International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:9, No:9, 2015

Specialized Reduced Models of Dynamic Flows in 2-Stroke Engines

S. Cagin, X. Fischer, E. Delacourt, N. Bourabaa, C. Morin, D. Coutellier, B. Carré, S. Loumé

Abstract—The complexity of scavenging by ports and its impact on engine efficiency create the need to understand and to model it as realistically as possible. However, there are few empirical scavenging models and these are highly specialized. In a design optimization process, they appear very restricted and their field of use is limited. This paper presents a comparison of two methods to establish and reduce a model of the scavenging process in 2-stroke diesel engines. To solve the lack of scavenging models, a CFD model has been developed and is used as the referent case. However, its large size requires a reduction. Two techniques have been tested depending on their fields of application: The NTF method and neural networks. They both appear highly appropriate drastically reducing the model's size (over 90% reduction) with a low relative error rate (under 10%). Furthermore, each method produces a reduced model which can be used in distinct specialized fields of application: the distribution of a quantity (mass fraction for example) in the cylinder at each time step (pseudo-dynamic model) or the qualification of scavenging at the end of the process (pseudo-static model).

***Keywords*—**Diesel engine, Design optimization, Model reduction, Neural network, NTF algorithm, Scavenging.

I. Introduction

THE complexity of current engineering problems leads us to model each element in our environment. In the field of engines, pollution issues and anti-pollution standards create a constant need for more accurate new models. Most studies focus on the combustion process due to its preponderant role in pollutant emissions. Less studied, but just as important, the scavenging process has a considerable influence on engine pollution. Especially in 2-stroke engines with ports, the scavenging process is fundamental to improve engine performance.

To study the scavenging impact on engine efficiency, we need to understand and model it. There are many solutions available to do this. CFD models are generally more complete models providing plenty of information about the process, but they are also resources and time consuming. On the other hand, reduced models are less accurate, but give a response in real time, or at least in a reasonably short period. Very simple scavenging models already exist ([1], [2], etc.), but these are too limited to be of any use for optimizing design. Several studies on parameter influence have already been carried out, but the complexity of engines forces us to take into account all the influent parameters on the scavenging process together.

In this paper, two model reduction techniques were tested: the β-NTF method and neural networks. These approaches are widely used in different areas, but not to model the scavenging process. However, each technique turned out to be very appropriate for modeling an aspect of scavenging. Their results, their fields of application, their advantages and disadvantages will be presented in this paper.

S. Cagin and X. Fischer are with Université de Bordeaux, I2M CNRS UMR 5295 33607 Bordeaux, France (e-mail: s.cagin@estia.fr) and with the ESTIA Recherche 64210 Bidart, France.

E. Delacourt, C. Morin and D. Coutelier are with UVHC, LAMIH CNRS UMR 8201 Le Mont Houy 59313 Valenciennes cedex 9, France

E. Delacourt, C. Morin, D. Coutelier and N. Bourabaa are with UVHC, ENSIAME Le Mont Houy 59313 Valenciennes cedex 9, France

B. Carré and S. Loumé are with AKIRA Technologies, ZA Saint Frédéric rue de la Galupe 64100 Bayonne, France

II. Context

A. 2-Stroke Diesel Engine

Two-stroke engines have suffered from high emissions and poor fuel saving, compared to more efficient 4-stroke engines. However, the car industry has recently shown a renewed interest for 2-stroke engines thanks to its advantages. First of all, 2-stroke engines are smaller and lighter than 4-stroke engines due to the lack of valves. They fire once every revolution so they provide higher power, and greater and smoother torque [3], [4]. The advantage is also linked to the development of new technologies which increase their efficiency. Real-time performance monitoring and evaluation have revolutionized engine design. However, before manufacturing engines, and in order to find the best design, it is essential to study its behavior and to test some configurations.

Engine performance depends on many factors, such as the quantity of injected gasoline or power duration. One of the crucial issues is the aerodynamics in the combustion chamber controlling the quality of scavenging and the composition of gases available for the combustion process. They play an important role in performance, pollutant emissions and engine efficiency. The operating principle of a 2-stroke Diesel engine (Fig. 1) with intake and exhaust ports is simpler than an engine with valves, but the advance of opening for both intake and exhaust ports are less adaptable: there is no possible variation of the opening crank angles. During the downward motion of the piston, the ports are uncovered and the scavenging process starts. The first ports opened are the exhaust ones because the pressure in the cylinder needs to drop first. Otherwise, some backflow through the inlet ports will be observed [3]. So, the burned gases begin to flow out the cylinder. As soon as the inlet ports are opened, the fresh gases enter the cylinder pushing out the burnt gases. After reaching the BDC (Bottom Dead Center), the piston moves upward, gradually covering the ports.

In addition to possible backflow, the main difficulties with the 2-stroke engine and a scavenging process with ports are the mixture of burnt and unburnt gases, and the short-cutting of the fresh charge as [5] underlines. Compared to the other issues, short-cutting is the most problematic. It characterizes the entry and exit of fresh gases before the end of the scavenging process which reduce the engine's efficiency. Short-cutting impact is particularly undesirable for spark-ignition engines because gasoline is incorporated into fresh gases. That is why its modeling is widely studied [6]. One of the first solutions used to reduce short-cutting was the addition of deflectors on the piston head. It had the advantage of significantly improving scavenging efficiency without any major design change. However, the evolution of piston head and especially the development of piston bowls (for direct injection systems), reducing pollutant emissions [7], made this solution obsolete. Remaining possibilities include the orientation of ducts and ports. To study all these parameters and their impact, some models are always used.

B. Model

A model is a representation of reality and is by definition abstract. A model is used to capture in a finite amount of parameters the way (or, at least, a part of) the world reacts. It is not exactly the reality, but it provides (within the boundaries set) a reliable estimation of the outputs [8]. A model consists of variables {V}, their ranges of values {D} and relations between variables {R}. To reduce a model implies lowering one of these three parameters. In this study, the reduction focuses on the number of variables.

