Atmospheric dispersion modeling using Artificial Neural Network based cellular automata

Pierre Lauret, Frederic Heymes, Laurent Aprin, Anne Johannet

Atmospheric Dispersion Modeling using Artificial Neural Network based Cellular Automata

Pierre Lauret[a], Frédéric Heymes[a], Laurent Aprin[a], Anne Johannet[a]

[a]Laboratoire de Génie de l'Environnement Industriel, Ecole des Mines d'Alès, 6 avenue de Clavières, 30100 Alès, France

Corresponding author: pierre.lauret@mines-ales.fr

Keywords: Atmospheric Dispersion, Artificial Neural Network, Cellular Automata, Computational Fluid Dynamics

Highlights

- A new atmospheric dispersion model is developed based on combination of Cellular Automata and Artificial Neural Networks (CA-ANN).
- Comparisons are made with CFD RANS standard *k-ε* model on 2D free field dispersion of methane.
- CA-ANN is faster than CFD standard *k-ε* by a factor from 1.5 to 120 in the modelled simulations while keeping accuracy.

Abstract

Forecasting atmospheric dispersion in complex configurations is a current challenge in fluid dynamics in terms of calculation time and accuracy. CFD models provide good accuracy but require a great computation time. Simplified or empirical models are designed to quickly evaluate the dispersion but are not adapted to complex geometry. Cellular Automata coupled with an Artificial Neural Network (CA-ANN) are developed here to calculate the atmospheric dispersion of methane (CH_4) in 2D. Efforts are made in reducing computation time while keeping an acceptable accuracy. A CFD simulations database is created and the Advection-Diffusion Equation is discretized to provide variables for the ANN. Neural network design is made thanks to best sampling selection, architecture selection and optimized initialization. The coefficient of determination is over 0.7 for most cases of the test set despite small errors accumulated through time steps. CA-ANN is faster than CFD models by a factor from 1.5 to 120.

1. Introduction

Industrial accidents involving atmospheric dispersion of flammable or toxic materials may generate extremely serious consequences. The disaster that occurred in *Bhopal*, India, in 1984 clearly shows the impact of toxic dispersion from a chemical industry. 40 tons of methyl isocyanate, or MIC, were released in 90 minutes from the Union Carbide fertilizer factory into the city of *Bhopal* after a cleaning operation. Sharan & Gopalakrishnan (1997) highlight the impact of topography and atmospheric conditions on the evolution of the plume. Especially, near field characteristics (in this case the presence of lakes near the factory) influenced directly the plume trajectory toward the city.

In case of flammable gas emissions, the consequences of a potential gas explosion depend especially on the plume size and on the concentrations before ignition. Therefore, the plume behavior just after the release is a key point to assess possible consequences. Brambilla et al. (2010) support the necessity to consider complex environment in specific cases. The Italian *Viareggio* train accident that occurred in 2009 led up to a liquefied petroleum gas transport tank leakage. The plume was dispersed in a specific manner because of the street configuration in the near field. The presence of buildings in the vicinity of the source is considered as the main important parameter in the plume dispersion.

The near field of the leakage, hence, appears to be significant for the pollution plume dispersion, and must be considered for forecasting this phenomenon.

To avoid such accidents, the risks analysis is currently performed with atmospheric dispersion models. Existing models currently distinguish two mechanisms, the wind flow and the dispersion process. Each model differs from the other, according to turbulence model. In terms of performance, two major criteria could be used: computational time and model accuracy.

The best model should be fast and accurate. Since this best model does not exist, available models are designed according to the goal to achieve, preferring accuracy or computation time. Nevertheless, attempts were made to combine both capacities. This study tries to make a step toward this objective using a method well known for its ability to represent any complex phenomena: neural networks. In order to take into consideration the spatial extension of the dispersion phenomenon, a new way of modeling is proposed combining cellular automata and neural network. The former represents the spatial dispersion of the phenomenon, the last implements the transition rule.

The case study proposes the horizontal 2D dispersion faced with a single obstacle but, as this approach is innovative and was not previously used for gas dispersion, several questions are addressed in this study:

(i) the constitution and sampling of the database,
(ii) the design of the neural model,
(iii) the way to assess the quality of the model.

