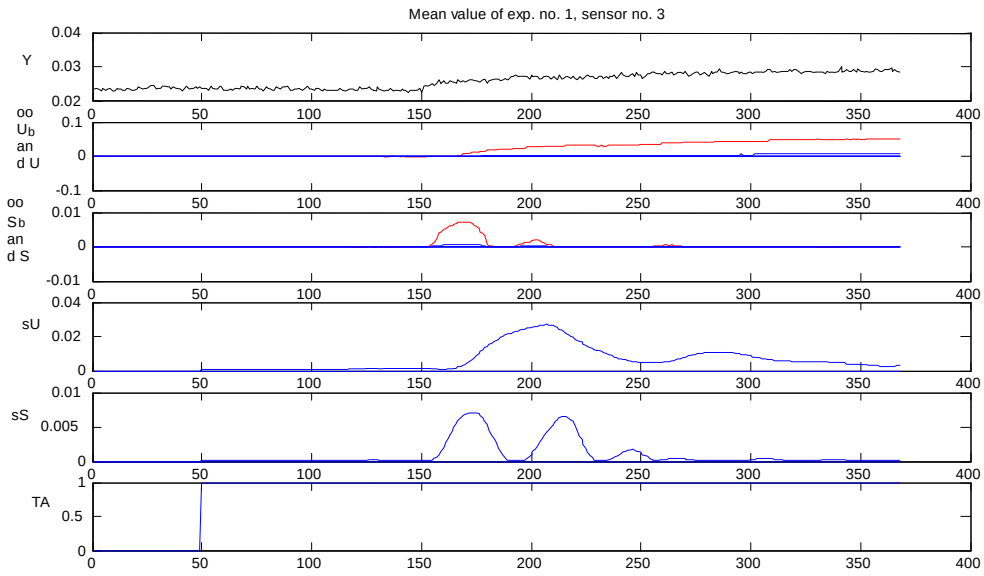

Figure 4.14: DMD-plot of experiment no. 1, sensor no. 3 (oil off).

4.1.5 The DMD plot

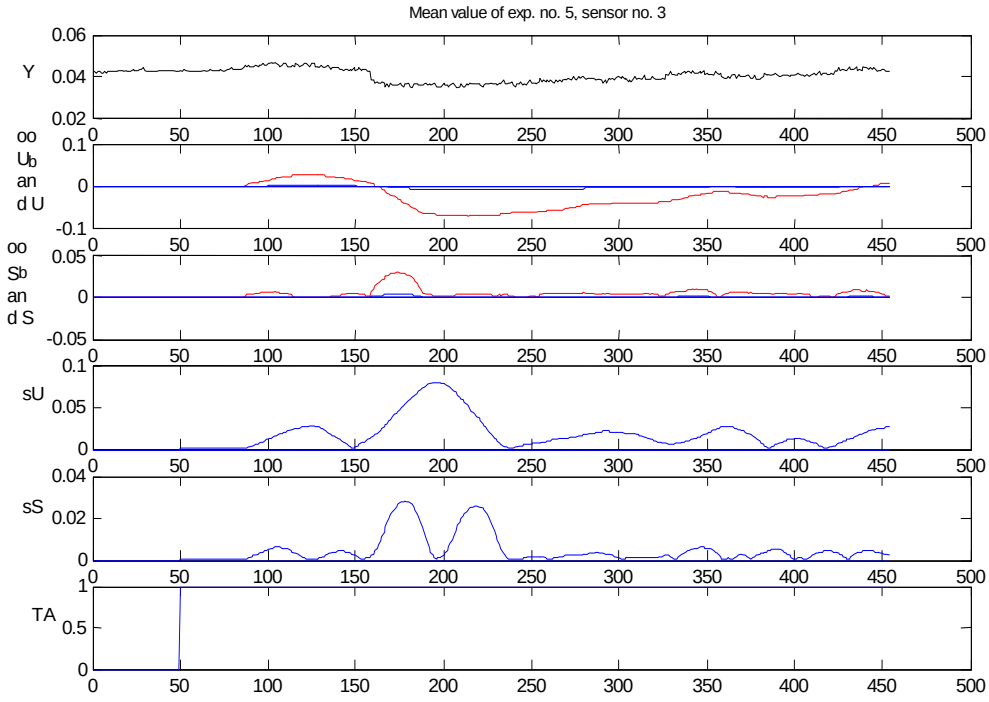

Figure 4.15: DMD-plot of experiment no. 5, sensor no. 3 (oil on).

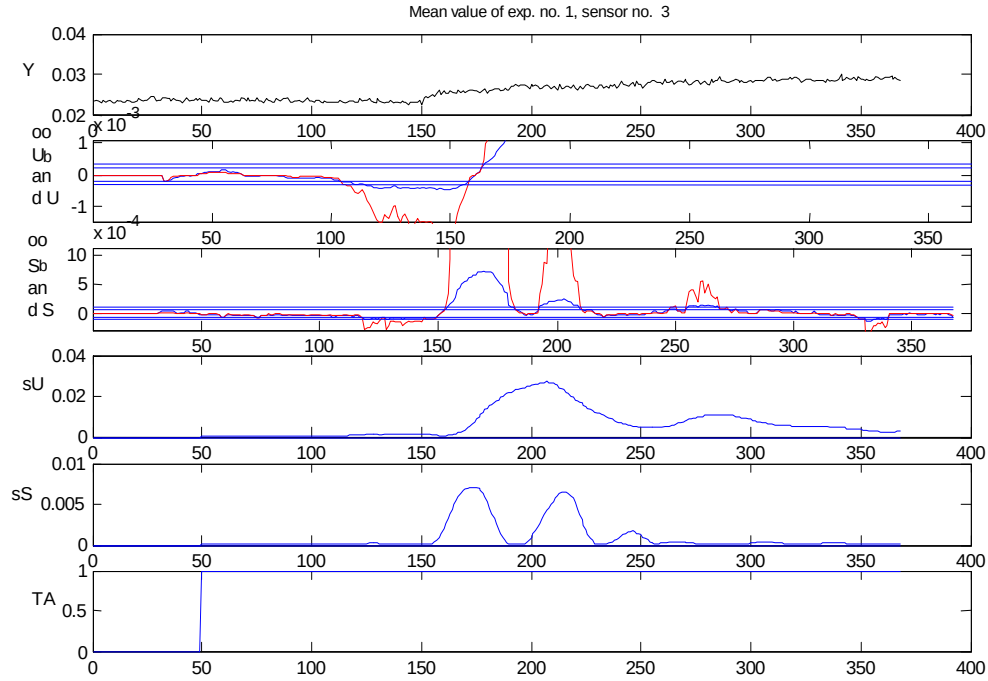

Figure 4.16: Zoomed DMD-plot of experiment no. 1, sensor no. 3 (oil on).