The need for model reduction comes from the fact that, most of the time, current engineering problems have prohibitive sizes. The size problem comes from:

- The problem's space dimension: a huge object (like buildings, for example) or spatial discretization (fine mesh);
- The problem's time dimension: phenomenon very slow or over years (weathering…) or use of tiny time steps;
- The problem's parameter number: many variables to be taken into account.

In the CFD engine model, the three sources are encountered. Because of local turbulence phenomena, very fine meshes were used with up to 50,000 nodes. To avoid any divergence of the calculations, a time step of 3.10-6s (which corresponds to 0.045 crank angles) was used. To simulate the complete scavenging process, around one week of calculations was needed for each design. Any engine control in real time is impossible. Finally, 8 design parameters were taken into account in this study to optimize engine efficiency [8]:

- Crankshaft angle,
- Height of the exhaust port,
- Advance of opening of intake ports,
- Advance of opening of exhaust ports,
- Boost pressure,
- Difference between intake and exhaust pressures,
- Intake duct slope,
- Exhaust duct slope.

The aim of the developed scavenging model is to help optimize the cylinder design. It needs to run fast or it will not be of any use in solving industrial problems. The model needs to be reduced to make it more practical.

C. Scavenging Models

Scavenging is the process by which the burnt gases are replaced by fresh gases in the cylinder. In the engine cycle, it starts right after the expansion phase and finishes when the compression phase begins (Fig. 1).

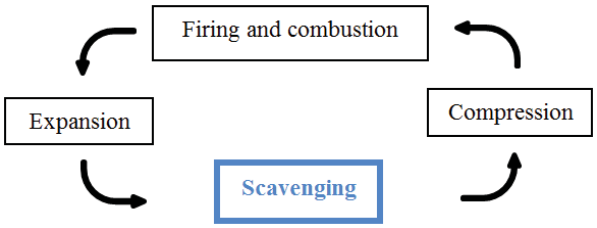

Fig. 1 2-Stroke engine cycle

Some behavioral models of scavenging in 2-stroke engines already exist. In all the models, three distinct processes are represented:

- Displacement (cf. Fig. 2 (a)): the burnt gases pushed out by entering fresh gases: fresh and burnt gases are considered as two separate zones,
- Mixing (cf. Fig. 2 (b)),
- Short-cutting (cf. Fig. 2 (c)).

In reality, scavenging is a combination of these three processes and the behavioral models try to combine them in order to be as realistic as possible. These theoretical models are generally divided into three groups: "one-phase", "multi-phase" and "multi-dimension" groups, as presented by Zhang [9]. None of these models consider any physical, fluid or thermal phenomenon; they only characterize mass fraction distribution according to empirical values of parameters.

In the one-phase group, we can find the perfect displacement and perfect mixing models (cf. Figs. 2 (a) and (b)), the first scavenging models, proposed by [2] and discussed by [14]. Very simplistic, they do not consider pressure or temperature variations or heat exchange between the gases and the walls. They are generally used to determine the efficiency of the scavenging process in the early stages of a design project. More realistic, "multi-phase" and "multi-dimension" models are more used. The classical models are as follows: The Maekawa "three-zone" model [1] (cf. Fig. 2 (c)), the "three-zone and multi-phase" model [15] and the "S-shape" [16]. However, they only provide the functional characteristics of scavenging introduced by [17] such as purity, the delivery ratio, etc. They are not usable to characterize the gas flow in the cylinder (such as velocity or pressure field) or the impact of any geometry change on the flow or scavenging. The specialization of their results makes them useless in most optimization design problems. Furthermore, they are very simplistic, which justifies the need for more complete models, such as CFD.

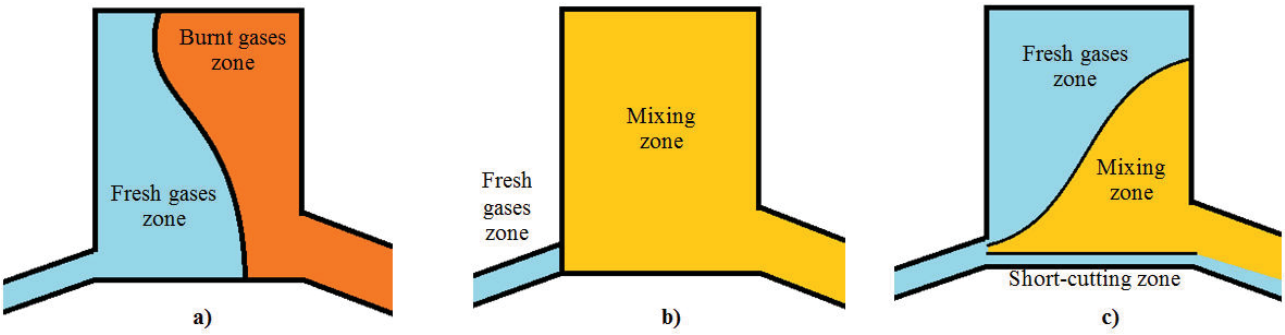

Fig. 2 Examples of scavenging models (a) Perfect displacement model [2], (b) Perfect mixing model [2], (c) "three-zone" model of Maekawa [1]

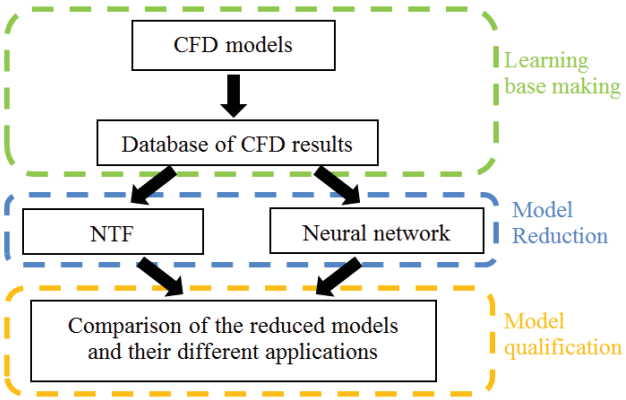

Fig. 3 Method of comparison

There are many CFD engine studies. All intake and exhaust configurations are studied: [10], [6], [11], [12] and [3] focused on flow phenomena in engines with ports; [4] and [13] were interested in engines with port (intake) and valve (exhaust) configuration. They mostly concerned small 2-stroke and 4-stroke engines because of the automotive industry's needs, but they are also used for large marine engines [5]. However, only a few papers focus on the CFD of the scavenging process, like [3] in engines with ports where the scavenging is very problematic.

