Aircraft engine health monitoring using Self-Organizing Maps

Etienne Côme, Marie Cottrell, Michel Verleysen, Jérôme Lacaille

Aircraft engine health monitoring using Self-Organizing Maps

Etienne Côme[1], Marie Cottrell[1], Michel Verleysen[2], and Jérôme Lacaille[3]

[1] SAMM - Universit Paris 1 Panthon-Sorbonne
90, rue de Tolbiac, 75013 Paris, France
{etienne.come,marie.cottrell}@univ-paris1.fr
[2] Université Catholique de Louvain, Machine Learning Group
Place du levant 3, 1348 Louvain-La-Neuve, Belgium
michel.verleysen@uclouvain.be
[3] Snecma, Rond-Point Ren Ravaud-Rau,
77550 Moissy-Cramayel CEDEX, France
jerome.lacaille@snecma.fr

Abstract. Aircraft engines are designed to be used during several tens of years. Ensuring a proper operation of engines over their lifetime is therefore an important and difficult task. The maintenance can be improved if efficients procedures for the understanding of data flows produced by sensors for monitoring purposes are implemented. This paper details such a procedure aiming at visualizing in a meaningful way successive data measured on aircraft engines. The core of the procedure is based on Self-Organizing Maps (SOM) which are used to visualize the evolution of the data measured on the engines. Rough measurements can not be directly used as inputs, because they are influenced by external conditions. A preprocessing procedure is set up to extract meaningful information and remove uninteresting variations due to change of environmental conditions. The proposed procedure contains three main modules to tackle these difficulties: environmental conditions normalization (ECN), change detection and adaptive signal modeling (CD) and finally visualization with Self-Organizing Maps (SOM). The architecture of the procedure and of modules are described in details in this paper and results on real data are also supplied.

Key words: Health monitoring, Self-Organizing Maps, Changes detection

1 Introduction

During the flights, some on-board sensors measure many parameters related to the behavior (and therefore the health) of aircraft engines. These parameters are recorded and used at short and long terms for immediate action and alarm generation, respectively. In this work, we are interested in the long-term monitoring of aircraft engines and we want to use these measurements to detect any deviations from a "normal" behavior, to anticipate possible faults and to facilitate the maintenance of aircraft engines.

This work presents a tool that can help experts, in addition to their traditional tools based on quantitative inspection of some relevant variables, to easily visualize the evolution of the engine health. This evolution will be characterized by a trajectory on a two-dimensional Self-Organizing Map. Abnormal aging and fault will result in deviations with respect to normal conditions.

The choice of Self-Organizing Maps is motivated by several points:

– SOMs are useful tools for visualizing high-dimensional data onto a low-dimensional grid;
– SOMs have already been applied with success for fault detection and prediction in plants and machines (see [1] for example).

The article is organized as follow. First, in Section 2, the data and the notations used throughout the paper are presented. The methodology and the global architecture of the proposed procedure are described in Section 3. Each step is defined and results on real data are given in Section 4. Finally, in Section 5, perspectives on trajectory analysis are supplied.

2 Data

Measurements are collected on a set of I engines. On each engine $i \in \{1, \ldots, I\}$, n_i measurements are performed successively flight after flight; there is thus no guarantee that the time intervals between two measures are approximately equal. Each observation is denoted by Z_{ij}, where $i \in \{1, \ldots, I\}$ is the engine number and $j \in \{1, \ldots, n_i\}$ is the flight number.

Each vector Z_{ij} contains two kinds of variables: those which are strictly related to the behavior of the engine (fuel consumption, static pressure, ...), and those which are related to the environment (temperature, altitude, ...). Let the p engine variables be denoted by $Y_{ij}^1, \ldots, Y_{ij}^p$ and the q environmental variables by $X_{ij}^1, \ldots, X_{ij}^q$. Each observation is therefore a $(p+q)$-vector Z_{ij}, where $Z_{ij} = [Y_{ij}, X_{ij}] = \left[Y_{ij}^1, \ldots, Y_{ij}^p, X_{ij}^1, \ldots, X_{ij}^q\right]$. The variables at disposal are listed in Table 1. There are $p = 5$ engine variables and $q = 15$ environmental variables. The dataset contains measurements for approximately one year of flights and $I = 91$ engines, that leads to a global dataset with $\sum_{i=1}^{91} n_i = 59407$ $(p+q)$-dimensional observations.