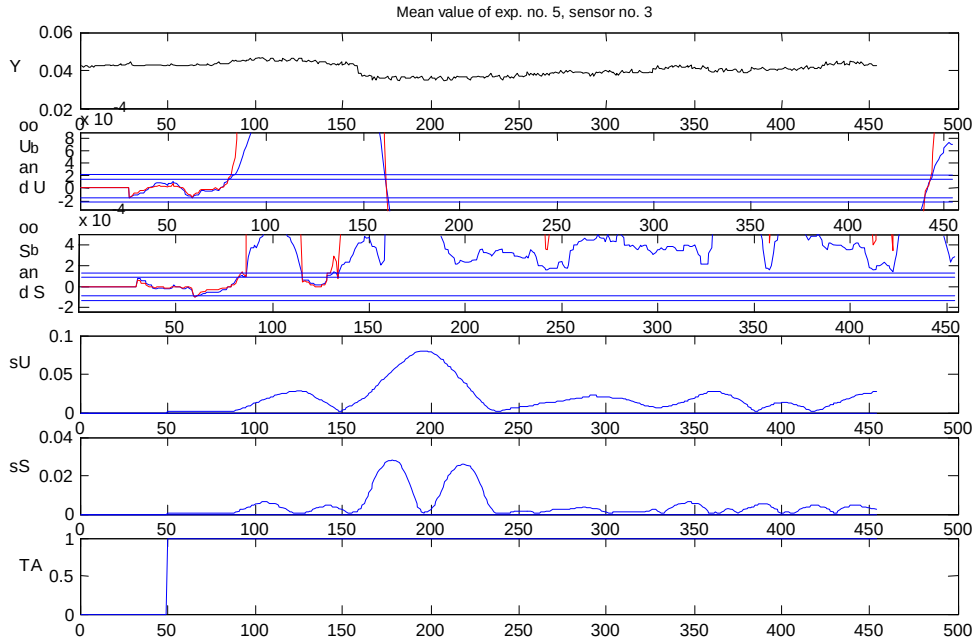

Figure 4.17: Zoomed DMD-plot of experiment no. 5, sensor no. 3 (oil on).

4.2 Off-line hypothesis test

When the on-line change detection algorithm has detected that the engine has changed condition, the second sub-task in the segmentation task is activated. The motivation is that the on-line change detection algorithm not necessarily has detected a change for real. It could be a false alarm, since the on-line algorithm is very simple and fast. Therefore, a hypothesis test is created in order to improve the overall systems reliability on whether an engine condition change has occurred or not.

The hypothesis is: “The engine has change condition”, and when the hypothesis is tested, the outcome will be that the hypothesis is either true or false (success or failure). Using an off-line hypothesis test in the automatic condition monitoring system, offers possibilities of regulating the on-line change detection algorithm. If the off-line hypothesis test is very reliable, then the thresholds for the decision functions in the on-line algorithm can be updated as time goes on. Imagine the situation where a large number of false alarms are present in a short amount of time. Here it could be preferred to enhance the threshold in order to increase the number of false alarms. However, one has to be careful doing this, since there is a risk of increasing the number of unseen alarms if the threshold is enhanced. In this thesis there will be no correspondence between the on-line change detection algorithm and the off-line hypothesis test due to lack of appropriate data set. The off-line hypothesis test applies a classical tool from statistics, the log-likelihood ratios, which will be explained in the next section.

4.2.1 The log-likelihood ratio for Gaussian distributed signals

When dealing with change detection in signals, one has to consider how to characterize a change. Several kinds of changes exist such as changes in the frequency, changes in the

magnitude, etc. In this thesis all feature signals are regarded as Gaussian distributed with a mean value μ and a variance σ^2, and changes in the feature signal are found by searching for changes in μ and σ^2. The mean value and the standard deviation are defined from [3] by,

$$\mu_X = \int_{-\infty}^{\infty} x \cdot p_X(x)\, dx, \tag{4.4}$$

$$\sigma_X^2 = \int_{-\infty}^{\infty} (x - \mu_X)^2 \cdot p_X(x)\, dx, \tag{4.5}$$

where X is a random variable and $p_X(x)$ is the probability density function of X. Since the variances of the feature signals are very small, it was decided to use the standard deviation σ instead of the variance. This also reduces the programming, since the outcome (almost every time) was that the standard deviation was used rather than the variance.

The log-likelihood ratio is a tool, which can be used to detect changes in a signal. In this work it is used also to measure how "good" a change is. E.g., the larger the difference between the mean value of the normal condition and the mean value of the new condition is, the easier it is to detect the change – this is a "good" change. This chapter offers a description of the changes in a feature signal, which is assumed to be Gaussian distributed and how the log-likelihood ratio is used in this project to characterize the different types of changes.

4.2.2 Feature signals and Gaussian distributions

Changes in the feature signals are treated as changes in the Gaussian distribution, i.e. changes in the mean value and the standard deviation. One could consider whether this approach is feasible in every situation or not, but this has shown to be true with this work. The argument is that given any feature signal, when no change occurs in the engine condition the feature signal must be in a certain region, which can be directly measured as a mean value with a standard deviation. When the engine condition changes, this should be responded by a change in the feature signal. Otherwise it is a poor feature signal. If the engine condition change is followed up by a new "stable" engine condition, then the feature signal in the same way also must respond by being "stable", and in a new region, which also can be regarded as a Gaussian distribution. In this way it is reasonable to regard the feature signals as Gaussian distributed signals, and the engine condition changes as changes in these Gaussian distributed signals. This simplifies the types of changes into three main cases, which are plotted in figure 4.18:

- Change in the mean value μ. The standard deviation σ is constant.
- Change in the standard deviation σ. The mean value μ is constant.
- Change in both the mean value μ and the standard deviation σ.