Despite evolutions in technology and computing, CFD calculations still require huge resources and time. In this study, the authors look for a compromise between very complete CFD models and very fast behavioral models. To do so, reduced models are established from CFD results, as shown in Fig. 3. After determining the different cylinder designs which will be tested, CFD models provide the data to quantify and qualify the scavenging process (mass fractions of gases, pressure field, etc.). All reduced models are based on CFD calculations (more details about CFD model are given in [18]). There are several scavenging models needed to qualify the entire phenomenon. In this study, two different models are developed and qualified depending on their applications. With regard to methods for model reduction, many methods already exist, such as SVD (Singular Values Decomposition), POD (Proper Orthogonal Decomposition), PGD (Proper Generalized Decomposition), Arnoldi-Lanczos, Reduced Basis, etc. Because of the structural problem and the need to optimize cylinder design, the β-NTF and Neural Network (NN) methods were chosen. The NTF (Non-negative Tensor Factorization) enables us to separate time, space and design parameters which can be useful for the next step, optimization. On the other hand, the neural network provides the expected value of the output depending on the inputs thanks to its capability to simulate the system response. Both are used in this study depending on the final model structure needed.

TABLE I
SPECIFICATION OF THE DIFFERENT MODELS

	Model
CFD	1 engine configuration (2D) / simulation Global data
NTF	1 dynamic reduced model for the 27 configurations Global data
Neural Network	2 different reduced models (dynamic and static) for the 27 configurations Local data

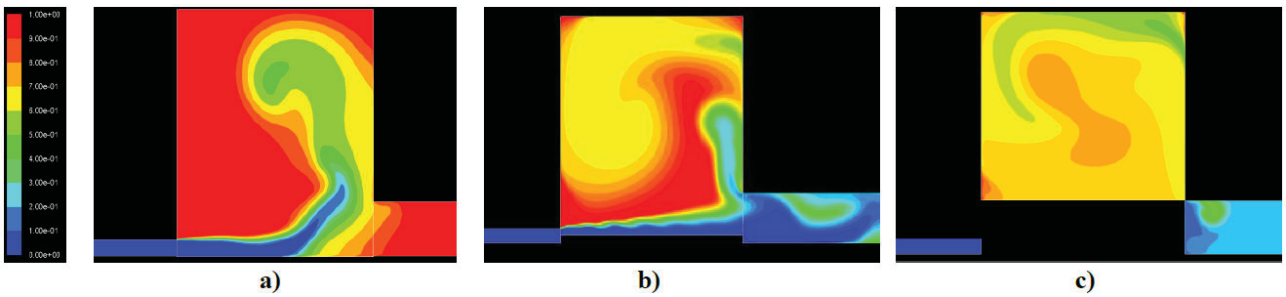

Fig. 4 CFD results, mass fraction of the fresh gases distribution for (a) θ=180 crank degrees, (b) θ=208 crank degrees, (c) θ= 245 crank degrees

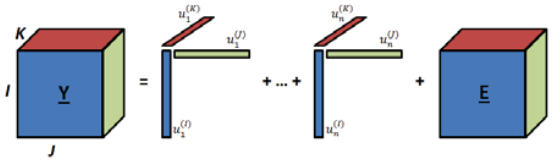

Fig. 5 NTF Concept

III. Model Development and Reduction

A. CFD Model and Database

The scavenging model clearly depends on the engine design. Because both intake and exhaust are done by ports, the need to develop a new scavenging model dedicated to our specific engine quickly appears. To do so, the CFD tool was used first. Both complete and accurate, the CFD model allows the several parameters to be observed: pressure, velocity, mass fraction of fresh and burnt gases, etc. (cf. Fig. 4). However, the problem of calculation time appears immediately because more than a day was needed to model the scavenging phase alone. The aim of the study is to integrate a scavenging model into a global 0D engine model: the scavenging model should provide results as fast as possible, which is not the case with CFD models. As explained in [18], it was chosen to model 27 designs of engine cylinder in CFD and to regularly extract data in order to generate a database. The database groups together all the CFD results and extracted data.

B. NTF Reduction

The NTF algorithm is very attractive because of its ability to take into account spatial and temporal correlations between variables more accurately than 2D Non-negative Matrix Factorization (NMF). As is the case with NMF, NTF also provides greater stability and a unique solution, as well as meaningful latent (hidden) components or features with physical or physiological meaning and interpretation [19]. Finally, the NTF algorithm is very simple to use in python and provides powerful implementation with multi-array data. The disadvantage of this method is the form of the reduced model. Indeed, the reduced model is still a 3D matrix. This form is very suitable for modeling the whole distribution of gases in the cylinder at each crank angle, but not so easy to implement in a 0D model, the final objective of this study.

1. Operating Principle

Parallel factor analysis (PARAFAC) is a tensor (multiway array) factorization method which allows us to find hidden factors (component matrices) from multidimensional data.

Non-negative Tensor Factorization (NTF) was first proposed by [20]. This algorithm is a generalization of Non-negative Matrix factorization (NMF) presented by [21]. NTF is based on a PARAFAC (Parallel Factors) analysis and its particularity is that it imposes nonnegative constraints on tensor and factor matrices. In addition to its efficiency and speed, the use of the NTF algorithm in this study was motivated by its ability to take into account spatial and temporal correlations between variables more accurately than 2D matrix factorizations, such as NMF (Nonnegative Matrix Factorization), and it usually provides sparse common factors or hidden (latent) components. Moreover, the spatial, temporal and parametric dimensions are totally independent from each other.