This paper is divided into six parts: after the introduction, a state of the art regarding dispersion modeling and neural networks modeling is proposed in parts 2 and 3. The proposed method is extensively described in part 4 addressing all the new questions as: database constitution, design of neural model, training and validation of the model. Part 5 proposes a presentation of the results and a discussion in relation with the ability of the proposed model to converge toward the targeted solution: CFD simulation. A conclusion is then proposed in section 6.

2. Atmospheric dispersion models review

2.1. Physics-based models

Usually, atmospheric dispersion modeling is done through the use of wind flow, calculated or determined, combined with dispersion modeling. In each model, turbulence is the main difficulty and has a major impact on the dispersion. Figure 1 details turbulence modeling for several main modeling methods detailed in the following:

- Dispersion modeling in gaussian models is calculated by solving Advection Diffusion equation using turbulent diffusion coefficient or standard deviations determined empirically. These models consider the wind flow as homogeneous.
- Models from Computational Fluid Dynamics solved the Navier-Stokes equations to determine the wind flow. Turbulence is solved using closure equations of the system. These equations are transport equations of turbulent quantities. The turbulent diffusion coefficient is introduced in the advection-diffusion equation to model the dispersion. This parameter is directly linked to the variables of the closure equations.
- Intermediate simplified CFD models exist. Diagnostic wind flow models are capable of reconstructing a steady-state wind field from initial experimental data. They are based on simplified steady-state solutions of the Navier-Stokes equations.
- In Lagrangian models, the wind flow is determined from CFD eulerian model. Dispersion is realized by following the behavior of particles linked to initial conditions.

Turbulence modeling is realized by adding fluctuation term to the mean velocity field in each particle position.

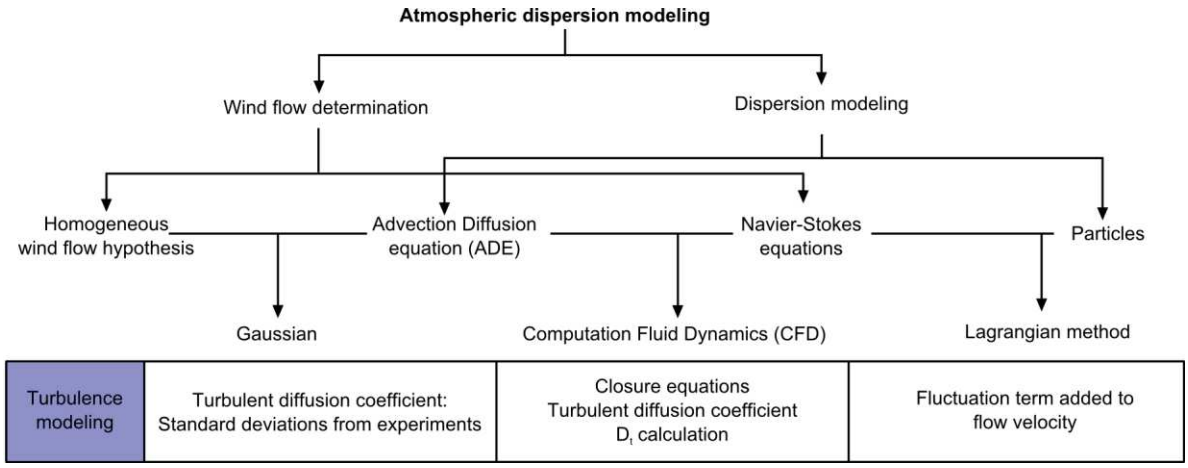

Figure 1 : Atmospheric dispersion modeling methods

Gaussian models correspond to an analytical solution of the advection diffusion equation for idealized circumstances using Reynolds averaging. The main assumptions are:

- Gas dispersion is considered as passive
- Dispersion through turbulent diffusion is both isotropic and homogeneous
- Molecular diffusion is neglected
- Obstacles and relief are not considered so that wind field is considered as uniform in terms of time and space.