3 Methodology

The goal is to build the trajectories of all the engines, that is to project the successive observations of each engine on a Self-Organizing Map, in order to follow the evolution and to eventually detect some “abnormal” deviation.

It is not valuable to use the rough engine measurements: they are inappropriate for direct analysis by Self-Organizing Maps, because they are strongly dependent on environment conditions and also on the characteristics of the engine (its past, its age, ...).

	Name	**Description**	**Type**	**Binary**
	aid	aircraft id		
	eid	engine id		
	fdt	flight date		
X_{ij}^1	temp	temperature	environment	
X_{ij}^2	nacelletemp	nacelle temperature	environment	
X_{ij}^3	altitude	aircraft altitude	environment	
X_{ij}^4	wingaice	wings anti-ice	environment	✓
X_{ij}^5	nacelleaice	nacelle anti-ice	environment	✓
X_{ij}^6	bleedvalve	bleed valve position	environment	✓
X_{ij}^7	isolationleft	valve position	environment	✓
X_{ij}^8	vbv	variable bleed valve position	environment	
X_{ij}^9	vsv	variable stator valve position	environment	
X_{ij}^{10}	hptclear	high pressure turbine setpoint	environment	
X_{ij}^{11}	lptclear	low pressure turbine setpoint	environment	
X_{ij}^{12}	rotorclear	rotor setpoint	environment	
X_{ij}^{13}	ecs	air cooling system	environment	
X_{ij}^{14}	fanspeedi	N1	environment	
X_{ij}^{15}	mach	aircraft speed	environment	
Y_{ij}^1	corespeed	N2	engine	
Y_{ij}^2	fuelflow	fuel consumption	engine	
Y_{ij}^3	ps3	static pressure	engine	
Y_{ij}^4	t3	temperature plan 3	engine	
Y_{ij}^5	egt	exhaust gas temperature	engine	

Table 1. Variables names, descriptions and type.

The first idea is to use a linear regression for each engine variable: the environmental variables (real-valued variables) and the number of the engine (categorical variable) are the predictors and the residuals of these regressions can be used as standardized variables (see [2] for details). For each engine variable $r = 1, \ldots, p$, the regression model can be written as:

$$Y_{ij}^r = \mu^r + \alpha_i^r + \lambda_1^r X_{ij}^1 + \ldots + \lambda_q^r X_{ij}^q + \epsilon_{ij}^r \tag{1}$$

where α_i^r is the engine effect on the r^{th} variable, $\lambda_1^r, \ldots, \lambda_q^r$ are the regression coefficients for the r^{th} variable, μ^r is the intercept and the error term ϵ_{ij}^r is the residual.

Figure 1 presents for example the rough measurements of the *corespeed* feature as a function of time (for engine 6) and the residuals computed by model (1). The rough measurements seem almost time independent on this figure, whereas the residuals exhibit an abrupt change which is linked to a specific event in the life of this engine. This simple model is therefore sufficient to bring to light interesting aspects of the evolution of this engine. However, the signals may contain ruptures, making the use of a single regression model hazardous.

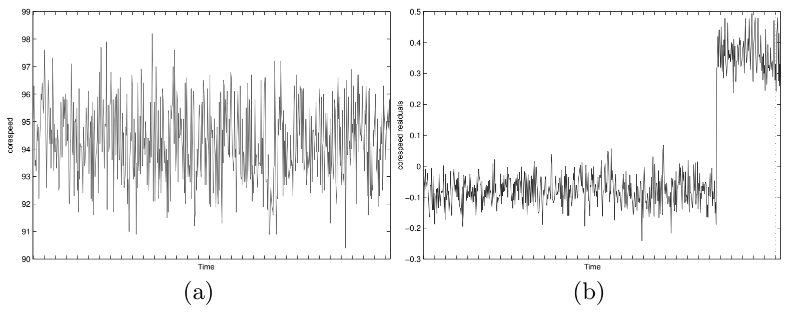


Fig. 1. (a) Rough measurements of the **corespeed** variable as a function of time for engine 6, (b) residuals of the same variable and for the same engine using a simple linear model with the environmental variables and the engine indicator as predictors (see Table 1).