We consider $\underline{Y}$[1] as a 3D tensor, $\underline{Y} = \{y_{ijk}\} \in \mathbb{R}^{I \times J \times K}$. We have no negative value $y_{ijk} \geq 0 \ \forall \ i, j, k$. The NTF concept is illustrated in Fig. 5.

$$\underline{Y} = \sum_{i=1}^{n} \left(u_i^{(I)} u_i^{(J)} u_i^{(K)} \right) + \underline{E} = \underline{\hat{Y}} + \underline{E} \quad (1)$$

with $\underline{Y}$ the complete model, $\underline{\hat{Y}}$ the reduced model and $\underline{E}$ the error or the noise, the decomposition generates a set of three unknown matrices U(n) such as $U^{(n)} = \left[u_1^{(I)}, u_2^{(I)}, \ldots, u_n^{(I)} \right]$, with n the number of modes and $I = (I, J, K)$.

2. Application to Scavenging Model

To use the NTF algorithm, the data should be organized so as to obtain the $\underline{Y}$ model. The three dimensions were respectively associated with space, time and input parameter combinations (cf. Fig. 6).

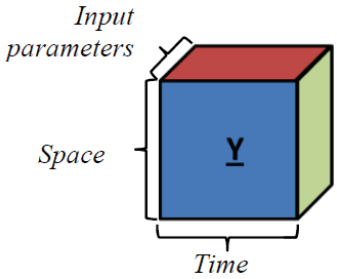

Fig. 6 NTF model of engine quantity

Based on the CFD model, several quantities (pressure, temperature, mass fraction of gases) were regularly extracted from each mesh node. For each quantity, a complete NTF model was defined. The study focuses on the scavenging efficiency of the 2-stroke engine; only the mass fractions of burnt and fresh gases will be presented here.

Due to piston motion during the simulation, some nodes were added or removed from the cylinder mesh (cf. [18]). The number of nodes changes between two successive crank angles. To overcome the variable data number, a normalized mesh (81x81 = 6561 elements) with constant numbers of nodes was created. For each crank angle, the CFD results were projected on the normalized mesh and the mass fraction quantities extracted from the normalized mesh nodes. The projection was an initial model reduction: the minimum number of nodes of the moving mesh is 20,000 elements.

C. Neural Networks

Neural networks provide the expected output of the system whatever the input. From the learning data, it can approximate

[1]We denote a tensor by a bold-face underlined capital letter, a matrix by a bold-face capital letter, a vector by a bold letter and a scalar by a plain letter

the system's behavior. The relevance and accuracy of neural network output outside the learning points should be checked, but its ability to handle all input combination makes it very useful. Converging towards the most suitable structure (number of neurons in layers…) and the best parameter values (biases and weights) is complex. Indeed, biases and weights are particularly sensitive to their initial values, which are randomly chosen. Successive tests should be carried out.

1. Operating Principle

A neural network can be seen as a complex mathematical function that accepts numerical inputs and generates numerical outputs. The general structure of a neural network is presented in Fig. 7.

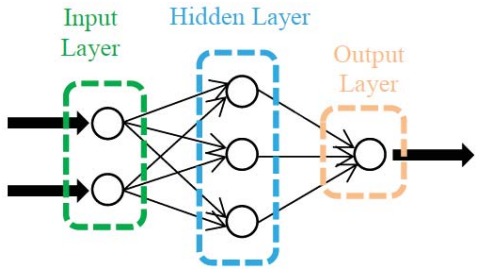

Fig. 7 Example of fully connected neural network

The values of the outputs are determined by the input values, the number of so-called hidden processing nodes, the hidden and output layer activation functions, and a set of weights and bias values.

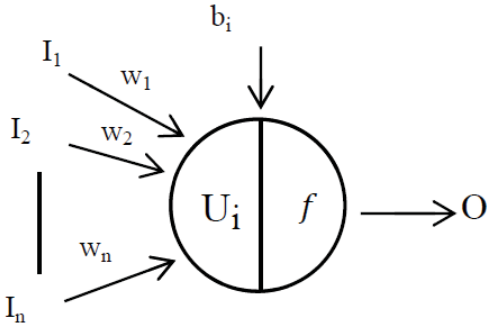

Fig. 8 Structure of neuron i

In Fig. 8, I_j are the n inputs of the node, each input I_j is respectively weighted by w_j. These weights determine the way each input impacts the neuron output O. The output O also depends on activation function f of the node. The node output is calculated thanks to (2):

$$S = f(\sum_i w_i E_i) + b_i = f(U_i) + b_i \quad (2)$$

U_i is the sum of the inputs weighted with their respective weight.

A fully connected neural network with m inputs, h hidden nodes, and n outputs has (m * h) + h + (h * n) + n weights and biases. For example, the neural network of Fig. 7 with 2 inputs, 3 hidden nodes, and 1 outputs has (2 * 3) + (3 * 1) = 9 weights and 3 + 1 = 4 biases. The number of weights rises very quickly with the neuron number of the hidden layer. Therefore, the first step is to choose the network's general architecture well. The number of input neurons is equal to the number of model inputs. Likewise, the number of output neurons is defined by the number of numerical results the network should provide. Only the number of hidden layers (and its number of neurons) and the activation functions can be freely chosen to improve the calculation time and accuracy of the output results.

After defining the neural network structure, the next step is its training. Training a neural network is the process of finding values for the weights and biases so that, for a set of training data with known input and output values, the computed outputs of the network closely match the known outputs. Several learning paradigms already exist. They are divided into three main groups according to their particular abstract learning tasks: supervised, unsupervised and reinforcement learning. Their differences result from the dataset, and especially the type of outputs.

With supervised training, a unique output vector is associated with each input vector. The neural network calculates the outputs according to the inputs and compares its results with the real ones. Depending on the error, the neural network adapts the weights and bias to be as close as possible to the real solution. The expected outputs have to be known to determine and reduce the network error.