In fact eight cases of changes exist, since a change in the mean value can be to a lower level or a higher level. The same is true for the standard deviation, and therefore there are four cases of changes, when both parameters change. These eight cases are gathered in table 4.1 and plotted in figure 4.19-20. When e.g. a change occurs in a parameter from a lower value to

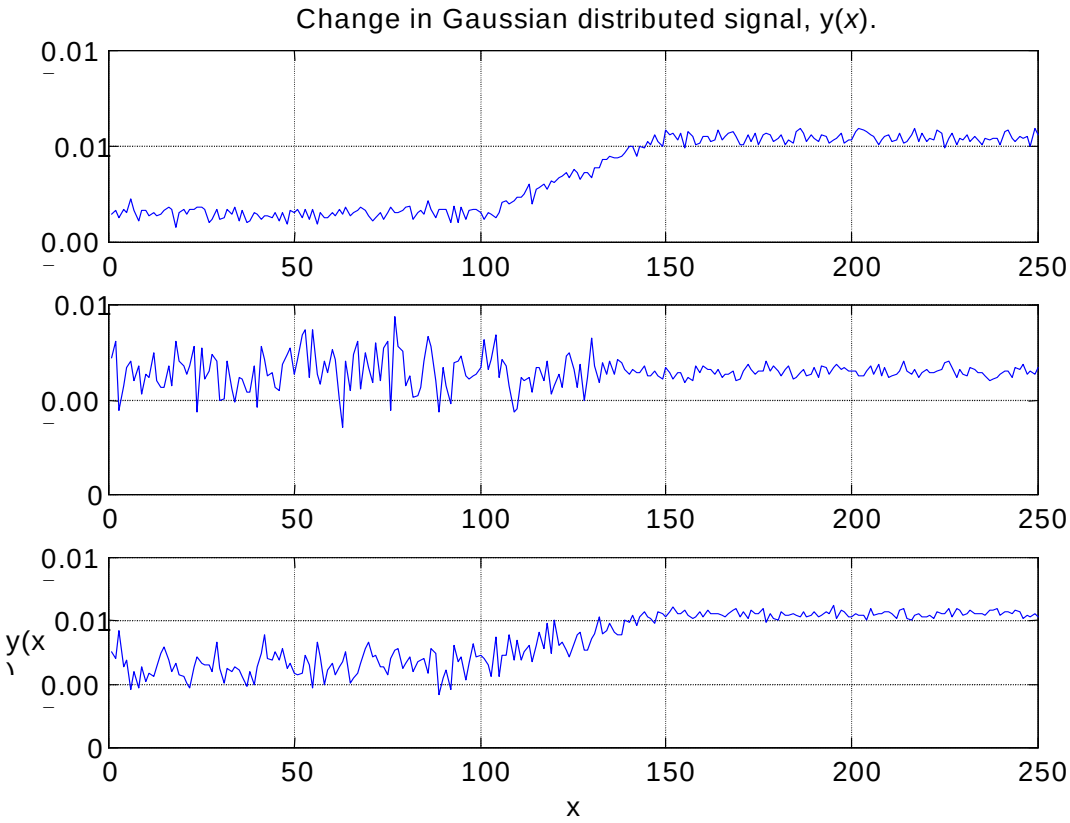

Figure 4.18: The three main cases of changes. Upper: Change in μ. Center: Change in σ. Lower: Change in both μ and σ.

a higher value, "*L*" for the normal condition parameter and "*H*" for the new condition parameter denote this. "-" means that the parameter does not change.

It is necessary to realize that in order to create a reliable and adequate change detection system, one has to consider how the change detection system perform in each of the eight cases. Imagine the situation when the mean value changes from a low value to a high value. If only this type of changes is assumed, only a single threshold is necessary. When the feature signal exceeds this threshold, the alarm is set. But if the change detection system also must detect changes from a high to a low level of the mean value, another threshold must be established.

Case	μ_0	μ_1	σ_0	σ_1
1	L	H	-	-
2	H	L	-	-
3	-	-	L	H
4	-	-	H	L
5	L	H	L	H
6	L	H	H	L
7	H	L	L	H
8	H	L	H	L

Table 4.1: All eight change cases.

4.2.2 Feature signals and Gaussian distributions

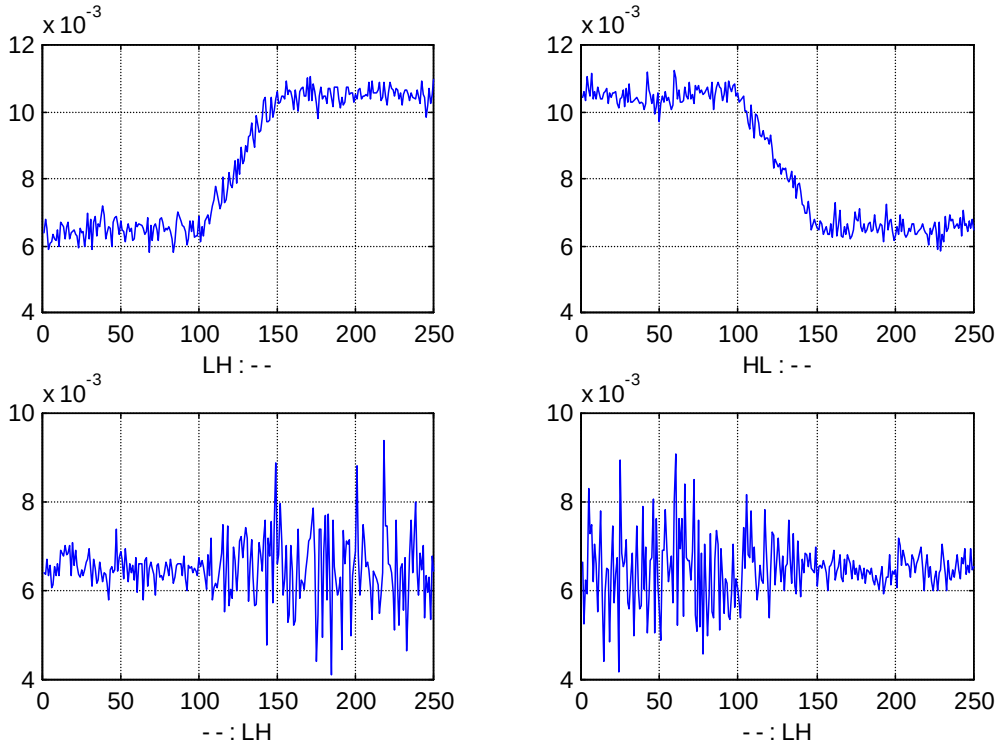

Figure 4.19: The four first change cases from table 4.1.