The main idea of this work is to replace model (1) by a new procedure which deals with the temporal behavior of the signals. The goal is therefore to detect the ruptures and to use different models after each rupture.

This new procedure is composed of two modules. The first module (Environmental Conditions Normalization, ECN) aims at removing the effects of the environmental variables to provide standardized variables, independent of the flight conditions. It is described in section 4.1

The second module uses an on-line change detection algorithm to find the above mentioned abrupt changes, and introduces a piecewise regression model. The detection of the change points is done in a multi-dimensional setting taking as input all the normalized engine variables supplied by the ECN module. The Change Detection (CD) module is presented in Section 4.2.

Finally, as a result of these first two steps, the “cleaned” database can be used as input to a Self-Organizing Map with a “proper” distance for trajectories visualization. The third module (SOM) provides the “map” on which the trajectories will be drawn.

This three-steps procedure is summarized in Figure 2.

4 Description of the three modules

4.1 Environmental Conditions Normalization - ECN

The first module aims at removing the effects of the environmental variables.

For that purpose, one regression model has to be fitted for each of the p engine variables. As the relationship between environmental and engine variables is complex and definitively not linear, the environmental variables can be supplemented by some non-linear transformations of the latter, increasing the number of explanatory variables. Interactions (all the possible products between two environmental variables), squares, cubes and fourth powers of the non binary

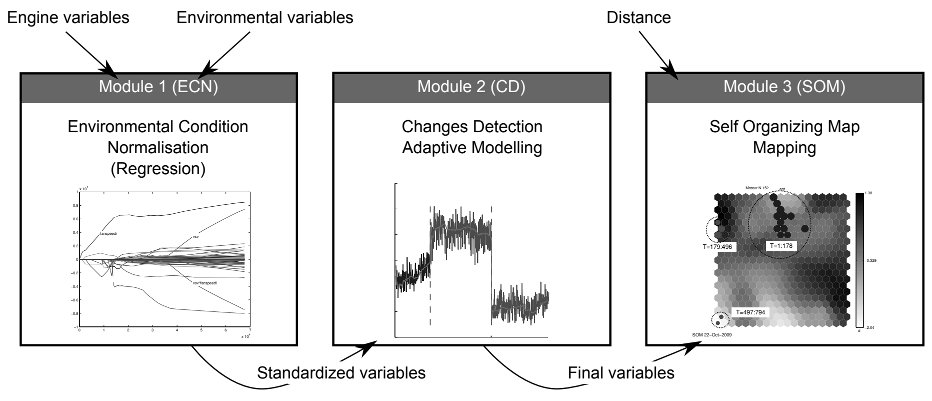


Fig. 2. Global architecture of the health monitoring tools.

environmental variables are considered. The number q of predictors in the model is therefore a priori equal to $(11 + 4) * (11 + 4 - 1)/2 = 105$ for the interactions variables and $11 * 4 + 4 = 48$ for the power of the continuous variable and the binary variables leading to a total of $q = 153$ predictors.

This number is certainly too large and some of them are clearly irrelevant due to the systematic procedure used to build the non-linear transforms of environmental variables.

A LASSO criterion [3] is therefore used to estimate the regression parameters and to select a subset of significant predictors. This criterion can be written using the notations from Section 2 for one engine variable Y^r, $r \in \{1, \dots, p\}$ as :

$$\boldsymbol{\beta}^r = \arg \min_{\boldsymbol{\beta}^r \in \mathbb{R}^q} \sum_{i,j=1}^{I,n_i} \left(Y_{ij}^r - \sum_{l=1}^{q} \beta_l^r \, X_{ij}^l \right)^2 , \sum_{l=1}^{q} |\beta_l^r| < C^r \qquad (2)$$

The regression coefficients are penalized by a L_1 penalty which forces some of them to be null for a well chosen value of C^r. The LARS algorithm [3] is used to estimate all the solutions of the optimization problem (2) for all possible values of C^r. The optimal value of C^r with respect to the prediction error estimated by cross-validation (with 20 blocs) is finally selected. The number of selected predictors and the coefficient of determination R^2 are listed in Table 2 for all engine variables. Engine variables are well explained by the proposed models as attested by the high value of the coefficients of determination.

	corespeed	*fuelflow*	*ps3*	*t3*	*egt*
nb vars	25	43	31	30	41
R^2_{obs}	0.9875	0.9881	0.9773	0.9636	0.8755

Table 2. Number of selected predictors and coefficients of determination for all engine variables.