With unsupervised training, no information about the expected outputs is provided: there is no supervisor. The network statistically analyzes and sorts the data according to shared properties.

With reinforcement learning, it is not possible to provide the real outputs to the network, but only qualitative outputs (right/wrong, success/fail, etc). The network maximizes its performance index during training and it is able to evaluate the relevance of its outputs (even if it does not know the real expected outputs).

Thanks to the results of ANSYS Fluent models and python treatments, the outputs of the engine cylinder with particular input combinations are known. So, supervised training has been selected.

2. Learning Algorithm

Most of the algorithms used to train neural networks use the gradient descendent algorithm (a first-order optimization algorithm).

The Rprop (Resilient backpropagation) algorithm was used, with backpropagation of errors to converge toward the best solution. First introduced in 1993 by Riedmiller and Braun [22], the Rprop algorithm is faster than classical backpropagation learning. Indeed, the effectiveness of backpropagation methods is highly sensitive to the value of the learning rate, whereas Rprop overcomes this difficulty. The Rprop algorithm provides an individual step-size to each weight in order to minimize oscillations and to optimize step-size during their updates.

The principle of the method is briefly presented by Igel and

Hüsken [23]. w_{ij} is the weight from neuron j to neuron i and E the error measured. E is differentiable with respect to the weights. Bias parameters are considered as weights from an extra constant input. Superscripts indicate the learning iteration. At each iteration, each weight is updated from the previous one thanks to (3):

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \quad (3)$$

Criteria can be defined to stop the learning algorithm (e.g., t exceeds a predefined value for example). The direction of each weight update is based on the sign of the partial derivative $\partial E/\partial w_{ij}$. The update amount of a weight Δw_{ij} (also called step-size) is individually adapted for each weight (at each iteration). Unlike other techniques, the Rprop algorithm adapts the step-sizes independently of the absolute value of the partial derivatives, it only takes into account the sign of the derivative. Because no weight-backtracking is used, the step-sizes are defined by:

$$\Delta w_{ij}^{(t)} = -\text{sign}\left(\frac{\partial E}{\partial w_{ij}}\right)\Delta_{ij}^{(t)} \quad (4)$$

The sign operator returns +1 if its argument is positive, −1 if the argument is negative and 0 otherwise. Δ_{ij} is initialized to a constant Δ_0. The benefits of this update scheme, described by Riedmiller [22], [24], are essentially a more stable learning step and a faster convergence of the error function.

Δ_{ij} is adjusted at each iteration as shown in (5):

$$\Delta_{ij}^{(t)} = \begin{cases} \min\left(\eta^+\Delta_{ij}^{(t-1)}, \Delta_{max}\right) & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \max\left(\eta^-\Delta_{ij}^{(t-1)}, \Delta_{max}\right) & \text{if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)} & \text{otherwise} \end{cases} \quad (5)$$

With $0 < \eta^- < 1 < \eta^+$. With (5), the step size increases if $\partial E/\partial w_{ij}$ possesses the same sign for two consecutive steps. On the contrary, $\Delta_{ij}^{(t)}$ decreases if the partial derivative signs change. This principle is widely used in other learning methods. The step size value is bound by Δ_{min} and Δ_{max}, defined by the user. Hence, the direction of the change of Δ_{ij} following (5) is in accordance with a gradient-based optimization of the network error with respect to Δ_{ij}.

IV. Modeling Methods

A. NTF Reduction

To limit both costs and time calculations, only 27 cylinder configurations were tested. The choice of input parameter combinations was made according to the L27 table from the Taguchi design experiment. The configurations will not be discussed in this paper, but the reader can find more details in [8], [18].

To reduce time calculations, only the scavenging process is modeled in CFD. Depending on the opening angle of the exhaust port, the simulation begins at θ = 94 crank angle after TDC (Top Dead Center). The time step used was 3.10-6s, the data were extracted at 22 time step intervals, equivalent to 0.99 crank angles, 174 time steps were simulated. As already explained, the NTF reduction and the β divergence [19] were used to model the distribution of species in the cylinder during the scavenging process. Referring to part 1, the dimensions of the $\underline{Y}$ matrix in this study are 174 x 6561 x 27. So, the CFD complete model has more than 30 million variables.

First, the influence of the number of design configurations on the average relative error was tested (Fig. 9). The number of design configurations directly impacts the model size and the dimension of the $\underline{Y}$ matrix. In this paper, the relative error was calculated with (6):

$$\text{relative error}_i = \frac{|\text{Output}_{\text{complete model i}} - \text{Output}_{\text{reduced model i}}|}{|\text{Output}_{\text{complete model i}}|} \quad (6)$$

To get the average error, (7) was used:

$$\text{average error} = \frac{\sum_{i=1}^{n} \text{relative error}_i}{n} \quad (7)$$

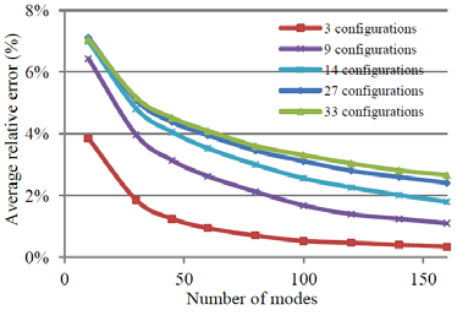

Fig. 9 Evolution of the relative error depending on the number of configurations

Fig. 9 shows that, for the same number of modes, the reduction error rises when the number of configurations increases. However, it also highlights the fact that the influence of the model size on the error decreases when the size of the model increase. Beyond 14 configurations, the error increases are almost negligible. Any new configuration results added to the CFD model will not impact the accuracy of the reduced model.

The results of the NTF reduction are summarized in Table II. Several modes were tested and for each mode the following were calculated:

- The average relative error between the normalized model and the NTF reduced model,
- The percentage of reduction (taking into consideration the number of elements before and after NTF reduction),
- The time needed to converge towards a satisfying reduced model.