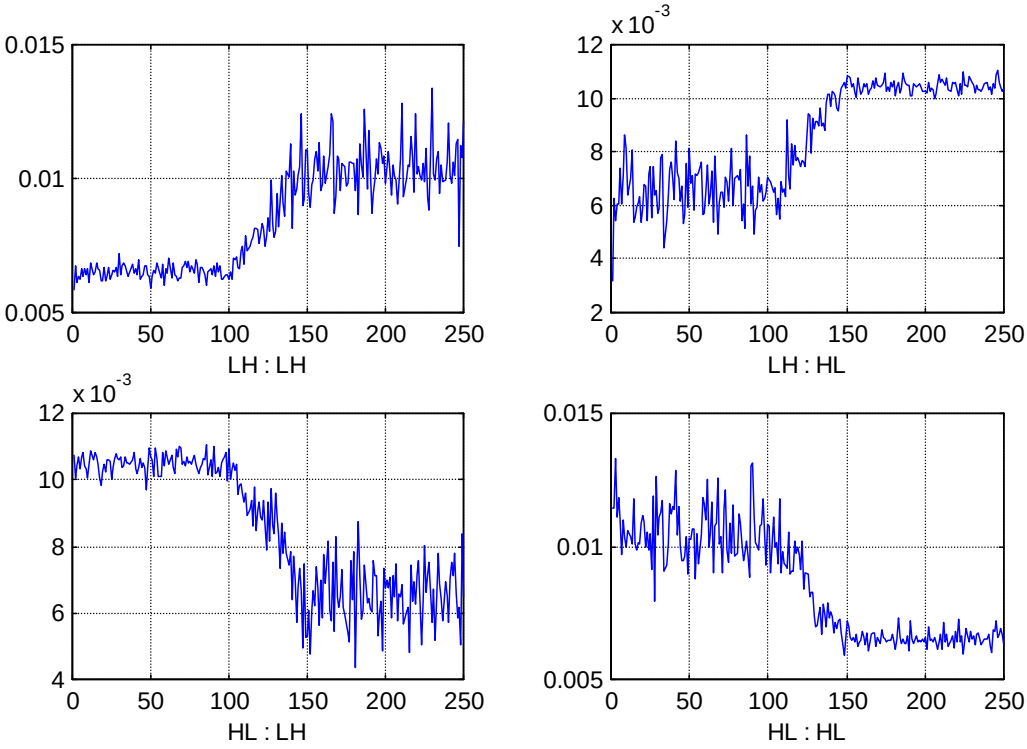

Figure 4.20: The four last change cases from table 4.1.

4.2.3 The log-likelihood ratio

Given a feature signal $y(x)$ which is Gaussian distributed and contains a change in the scalar parameter θ, the log-likelihood ratio is from [1] defined by,

$$s(y) = \ln \frac{p_{\theta_1}(y)}{p_{\theta_0}(y)}, \tag{4.6}$$

where $\theta = \theta_0$ before the change and $\theta = \theta_1$ after the change and where $p_{\theta_i}(y)$ is Gaussian distributed, which from **[1]** is defined by,

$$p_{\theta_i}(y) = \frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(y-\mu_i)^2}{2\sigma_i^2}}. \tag{4.7}$$

Figure 4.21 shows a feature signal that is Gaussian distributed and contains a change in both scalar parameters μ and θ (LH : LH). The probability density functions of these two Gaussian distributions are also plotted. The idea behind the log-likelihood ratio is, that if no change is present in the signal, the probability density function remains constant, which causes the log-likelihood ratio to be zero. On the other hand, when a change is present, the probability density function of the normal condition is small and the probability density function of the new condition is high. This causes the log-likelihood ratio to be high. In figure 4.22 this property of the log-likelihood ratio is shown.

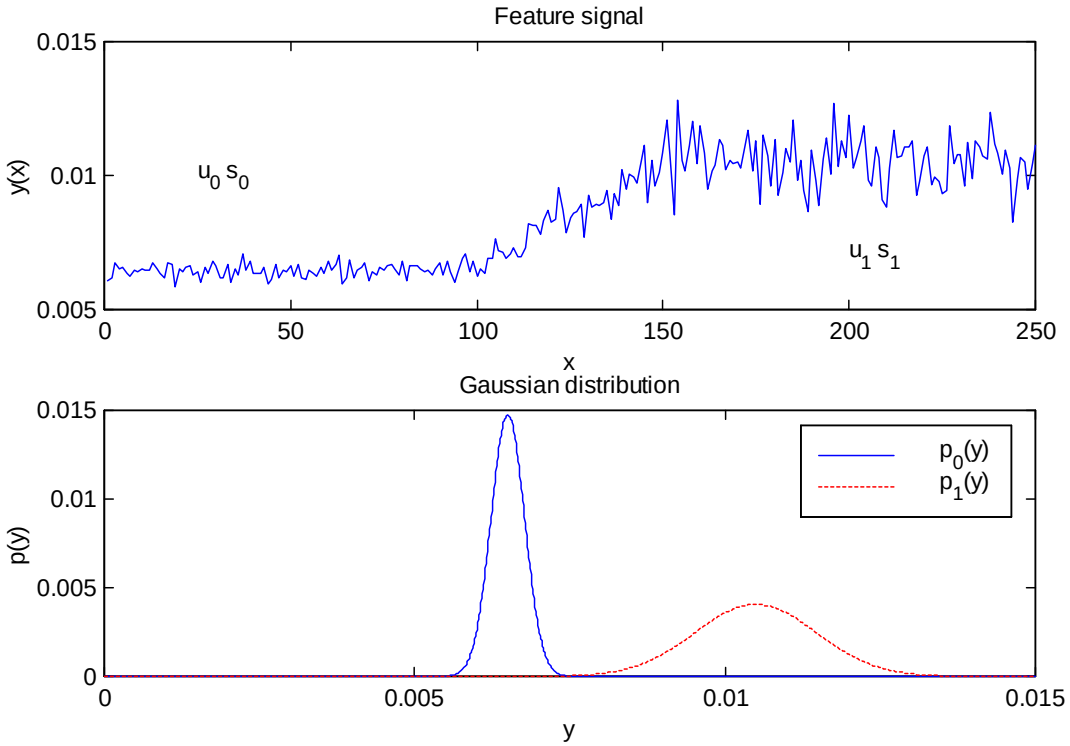

Figure 4.21: Gaussian distributed feature signal and its probability density functions.