A qualitative inspection of the model results was also carried out with the help of engine experts. The regularization path plot (as shown in Figure 3) is very interesting from the point of view of the experts, because it can be compared with their previous knowledge. Such a curve clearly highlights which are the more relevant predictors and they appear to be in very good adequateness with the physical knowledge on the system.

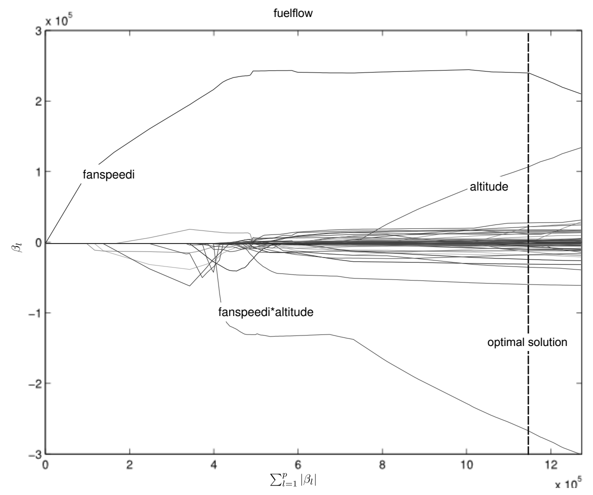


Fig. 3. Regularization path for the *fuelflow* variable: regression coefficients evolution with respect to C^r. The more significant explanatory variables are given and the best solution with respect to cross-validation is depicted by a vertical line.

In summary, the first preprocessing module (ECN) provides 5 standardized engine variables, which are independent of environmental conditions. These new variables still contain some significant aspects such as linear trends and abrupt changes at specific dates. We therefore propose to use an on-line Change Detection algorithm (CD) together with an adaptive linear model to fit the data.

4.2 Change Detection - CD

To take into account the two types of variation (linear trend and abrupt changes), we implement an algorithm based on the ideas from [4] and [5]. The solution is based on the joint use of an on-line change detection algorithm to detect abrupt changes and of a bank of recursive least squares (RLS) algorithms to estimate the slow variations of the signals. The algorithm works on-line in order to allows projecting new measurements on the map as soon as new data are available.

The method can be described as follows :

1) One RLS algorithm is used for each one of the p engine variable to recursively fit a linear model. At each date l, for each standardized engine variable r, for each engine i, one has to minimize the following criterion :

$$(\alpha_l^r, \beta_l^r) = \arg \min_{\alpha\in\mathbb{R},\beta\in\mathbb{R}} J(\{Y_{i1}^r, \dots Y_{il}^r\}, \alpha, \beta) \tag{3}$$

$$= \arg \min_{\alpha\in\mathbb{R},\beta\in\mathbb{R}} \sum_{j=1}^{l} \lambda^{(l-i)} (Y_{ij}^r - (\beta.j + \alpha))^2 \tag{4}$$

The estimates α_l^r and β_l^r are respectively the intercept and the slope of the linear relationship. These estimates are then used to remove the slow variations of the signals by defining the quantity:

$$\varepsilon_l^r = Y_{il}^r - (\beta_l^r.l + \alpha_l^r) \tag{5}$$

2) These values are then computed for each standardized engine variable and concatenated in a vector $\boldsymbol{\varepsilon}_l = [\varepsilon_l^1, \dots, \varepsilon_l^p]$, which is then used in a multi-dimensional Generalized Likelihood Ratio (GLR) algorithm [6] to detect the abrupt changes of the signals. The GLR algorithm is a sequential test procedure based on the following model :

$$\boldsymbol{\varepsilon}_k \sim \mathcal{N}_p(\theta(k), \Sigma), \forall k > 0,$$

with :

$$\theta(k) = \begin{cases} \theta \in \Theta_0 & \text{si } k < t_0, \\ \theta \in \Theta_1 & \text{si } k \geq t_0, \end{cases} \tag{6}$$

where t_0 is the unknown change time and Θ_0 and Θ_1 are two non-overlapping subsets. They are defined by two hyper-spheres centered on θ_0 as shown in Figure 4. In such a case, Θ_0 and Θ_1 are defined by:

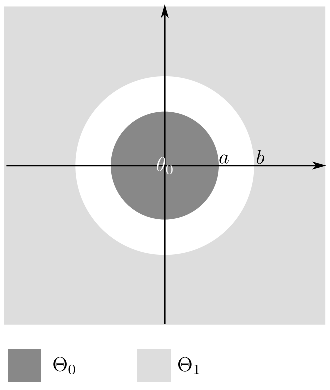


Fig. 4. Subsets for θ_1 and θ_0

$$\Theta_0 = \left\{\theta : (\theta - \theta_0)^t \Sigma^{-1} (\theta - \theta_0) \leq a^2\right\} \tag{7}$$

$$\Theta_1 = \left\{\theta : (\theta - \theta_0)^t \Sigma^{-1} (\theta - \theta_0) \geq b^2\right\}, \tag{8}$$

with $a \leq b$. The sequential test problem is then solved by defining the alarm date t_a to be the first time where the test statistic g_k is above a specified threshold h :

$$t_a = \min\{k \geq 1 : g_k \geq h\}, \tag{9}$$

with the following definition of the test statistic :

$$g_k = \max_{1 \leq j \leq k} \ln \frac{\sup_{\theta:||\theta-\theta_0||_\Sigma \geq b} \prod_{i=j}^{k} p_\theta(\varepsilon_i)}{\sup_{\theta:||\theta-\theta_0||_\Sigma \leq a} \prod_{i=j}^{k} p_\theta(\varepsilon_i)}. \tag{10}$$

where $p_\theta(.)$ denotes a density of a multivariate Gaussian of mean θ and variance Σ.

With S_j^k given by:

$$S_j^k = \ln \frac{\sup_{\theta:||\theta-\theta_0||_\Sigma \geq b} \prod_{i=j}^{k} p_\theta(\varepsilon_i)}{\sup_{\theta:||\theta-\theta_0||_\Sigma \leq a} \prod_{i=j}^{k} p_\theta(\varepsilon_i)}, \tag{11}$$

the maximization problem (10) has an analytical solution in the Gaussian case and S_j^k takes the following value :

$$S_j^k = \begin{cases} -\frac{(k-j+1)}{2}(\chi_j^k - b)^2, \text{ if } \chi_j^k < a, \\ \frac{(k-j+1)}{2}\left(-(\chi_j^k - b)^2 + (\chi_j^k - a)^2\right), \text{ if } a \leq \chi_j^k \leq b, \\ +\frac{(k-j+1)}{2}(\chi_j^k - a)^2, \text{ if } \chi_j^k \geq b, \end{cases} \tag{12}$$

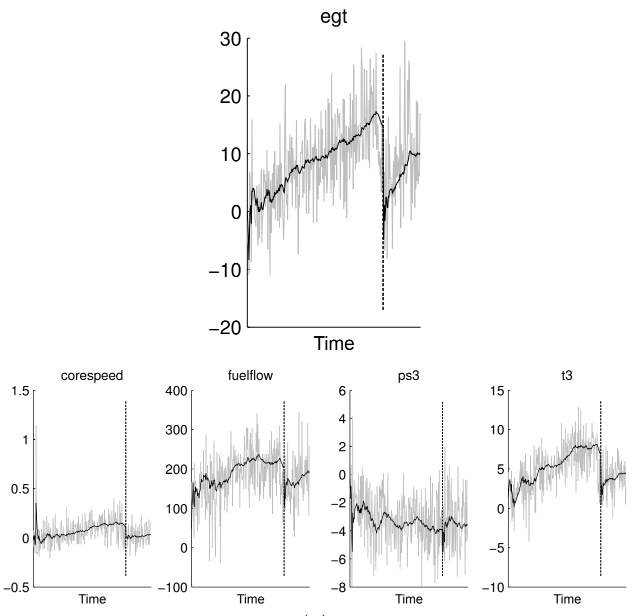
with $\chi_j^k = \sqrt{(\bar{\varepsilon}_j^k - \theta_0)^t \Sigma^{-1} (\bar{\varepsilon}_j^k - \theta_0)}$ and $\bar{\varepsilon}_j^k = \sum_{i=j}^{k} \frac{1}{(k-j)} \varepsilon_i$.