As seen in Table II, number of modes should be chosen carefully. This number influences both the reduction error and the time needed to establish the reduced model. The duration of the model development proportionally increases with the number of modes. With 160 modes, more than 5 days are

needed to get the complete reduced model. On the contrary, the average relative error was calculated comparing the normalized model and the NTF reduced one. As expected, the error decreases when the number of modes rises (Fig. 10). In addition, over 10 modes, the error is always under 8% which confirms the NTF method is well adapted to extensive models: It is able to compress 99.8% introducing less than 8% of errors. On the contrary, an error of 2% seems to be the minimum that can be achieved (Fig. 10). Because of time increases, to go over 160 modes seems useless: it will reach the time needed to solve the CFD model.

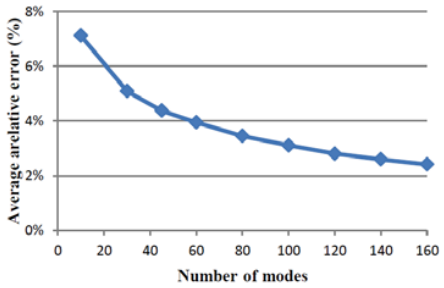

Fig. 10 Average relative error depending on the number of modes

The distribution of the average relative error (cf. Fig. 11) underlines the fact that the error is maximum at θ=180.13 crank angle. This corresponds to the moment when the ports are fully opened and the gas exchanges are greatest. Until θ=125 crank angle, all intake ports are closed. Only burnt gases are in the cylinder which explained why the error equals 0. After, depending on the configuration, the intake ports open and the exchanges of gases start which explained the gap between normalized and NTF reduced models, and the error increases.

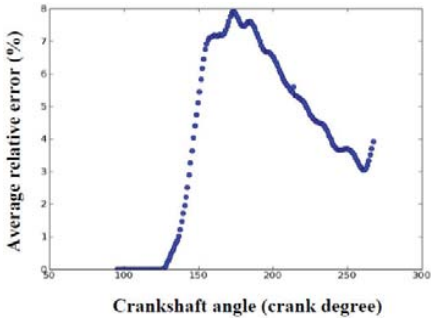

Fig. 11 Distribution of average relative error of all configurations depending on the crank angle for 160 modes

TABLE II
RESULTS OF NTF REDUCED MODELS

Modes	Average rel. error	Normalized model size	Reduced model size	% of reduction	Time
10	7.12%	30,823,578	67,620	99.78%	18.6h
30	5.09%	30,823,578	202,860	99.34%	32.5h
45	4.38%	30,823,578	304,290	99.01%	44.2h
60	3.94%	30,823,578	405,720	98.68%	56.3h
80	3.45%	30,823,578	540,960	98.24%	71.2h
100	3.11%	30,823,578	676,200	97.81%	82.2h
120	2.80%	30,823,578	811,440	97.37%	99.2h
140	2.60%	30,823,578	946,680	96.93%	109.3h
160	2.41%	30,823,578	1,081,920	96.49%	123.2h

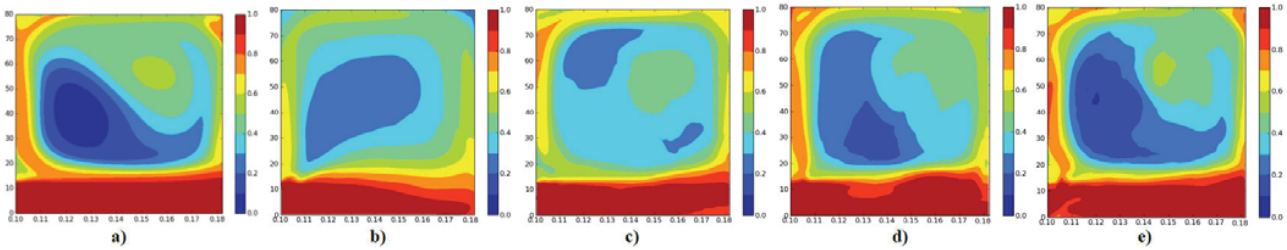

Fig. 12 Distribution of fresh gases mass fraction of configuration 15 at θ=180.13 crank angles for (a) Normalized model (b) 10 modes (c) 60 modes (d) 120 modes (e) 160 modes

Configuration 15 at θ=180.13 crankshaft angles corresponds to the maximum error observed with the NTF reduction, (19.6%) whereas for all the other configurations the maximum average relative error is always under 15%. So, this configuration and this angle will be used as examples in the following parts.

The impact of the number of modes is particularly visible in Fig. 12. The figure also illustrates the error decrease. The 160 mode model appears to be the closest to the normalized model.

Thanks to normalization, no error is introduced: only a loss of information can be observed. However, we are interested in the whole distribution in the cylinder, not just by small local phenomena. So, the loss of accuracy compared to the need for normalized data and the reduction model (between 65% and 80% compression depending on the number of mesh nodes) has no repercussion on the model's error. On the contrary, the NTF algorithm introduces errors between the models before and after reduction (cf. Fig. 13).

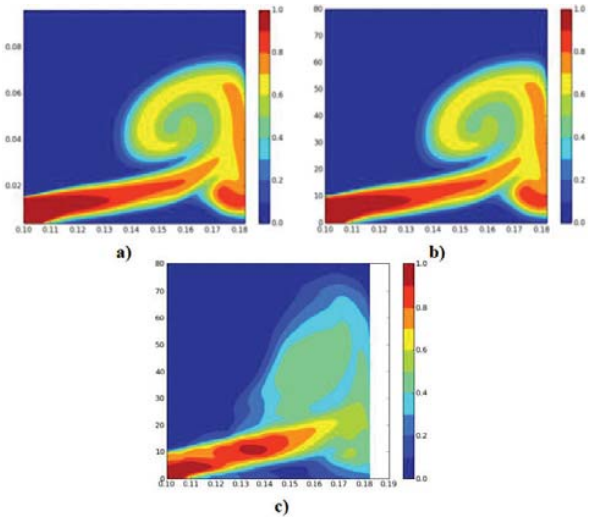

Fig. 13 Configuration 15 at θ=151.42 crank angles with (a) CFD model (b) normalized model (c) NTF reduced model

Due to its matrix form, the NTF reduced model is only usable for the 27 configurations tested. It cannot directly provide results for any other design configurations. On the other hand, it can be useful to know the field of any quantity at any crank angle. At any time, this model can indicate if any backflow, short-cutting or mixing occurs and quantify it. Finally, the reduction of more than 95% is very useful to know the distribution of any quantity (pressure, temperature, etc.) at any time.