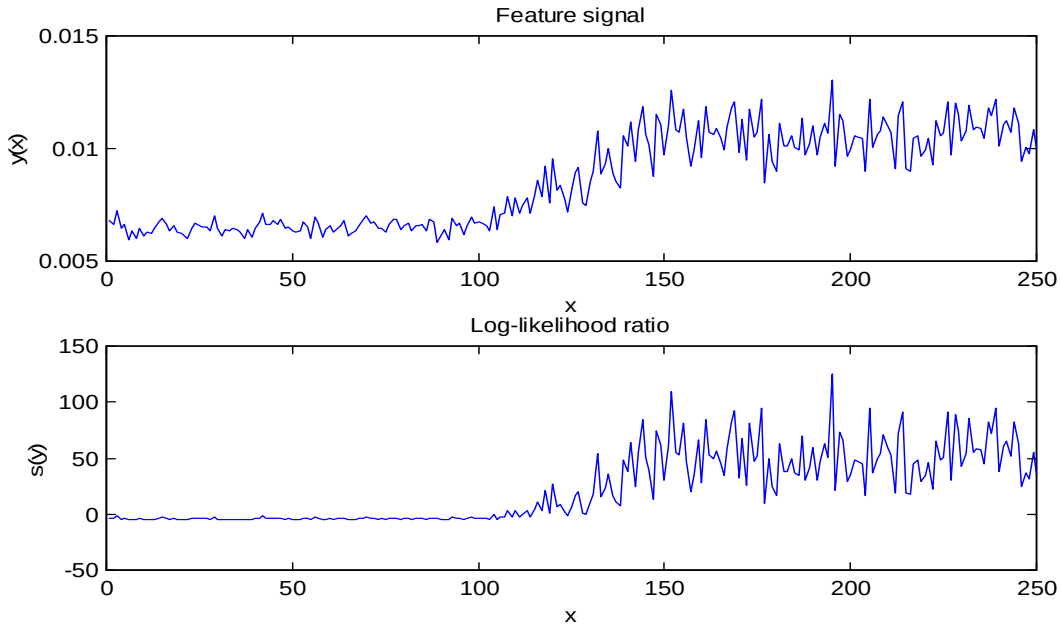

Figure 4.22: Gaussian distributed feature signal and the corresponding log-likelihood ratio. When the engine changes condition, the ratio changes from app. zero to a high value.

Now it is clear that the log-likelihood ratio is a powerful tool to detect changes in signals, but it is very important to notice, that the procedure presented in the above figure only works for off-line algorithms. One has to know both the normal condition distribution parameters and the new condition distribution parameters, and this is not the case in on-line algorithms.

In the figure above the change was in the mean value of the feature signal with a specific set of distribution parameters. What happens with the log-likelihood ratio if other distribution parameters are used? This is investigated in the proceeding sections for the three main change cases, change in the mean value, change in the standard deviation and change in both parameters.

4.2.4 Case one: Change in the mean value

In this case only a change in the mean value is considered, thus s(y) is,

$$s(y) = \ln \frac{p_{\mu_1}(y)}{p_{\mu_0}(y)}. \tag{4.8}$$

When the two distributions are assumed to be Gaussian, s(y) is,

$$s(y) = \ln \frac{\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu_1)^2}{2\sigma^2}}}{\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu_0)^2}{2\sigma^2}}}. \tag{4.9}$$

With algebra this is shown to be,

$$s(y) = \frac{\mu_1 - \mu_0}{\sigma^2}\left(y - \frac{\mu_0 - \mu_1}{2}\right). \tag{4.10}$$

Next, indices are introduced to help visualizing the behavior of y and s(y), that is the change in the mean value, and y_i is now assumed to be samples drawn from the normal distribution with mean equal to 0 and variance equal to 1,

$$s_i = \frac{\mu_1 - \mu_0}{\sigma^2}\left(y_i\sigma + \mu_i - \frac{\mu_0 - \mu_1}{2}\right). \tag{4.11}$$

To investigate s_i for various combinations of μ_0, μ_1 and σ one could just sweep all three parameters through realistic intervals. This is, however, not efficient since it is only necessary to sweep a single parameter, i.e. $\frac{\Delta\mu}{\sigma}$. This is collected in postulate 1.

Postulate 1:

If the ratio $\frac{\Delta\mu}{\sigma}$ *remains constant in the case of a change in the mean value of a Gaussian distributed signal, then the log-likelihood ratio also remains constant.*

Proof:

Given a ratio $\frac{\Delta\mu}{\sigma}$ the log-likelihood ratio is from (4.11),

$$s_i = \frac{\mu_1 - \mu_0}{\sigma^2}\left(y_i\sigma + \mu_i - \frac{\mu_0 - \mu_1}{2}\right). \tag{4.12}$$

This is the same as,

$$s_i = \frac{\Delta\mu}{\sigma\sigma}\left(y_i\sigma + \mu_i + \frac{\Delta\mu}{2}\right). \tag{4.13}$$

If $\frac{\Delta\mu}{\sigma}$ is constant then the ratio can be extended by a constant c,

$$\frac{\Delta\mu}{\sigma} = \frac{c\Delta\mu}{c\sigma}. \tag{4.14}$$

Introducing the extension in (4.13) gives,

$$s_i = \frac{c\Delta\mu}{c\sigma\, c\sigma}\left(y_i c\sigma + c\mu_i + \frac{c\Delta\mu}{2}\right), \tag{4.15}$$

since the standard deviation no longer is σ but $c\sigma$ and the mean values of condition 0 and no longer are μ_0 and μ_1 but $c\mu_0$ and $c\mu_1$. In (4.15) all c's balance each other and it is easy to conclude that (4.15) is equal to (4.12). QED

At this point a question arises: What is the range of the ratio $\frac{\Delta\mu}{\sigma}$? The ratio must be different from 0 since this corresponds to no change in the mean value. But what happens if the change is from a high to a low mean value, that is negative ratios? Will postulate 1 still be valid? The answer is yes, which can be seen from (4.16),

$$s_i = -\frac{\Delta\mu}{\sigma^2}\left(y_i\sigma + \mu_i - \frac{\Delta\mu}{2}\right). \tag{4.16}$$

From this and by applying the extension of $-\frac{\Delta\mu}{\sigma}$ by c, it can be confirmed that postulate 1 is still valid. Thus the range of the ratio $\frac{\Delta\mu}{\sigma}$ is,

$$\frac{\Delta\mu}{\sigma} \in \boldsymbol{R}/0\,. \tag{4.17}$$

4.2.5 Case two: Change in the standard deviation

When a change in the standard deviation is considered the log-likelihood ratio is defined in the same manner as with a change in the mean value,

$$s(y) = \ln\frac{p_{\sigma_1}(y)}{p_{\sigma_0}(y)}. \tag{4.18}$$

Insertion of Gaussian distributions gives,

$$s(y) = \ln\frac{\frac{1}{\sigma_1\sqrt{2\pi}}e^{-\frac{(y-\mu)^2}{2\sigma_1^2}}}{\frac{1}{\sigma_0\sqrt{2\pi}}e^{-\frac{(y-\mu)^2}{2\sigma_0^2}}}. \tag{4.19}$$