3) Finally, when an alarm is sent by the GLR algorithm, all the RLS algorithms are re-initialized.

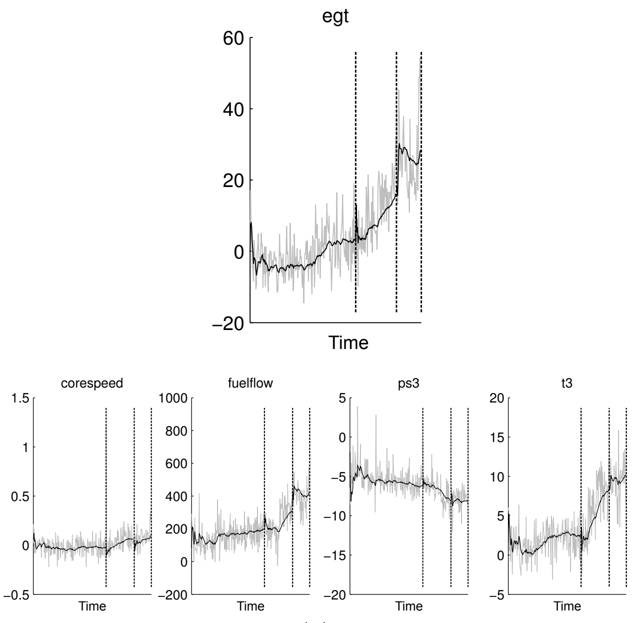
The results supplied by this algorithm are the following :

- the alarm dates supplied by the multi-dimensional GLR algorithm;
- cleaned signals estimated by the RLS algorithm;
- slopes and intercepts estimated by the RLS algorithm.

Figure 5 presents the obtained results for two engines. One abrupt change was found for the first engine and 3 for the second; all of them seem to be reasonable and a comparison between estimated alarm dates and recorded real events of the engine life have confirmed this fact. The estimated signals are also shown on these two figures.



(a)



(b)

Fig. 5. Change detection results for engines 2 and 41. Alarms are depicted by vertical lines, input signals are shown in light gray and signal estimates using RLS are depicted by a black line. One variable *egt* is bigger than the other to present more clearly the RLS estimate of the signal.

4.3 Self-Organizing-Maps - SOM

The cleaned signals provided by the previous two modules are then used as input to a SOM for visualization purpose. A [20×20] square SOM is defined to project the observations. The Matlab toolbox [7] was used to implement it and the distance was carefully chosen since the standardized engine variables are very correlated as shown by the correlation matrix in Figure 6: several correlation coefficients have an absolute value greater than 0.6. A Mahalanobis distance is therefore used to whiten the data. The observations are normalized and a classical learning scheme is used to train the map.

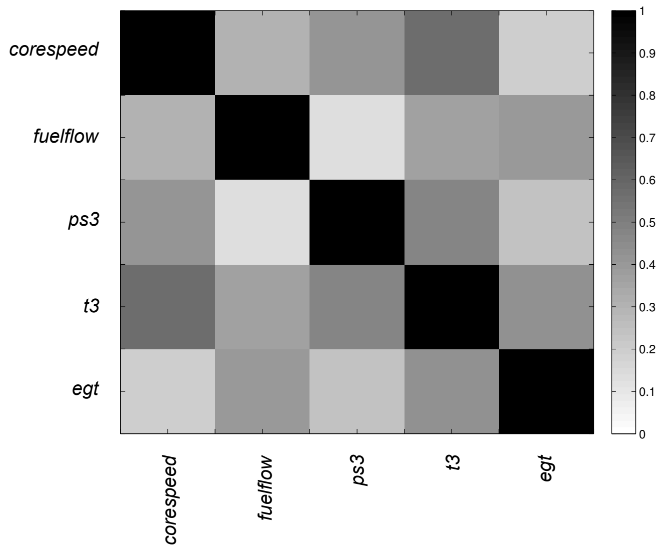


Fig. 6. Correlations between input variables

Figure 7 shows the map colored according to the values of the five engine variables. It is clearly visible that the organization of the map is successful (all variables are smoothly varying on the map).