Thanks to interpolation and the kriging method, the NTF reduced model will be used to forecast the evolution of the distribution of gases inside the cylinder during the whole scavenging process whatever the configuration.

B. Neural Network Reduction

Thanks to their structure, the neural networks are really useful for optimizing cylinder design. Indeed, after training, it is capable of interpolating the system's behavior whatever the inputs. Of course, there is no information on the accuracy of the results outside the learning points, but it should provide an estimation of the expected results.

With neural networks, multiple parameters can be selected to improve the network's efficiency such as:

- Number of hidden layers,
- Number of neurons in the hidden layers,
- Activation functions.

The models are developed to then be integrated in a 0D model simulating the whole engine environment [18]. To do so, the simpler the neural network is, the easier its integration is in the 0D model. This is why only one hidden layer is used.

The neural networks were developed thanks to python and pybrain codes. The network is used to qualify the scavenging process evaluating the delivery ratio λ and the trapping η_{tr}, scavenging η_{sc} and charging η_{ch} efficiencies at the end of the process. The parameters used as inputs are presented in Section II *B*. Because the process is evaluated when the ports are closed, the crank angle is not considered as an input. In any case, 10% of the dataset is dedicated to verification during the training phase; the other data are used for the learning aspect.

Firstly, only one neural network was developed with four outputs. After several tests with different structures (changing the number of neurons and the activation function type), the results underlined the network's inability to reduce the relative error of all four outputs at the same time. At least, always one output has an error over 80%, whereas the other errors are between 10% and 20%. It was decided to use one neural network for each parameter. The results for the different parameters are summarized in Table III: depending on the number of hidden neurons and the activation functions selected, the relative error for the learning data and the relative error for all the data are calculated. The best structure for each output is highlighted in green. Weights and biases of the network are randomly initialized, and the solution the network converges towards is directly linked to these initial values. Several tests were carried out to achieve these results.

TABLE III
NEURAL NETWORK RESULTS

Output	Activation function	Number of hidden neurons	Relative error	
			Training data	All data
λ	Sigmoid	8	6.19%	12.3%
η_{tr}		8	0.65%	2.5%
η_{ch}		8	0.09%	11.9%
η_{sc}		8	2.62%	10.3%
λ	Sigmoid	10	2.56%	7.9%
η_{tr}		10	0.01%	12.4%
η_{ch}		10	0.05%	15.7%
η_{sc}		10	0.11%	11.3%
λ	Sigmoid	12	5.58%	9.4%
η_{tr}		12	0.02%	7.8%
η_{ch}		12	0.05%	18.4%
η_{sc}		12	6.55%	10.4%
λ	Tanh	8	5.31%	10.4%
η_{tr}		8	0.78%	7.0%
η_{ch}		8	2.17%	9.8%
η_{sc}		8	3.02%	5.2%
λ	Tanh	10	2.13%	16.6%
η_{tr}		10	0.34%	11.4%
η_{ch}		10	0.10%	23.6%
η_{sc}		10	1.90%	16.0%
λ	Tanh	12	2.52%	13.9%
η_{tr}		12	0.02%	7.0%
η_{ch}		12	0.36%	7.1%
η_{sc}		12	0.02%	14.3%

The first striking result is that the best structure of each output is different from the others. This confirms the need to develop one neural network for each parameter. In addition, the relative error of the delivery ratio and the charging efficiency is over 7%. It is too high to be satisfying. The first advantage in calculating these four outputs is to indicate the influence of each input on the scavenging process in order to optimize the cylinder design. With a relative error of almost 8%, the relevance of the neural network's results is questionable. To reduce the error the dataset should be

completed with other CFD results. Another way to reduce the error is to initialize the weights and biases with better values in order to converge towards a better solution or to change the network structure to something more appropriate.

In any case, neural networks are very flexible with the input values which are very useful in optimization problems. Even if, outside the learning points, the results can be qualified, they are generally very efficient and justify their widespread use in various sectors.

V. Summary of the Results

The performances of the three models are summarized in Table IV.

TABLE IV
Comparison of Model Performances

	CFD	NTF (160 modes)	Neural Network
Calculation time	1 week x 27 designs	5 days	30 min
Size	20,000 at θ = 94 crank angles (beginning) 50,000 at θ = 180 crank angles	30.823.578	1
Average relative error (compared to CFD results)		2.41%	2.5% – 7.9%
Number of iterations	5,167,800	4,000	> 170,000

Table V does a comparison of the different models used in this study.