It can be shown that this is the same as,

$$s(y) = \ln\frac{\sigma_0}{\sigma_1} + \left(\frac{1}{\sigma_0^2} - \frac{1}{\sigma_1^2}\right)\frac{(y-\mu)^2}{2}. \tag{4.20}$$

Introducing indices and regarding y as samples drawn from the normal distribution gives,

$$s_i = \ln\frac{\sigma_0}{\sigma_1} + \left(\frac{1}{\sigma_0^2} - \frac{1}{\sigma_1^2}\right)\frac{(y_i\sigma_i + \mu_i - \mu)^2}{2}. \tag{4.21}$$

Again, it is postulated that it is only necessary to sweep a single parameter, i.e. the ratio between the two standard deviations $\frac{\sigma_0}{\sigma_1}$. This is summarized in postulate 2.

Postulate 2:

If the ratio $\frac{\sigma_0}{\sigma_1}$ remains constant in the case of a change in the standard deviation of a Gaussian distributed signal, then the log-likelihood ratio also remains constant.

Proof:

The ratio between the standard deviations can be extended by a constant *c*,

$$\frac{\sigma_0}{\sigma_1} = \frac{c\sigma_0}{c\sigma_1}. \tag{4.22}$$

This is inserted in (4.21) and we notice that μ_i is equal to μ, since there is no change in the mean value,

$$s_i = \ln\frac{c\sigma_0}{c\sigma_1} + \left(\frac{1}{(c\sigma_0)^2} - \frac{1}{(c\sigma_1)^2}\right)\frac{(y_i c\sigma_i)^2}{2}. \tag{4.23}$$

This is rearranged to,

$$s_i = \ln\frac{c\sigma_0}{c\sigma_1} + \frac{1}{c^2}\left(\frac{1}{\sigma_o^2} - \frac{1}{\sigma_1^2}\right)\frac{c^2(y_i\sigma_i)^2}{2} \tag{4.24}$$

From this it can be confirmed that the *c*'s balance each other and that (4.24) is equal to (4.21). Thus only a sweep in $\frac{\sigma_0}{\sigma_1}$ is necessary. QED

The range of $\frac{\sigma_0}{\sigma_1}$ can't be negative since one of the standard deviations then will be negative, and this can not be true. The ratio can not be 1 since this corresponds to no change in the standard deviation. To avoid division by zero in (4.21) neither σ_0 nor σ_1 can be zero. Thus, the range of $\frac{\sigma_0}{\sigma_1}$ is,

$$\frac{\sigma_0}{\sigma_1} \in \boldsymbol{R}_+/1\ . \tag{4.25}$$

4.2.6 Case three: Change in both mean value and standard deviation

Now the log-likelihood ratio is defined with two change parameters,

$$s(y) = \ln \frac{p_{\mu_1,\sigma_1^2}(y)}{p_{\mu_0,\sigma_0^2}(y)}. \tag{4.26}$$

Since Gaussian distributions are assumed this is the same as,

$$s(y) = \ln \frac{\frac{1}{\sigma_1\sqrt{2\pi}} e^{-\frac{(y-\mu_1)^2}{2\sigma_1^2}}}{\frac{1}{\sigma_0\sqrt{2\pi}} e^{-\frac{(y-\mu_0)^2}{2\sigma_0^2}}}. \tag{4.27}$$

Again with algebra is done, thus giving,

$$s(y) = \ln \frac{\sigma_0}{\sigma_1} + \frac{(y-\mu_0)^2}{2\sigma_0^2} - \frac{(y-\mu_1)^2}{2\sigma_1^2}. \tag{4.28}$$

When indices are introduced and *y* is samples drawn from the normal distribution we have,

$$s_i = \ln \frac{\sigma_0}{\sigma_1} + \frac{(y_i\sigma_i + \mu_i - \mu_0)^2}{2\sigma_0^2} - \frac{(y_i\sigma_i + \mu_i - \mu_1)^2}{2\sigma_1^2}. \tag{4.29}$$

Not surprisingly a single sweep parameter is sufficient to calculate the log-likelihood ratio for all legal combinations of the four parameters.

Postulate 3:

If the ratio $\frac{\Delta\mu}{\Delta\sigma}$ *remains constant in the case of a change in both the mean value and the standard deviation in a Gaussian distributed signal, then the log-likelihood ratio also remains constant.*

Proof:

The ratio $\frac{\Delta\mu}{\Delta\sigma}$ is extended by a constant *c*,

$$\frac{\Delta\mu}{\Delta\sigma} = \frac{c\Delta\mu}{c\Delta\sigma}. \tag{4.30}$$

Then the log-likelihood ratio is,

$$s_i = \ln\frac{c\sigma_0}{c\sigma_1} + \frac{(y_i c\sigma_i + c\mu_i - c\mu_0)^2}{2(c\sigma_0)^2} - \frac{(y_i c\sigma_i + c\mu_i - c\mu_1)^2}{2(c\sigma_1)^2}, \tag{4.31}$$

which is the same as,

$$s_i = \ln\frac{c\sigma_0}{c\sigma_1} + \frac{c^2(y_i\sigma_i + \mu_i - \mu_0)^2}{2c^2\sigma_0^2} - \frac{c^2(y_i\sigma_i + \mu_i - \mu_1)^2}{2c^2\sigma_1^2}. \tag{4.32}$$

Since the *c*'s balance each other (4.32) is equal to (4.29), thus Postulate 3 is valid. QED

The ratio $\frac{\Delta\mu}{\Delta\sigma}$ must not be 0 nor $\pm\infty$ since this corresponds to no change in the mean value and the standard deviation, respectively. Neither σ_0 nor σ_1 can be zero since division by zero in (4.29) must be avoided. Therefore the range of the ratio is,

$$\frac{\Delta\mu}{\Delta\sigma} \in \boldsymbol{R}/\{0, \pm\infty\}, \sigma_0 \text{ and } \sigma_1 \neq 0. \tag{4.33}$$

4.2.7 The basic principles in the off-line hypothesis test

Now the basic principles in the off-line hypothesis test can be listed:

- The off-line hypothesis test receives a window of cycles of the feature signal. In this window a change can be present or a no-change can be present.
- Then it is assumed that there is a change in the mean value of the windowed feature signal, and the corresponding ratio is calculated by means of the estimated distributions in the beginning and the end of the window. From a "critical value" it is decided whether the ratio is good or bad. If the ratio is good, then a change in the mean value has happened – the hypothesis is true. Otherwise, no change has happened.
- Then it is assumed that a change has occurred in the deviation, the corresponding ratio is calculated again, and the process continues in a similar way as before.
- Finally, a change in both the mean value and the deviation is assumed, and the process continues again as before.