Figure 8 presents two examples of engine trajectories on the map, which clearly have different shapes. For the first engine, available maintenance reports inform us that this engine suffers from an deterioration of its high pressure core. This fault is visible on the map at the end of the trajectory: the engine which was projected on the middle north of the map during a large part of its trajectory, suddenly moves towards the nord-east corner of the map. This area of the map furthermore correspond to anomalous values of the engine variables as shown by the component plane representation of the map (see Figure 7). The second engine which is affected by another fault has also a trajectory which is interesting even if the interpretation is less obvious. It moves to an area characterized by a

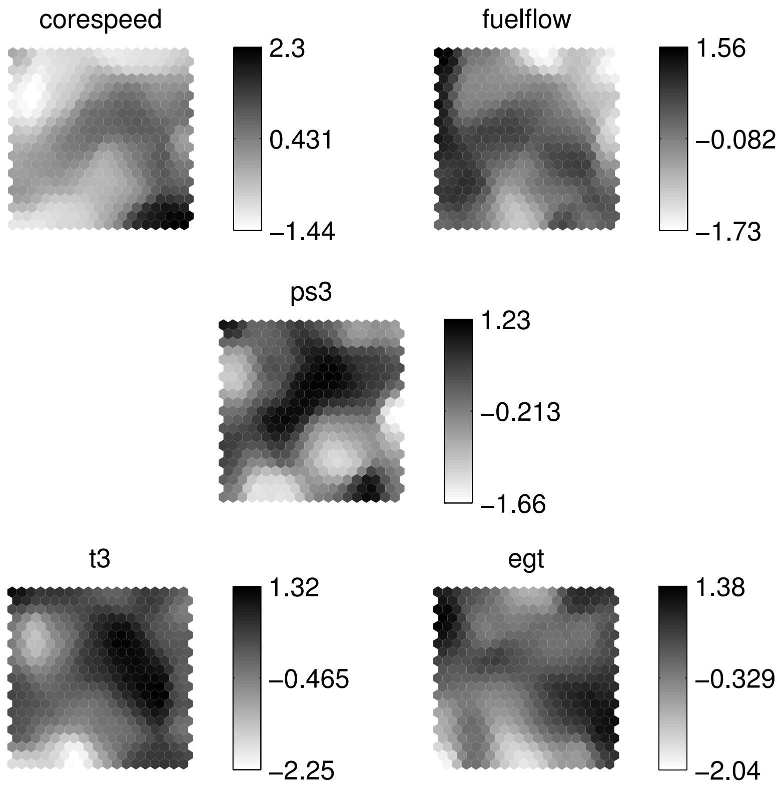
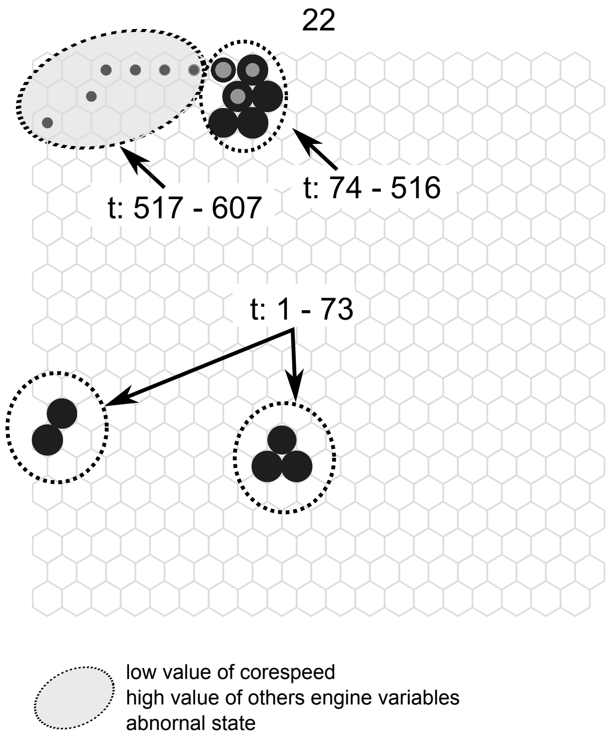