TABLE V
Comparison of NTF and NNR Methods

	Advantages	Disadvantages	Field of application
CFD	✓ Complete model ✓ Based on physical and behavioral principles	× Resource consuming × Time consuming × Model size	To create a database or to check the performances of a specific design
NTF	✓ Repeatability of the method ✓ Separation of the variables ✓ Very low average error ✓ Size of the database	× Matrix form of the model	To model the evolution of the distribution of a quantity (mass fraction for example)
Neural network	✓ Adaptable outside the learning points, capable to interpolate ✓ Model form: 1 analytical equation ✓ Very fast reduction	× Dependent to the initialization of its parameters	To evaluate a quantity for a particular combination of inputs

VI. Conclusion

The lack of efficient and realistic scavenging models in two-stroke Diesel engines motivated this study. Based on CFD 2D models, 2 different scavenging models were developed and reduced. Depending on the model's use, several reduction techniques can be used. To select the most suitable one, several criteria are usually defined: the dataset form, the amount of data, the maximum error tolerated… However, the most important criterion is the way the reduced model will be used afterwards! As presented in this study, NTF reduction leads to matrix reduced models, whereas the neural network develops analytical models. Their respective forms are suitable to the utilization done after. The NTF reduced model allows the observation of the gas flow inside the cylinder at every time step. It locally qualifies the flow, but it does not provide any global information. On the other hand, the neural network reduced (NNR) model qualifies the scavenging process in its entirety. It is impossible to know the evolution of the process step by step but the influence of each design parameter is clearly identified. In addition, the NNR model offers the possibility of testing configurations other than only those in the dataset, unlike NTF reduced models. In spite of its rigidity in terms of input data, NTF reduced models provide greater accuracy a very high data compression (over 90% compression and almost 2% relative error): its efficiency is undeniable. On the contrary, the dependence of neural networks on initial values (weights and biases) makes their convergence unpredictable and hard to control.

References

[1] Maekawa M. "Text of Course". JSME G36. 23, 1957;
[2] Hopkinson B. "The Charging of Two-Cycle Internal Combustion Engines". Journal of the American Society for Naval Engineers. 26, 3, 974–85, 1914;
[3] Mattarelli E. "Virtual design of a novel two-stroke high-speed direct-injection Diesel engine". International Journal of Engine Research. 10, 3, 175–93, 2009;
[4] Trescher D. "Development of an Efficient 3–D CFD Software to Simulate and Visualize the Scavenging of a Two-Stroke Engine". Archive of Computational Methods in Engineering. 15, 1, 67–111, 2008;
[5] Lamas Galdo M.I., Rodríguez Vidal C.G. "Computational Fluid Dynamics Analysis of the Scavenging Process in the MAN B&W 7S50MC Two-Stroke Marine Diesel Engine". Journal of Ship Research. 56, 3, 154–61, 2012;
[6] Hong G., Mack A.N.F., Menolotto F., Jamieson C.S., Main S.G. "Numerical Visualization of Air Short-Circuiting in a Small two-stroke SI engine". SAE Technical paper 2004-32-0009. 2004;
[7] Prasad B.V.V.S.U., Sharma C.S., Anand T.N.C., Ravikrishna R.V. "High swirl-inducing piston bowls in small diesel engines for emission reduction". Applied Energy. 88, 2355–67, 2011;
[8] Cagin S., Fischer X., Bourabaa N., Delacourt E., Morin C., Coutellier D. "A Methodology for a New Qualified Numerical Model of a 2-Stroke Diesel Engine Design". The International Conference On Advances in Civil, Structural and Mechanical Engineering - CSME 2014. Hong-Kong; 2014.
[9] Zhang Q.X. "Modelling The Scavenging Process in a Two-Stroke I.C. Engine". Thesis (Coursework Master thesis). 1995.
[10] Kato S., Nakagawa H., Kawahara Y., Adachi T., Nakashima M. "Numerical analysis of the scavenging flow in a two-stroke-cycle gasoline engine". JSME international journal. Series 2, Fluids engineering, heat transfer, power, combustion, thermophysical properties. 34, 3, 385–90, 1991;
[11] Pitta S.R., Kuderu R. "A computational fluid dynamics analysis on stratified scavenging system of medium capacity two-stroke internal combustion engines". Thermal Science. 12, 1, 33–42, 2008;
[12] Noor M.M., Kadirgama K., Devarajan R., Al. E. "CFD Simulation and Validation of the In-Cylinder Within a Motored Two Stroke Si Engine". 2nd International Conference on Science & Technology, Penang. 2008.
[13] Ingvorsen K.M., Meyer K.E., Schnipper T., Walther J.H., Mayer S.

"Swirling Flow in Model of Large Two Stroke Diesel Engine". 16th International Symposium on Applications of Laser Techniques to Fluid Mechanics, Lisbon. 2012.
[14] Blair G.P. "Design and Simulation of two-stroke engines". Engineers, Society of Automotive; 1996.
[15] Benson R.S., Whitehouse N.D. "Internal Combustion Engines". Pergamon Press; 1983.
[16] Sher E. "A new practical model for the scavenging process in a two-stroke cycle engine". SAE Technical paper 850085. 1985;
[17] Heywood J.B. "Internal Combustion Engines Fundamentals". McGraw-Hill. Duffy A, Moms JM, editors. 1988.
[18] Cagin S., Bourabaa N., Delacourt E., et al. "Scavenging Process Analysis in a 2-Stroke Engine by CFD Approach for a Parametric 0D Model Development". 7th International Exergy, Energy and Environment Symposium. Valenciennes; 2015.
[19] Cichocki A., Zdunek R., Choi S., Plemmons R., Amari S.-I. "Non-Negative Tensor Factorization using Alpha and Beta Divergences". IEEE International Conference on Acoustics, Speech and Signal Processing. Honolulu, HI; 2007.
[20] Shashua A., Hazan T. "Non-negative tensor factorization with applications to statistics and computer vision". ICML '05 Proceedings of the 22nd International Conference on Machine learning. New-York; p. 792–9, 2005.
[21] Lee D.D., Seung H.S. "Learning the parts of objects by nonnegative matrix factorization". Nature. 401, 788–91, 1999;
[22] Riedmiller M., Braun H. "A direct adaptive method for faster backpropagation learning: the RPROP algorithm". In: Ruspini EH, editor. Proceedings of the IEEE International Conference on Neural Networks. New-York: IEEE Press; p. 586–91, 1993.
[23] Igel C., Hüsken M. "Empirical evaluation of the improved Rprop learning algorithms". Neurocomputing. 50, C, 105–23, 2003;
[24] Riedmiller M. "Advanced supervised learning in multi-layer perceptrons—from backpropagation to adaptive learning algorithms". Computer Standards Interfaces. 16, 5, 265–78, 1994;