If just one of the above three "sub" -hypothesis is true, then the main hypothesis will be accounted for as being true. If no sub-hypothesis is true, then the main hypothesis neither will be true.

4.3 Off-line change point estimation

The last sub-task in the segmentation task is to estimate the change points suggested by the on-line change detection algorithm more precisely when the off-line hypothesis test is a success, i.e. it is very probable that there is a change in the feature signal. In contrast to the

on-line change detection task, the literature, [1], [2] suggests several approaches, since the issue has changed from *unknown* detection to *known* detection. Some of these approaches are:

- Shewhart algorithm.
- Geometric Moving Average (GMA) algorithm.
- Finite Moving Average (FMA) algorithm.
- Filtered Derivative (FD) algorithm.
- CUSUM algorithm.

All algorithms are described in Basseville et al. [1], and have been implemented and investigated in the beginning of the project. Especially the Shewhart algorithm has been applied in this work, because it is the simplest algorithm and one of the first change detection algorithms in history. It uses the log-likelihood ratio described in the off-line hypothesis test section, to determine the change point.

However, non of the algorithms are used in the final automatic change detection system, since they all share a single major disadvantage, which is that a threshold must be determined by a technician. This is not preferable, since the technicians then must gain even more knowledge about the different changes and their responses in the feature signals. The basic idea behind *unknown* automatic condition monitoring in this work is that standardization must be used as the point of development. Thus, the fewer decisions made by technicians, the better *unknown* automatic condition monitoring system.

4.3.1 Maximum likelihood

Fortunately, a classical and well-known solution exists to the problem of estimating the change points off-line without using a threshold. The solution is called *maximum likelihood* and is described in both [1] and [2]. The principle in the maximum likelihood approach is to create a *likelihood* function, which is a function of certain parameters and the data. The optimal parameters are then the parameters, where the likelihood function is maximal.

In this thesis, the parameters are the two change points cp_1 and cp_2. The first change point corresponds to the cycle where the engine *leaves* its normal condition, and the second change point corresponds to where the engine enters the *new* condition. Now the likelihood function L can be created as,

$$L(cp_1, cp_2, y) = \prod_{i=1}^{cp_1} p_0(y_i) \prod_{i=cp_1+1=cp_2}^{N} p_1(y_i). \tag{4.34}$$

Here y is the windowed feature signal with length N selected by the on-line change detection algorithm and where the off-line hypothesis test has confirmed that a change is present. It is assumed that the change is abrupt, i.e. the change is over a single, or very few cycles. p_0 and p_1 are the probability density functions of y in the normal condition and the new condition, respectively. The probability density functions are in the interval [0; 1]. Thus L is also in this interval. Usually, the majority of the values in L are very small, and therefore, it is normal to apply the negative logarithm of the likelihood function,

$$L(cp_1, cp_2, y) = -\log\left(\prod_{i=1}^{cp_1} p_0(y_i) \prod_{i=cp_1+1}^{cp_2} p_1(y_i)\right). \quad (4.35)$$

4.3.2 Maximum likelihood implementation

The log-likelihood function in (4.35) assumes that the drift from the normal condition to the new condition is abrupt. This is very inappropriate to assume, since the feature signals generated in chapter 3 consist of both abrupt and non-abrupt changes. If the assumption is not changed, the off-line change point estimation task only has a chance of working adequately in the case of abrupt changes. Therefore, a more realistic situation is assumed, which is,

Assumption

The feature signal is assumed to consist of three parts, a normal condition, a new condition and a linear drift from the normal condition to the new condition.

The two "steady" conditions, the normal and the new, are assumed to be Gaussian distributed with a mean value and a standard deviation. The linear drift is defined to be a condition where the *beginning* of the condition has the same mean value and deviation as the *normal* condition, and the *end* of the condition has the same mean value and deviation as the *new* condition. The increase in the mean value and the deviation follows a straight line between the beginning and end parameters. An example of such a feature signal is given in figure 4.23.

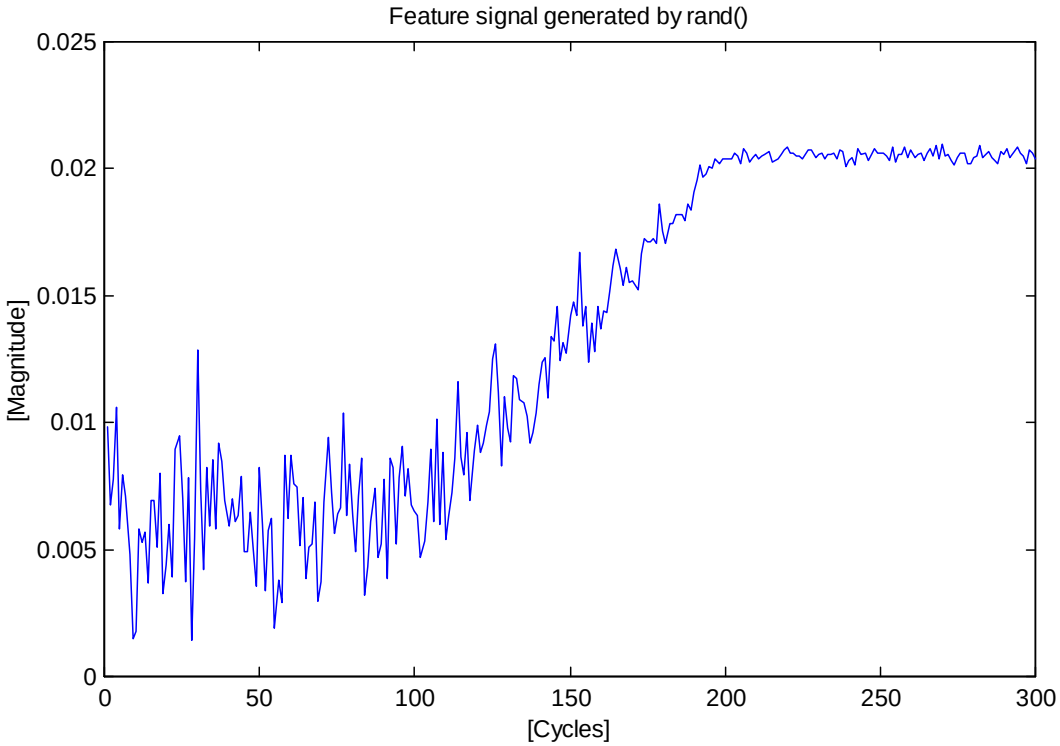

Figure 4.23: The feature signal is assumed to consist of a normal condition (100 cycles), a new condition (100 cycles), and a linear drift (100 cycles) between the conditions.