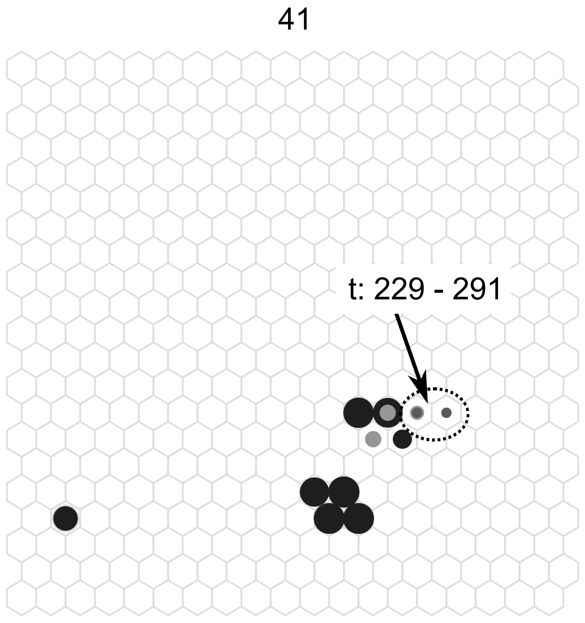
Fig. 7. Visualization of engine variable as map background; each cell of the map is colored according to the value of the selected variable.

high value of the *exhaust gas temperature* which is known to be an indicator of possible trouble.

The visual representations on the Kohonen map provide a synthetic and meaningful representation for the temporal evolution of the engines. The next step is then to characterize the different shapes of trajectories, to define a suitable distance measure between these trajectories, and to define typical behaviors related to typical faults. Further work will consist in defining a proper distance between two trajectories (or parts of trajectories) in order to propose a request-type access to the database. Taking a piece of trajectory as input, the system will finally be able to recover the most similar trajectories of the database. Such similar trajectories can then be used to predict the possible evolution of the monitored engine. *Edit-type* distance widely used in biostatistics could be of interest for this task.



(a)



(b)

Fig. 8. Trajectories of engines 22 (a) and 41 (b) on the map. The sizes of the dots are proportional to the measurement date : smallest dots correspond to recent measurements, larger dots to older measurements. The colors correspond to the segments found by the change detection algorithm (t_a = 394 and 522 for engine 22; t_a = 180, 249 and 291 for engine 41).

5 Conclusion and perspectives

The method proposed in this paper is a useful tool to summarize and represent the temporal evolution of an aircraft engine health flight after flight. The regression approach taken to deal with the problem of environmental condition normalization seem to be effective. The joint use of an adaptive algorithm to estimate signal evolution (RLS) and of a change points detection method (GLR) is also an interesting solution to deal with the non-stationarity of the signals. Finally, Self-Organizing Maps can be used to show the engine health evolution in a synthetic manner.

References

1. Svensson, M., Byttner, S., Rgnvaldsson, T.: Self-organizing maps for automatic fault detection in a vehicle cooling system. In: 4th International IEEE Conference on Intelligent Systems. Volume 3. (2008) 8–12
2. Cottrell, M., Gaubert, G., Eloy, C., Franois, D., Hallaux, G., Lacaille, J., Verleysen, M.: Fault prediction in aircraft engines using self-organizing maps. In: Advances in Self-Organizing Maps. Volume 5629/2009., Springer 37–44
3. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.J.: Least angle regression. Annals of Statistics **32**(2) (2004) 407–499
4. Gustafsson, F.: Adaptative filtering and change detetction. John Wiley & Sons (2000)
5. Ross, G., Tasoulis, D., Adams, N.: Online annotation and prediction for regime switching data streams. In: Proceedings of ACM Symposium on Applied Computing. (March 2009) 1501–1505
6. Basseville, M., Nikiforov, I.: Detection of Abrupt Changes: Theory and Application. Prentice-Hall (1993)
7. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Som toolbox for matlab 5. Technical Report A57, Helsinki University of Technology (april 2000)