The log-likelihood function is in this case,

$$L(cp_1, cp_2, y) = -\log\left(\prod_{i=1}^{cp_1} p_0(y_i) \prod_{i=cp_1+1}^{cp_2-1} p_{0\to 1}(y_i) \prod_{i=cp_2}^{N} p_1(y_i)\right). \tag{4.36}$$

This is the first maximum likelihood method applied in this work. When it was tested on the feature signals in chapter 5 it showed good performance in some cases, but unfortunately not in all. It also suffers from the assumptions that the normal condition and the new condition are steady, and that the drift is linear – this is not an adequate assumption. One solution to overcome the linear drift assumption was implemented. This is referred to as the second maximum likelihood method.

In the second method it is still assumed that the normal and the new condition are "steady", but the drift between the conditions is ignored. Then the two change points are estimated separately by modifying the log-likelihood function so it regards the feature signal to consist a normal condition and "the rest", i.e. something that does not belong to the normal condition. In other words we search for the point where the engine *leaves* its normal condition. Then the log-likelihood function is,

$$L(cp_1, y) = -\log\left(\prod_{i=1}^{cp_1} p_0(y_i) \prod_{i=cp_1+1}^{N} (1 - p_o(y_i))\right). \tag{4.37}$$

When the first change point has been estimated, the process is repeated, but is now started from the end of the feature signal and goes backwards in order to estimate the second change point. In fact it is far easier to flip the feature signal in Matlab and then use the same code as was used for estimating cp_1. This is also done in the off-line change point estimation code.

Both maximum likelihood methods are tested in the test section. It is worth mentioning that the result is that non of the methods works good enough in all cases. A good outcome is, however, that the second method got rid of some of the problems with the first method.

4.4 Panel of experts

In this thesis, four sensor signals were provided, each providing at least two feature signals[5] to the change detection system. The idea with this work is to apply the automatic change detection system on each of the feature signals in order to detect and estimate the changes in the feature signals. Since an AE is present in only a short space around its generation point, and since the AE sensors are placed at different positions on the engine, it is very likely that some types of engine condition changes can not be observed in all feature signals at the same time. Thus a situation can rise where one detection system applied on one feature signal will set an alarm and other detection systems applied on other feature signals wont set an alarm. The question is then: "Has the engine changed condition or not?"

This is a very interesting problem, which is referred as *the panel of experts*. To explain the name, one can regard the automatic change detection systems as experts who look at some

[5] If the mean value and deviation approach is used.

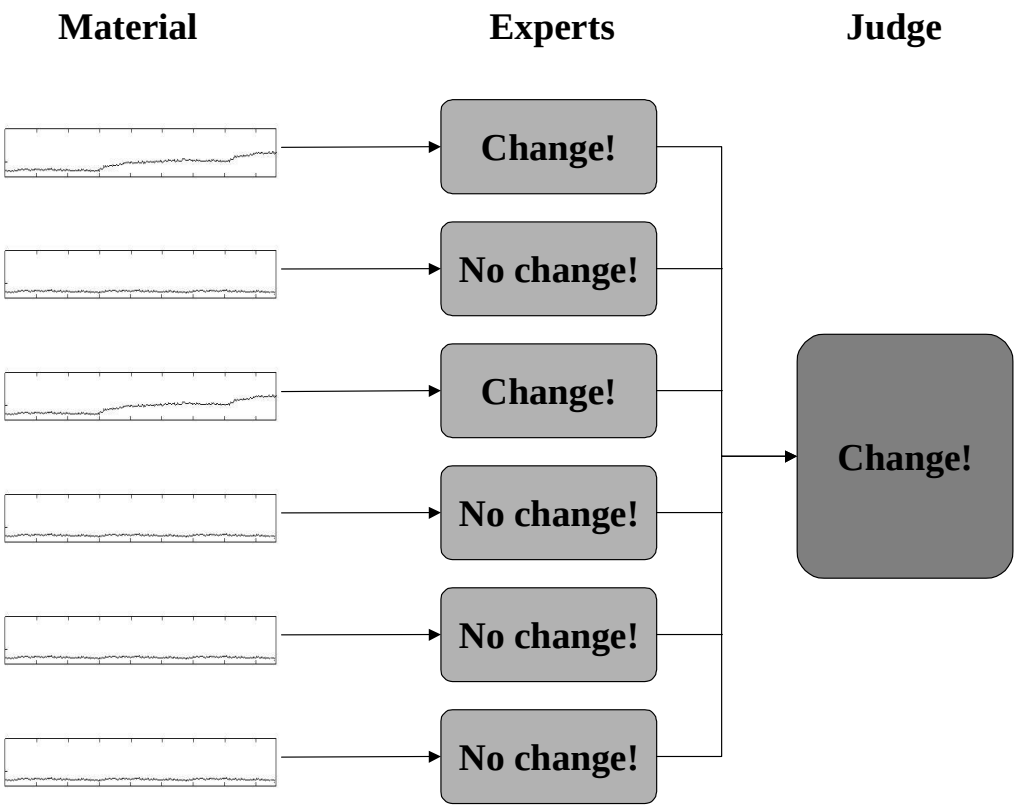

Figure 4.24: Panel of experts. The feature signals help the experts to provide statements. A "judge" combines the multiple statements down to a single statement.

kind of material (the feature signals) and from this material they provide statements like: "The engine changed condition at this particular cycle", or "The engine is still in the normal condition", etc. Then the problem is how to combine these multiple statements down to a *single* statement on the engine condition.

This has not been investigated in this work. However, at this point it seems reasonable to choose that if just a single expert states that a change has occurred, then this is true. The reason is the before mentioned damping property with AE's. This will of course demand the change detection system to be very reliable. Research has shown that implementation of a *majority voter system* [9], [18], which is a type of expert panel, in fact improves the system performance. In figure 4.24 a proposal on a panel of experts is given.