Therefore, Gaussian models are efficient in far field evaluation of atmospheric dispersion, for passive gas. They are mostly used to assess long term impact of industrial activities on the environment. Dispersion and physicochemical processes are included through specific parameterizations. These models require experimental dispersion coefficients calibrated from field experiments. Prairie grass (Barad, 1958) was one of the first campaign of field experiments. More recently, these coefficients were tested for urban environment using Indianapolis experiments (Hanna et al., 1999). These models are mainly adapted to operational purpose or emergency management due to the short computation time. However, complex geometries and site topology are generally not appropriately addressed. It notices that some modifications were proposed in the literature in order to adapt Gaussian models to non-passive gas dispersion. These integral models are based on properties conservation through the resolution of the fluid mechanics simplified equations. Atmospheric dispersion is split into different steps and specific models are applied for each one. For the final step, corresponding to the atmospheric dispersion modeling, the gas is considered as passive and Gaussian model is applied. This conservative approach also needs parameters defined by experiments and due to the use of a Gaussian model; integral models suffer from

same limitations considering weather, complex geometries and site topology. Comparisons between Gaussian models and Lagrangian codes are well described by Caputo et al., (2003).

Diagnostic 3D wind flow models, also called kinematic models, generate a wind field by sustaining some physical constraints. In mass-consistent models, numerical solution of the steady-state three-dimensional continuity equation for the mean wind components is imposed. Parametric relations or wind data are used to consider momentum and energy equations which are not solved explicitly. In consequence, diagnostic wind flow models are specifically adapted to predict effects of orography (Castellani et al., 2015) but cannot take into account of thermal effects or effects due to pressure changing gradients.

Computational Fluid Dynamics (CFD) are numerical models solving Navier Stokes equations to obtain wind and turbulence fields. The Advection-Diffusion Equation (ADE) is solved using the previous results. The solver computes momentum and mass balance on a mesh, using domain initialization and boundary conditions. Closure equations are needed for modeling turbulence. The most common are given by Launder and Spalding (1974) introducing turbulent kinetic energy k and its dissipation rate ε. These models provide quite accurate results for turbulence. More finer is the mesh, more accurate is the simulation that it gets. Consequently, the computation time increases fastly with complex geometries and dimensions of studied area. Furthermore, setting up this kind of models requires specific expert skills.

Lagrangian models calculate trajectories of a large number of particles, depicting the pollutant. Wind velocity and turbulence field are usually computed using CFD. Statistical information such as standard deviation of wind fluctuation and autocorrelation time are needed. Trajectory simulation is computed using the mean velocity field enhanced at each time of a fluctuation component. This kind of model is relatively fast to compute once the wind field is known (Vendel, 2011).

To sum up, modeling of flow and dispersion over a complex terrain with accuracy is a difficult task that requires generally expensive computation. It is seldom compatible with operational needs of emergency response or risk assessment.

2.2. Alternative models

Previous attempts aiming to reduce the computational time in CFD were achieved in the last decade.

Lattice Boltzmann methods (LBM) are a class of CFD models similar to cellular automata at microscopic scale. These kinds of models consider a mesh of nodes, where particles of

same mass are located, depending of the initialization. Considering the first model proposed by Hardy et al. (1973), the discrete Boltzmann equation is solved following Pauli exclusion principle: two particles cannot be located on the same node if they have an initial velocity. Two situations can possibly happen: free translation, depending on the initial velocity of the particle and collision between two or more particles. In this latter case, direction of the movement is changed based on rules initially defined. Several models exist and differ from each other in the definition of the rule and the geometry of the mesh. For example, Chopard et al. (2002) used the generic dynamics of Lattice Boltzmann fluid models and derived the corresponding macroscopic behavior. Considering complex geometry with such methods is not difficult because the mesh, as in cellular automata, is defined by the model. Moreover, parallel computing is easily done due to local interactions of the particles. Nevertheless, computation time is important and turbulence modeling needs more complex formulations.

Another strategy was performed by Vendel (2011). The author considers that about 99% of the computation time of a CFD code is dedicated to wind field calculation. The remaining 1% is related to the dispersion model. In order to reduce the computation time on this 99%, and before the operational use phase, he suggests using accurate and detailed CFD simulations to create a wind field database. During the operating phase, the wind field is interpolated from the database according to the meteorological conditions recorded on the site. Then, the pollutant concentrations evolution is computed through a Lagrangian dispersion model. Advantages of the Vendel's approach are:

- Very short computing time, this method improves calculation time by a factor of 40
- Good accuracy, depending on the quality of CFD database calculation
- Easy to use, no convergence problems

Of course, there are some obvious disadvantages:

- Site-specific modeling because of the need of a preexisting database;
- Interpolation errors if the scenario is very different from database;
- No possibility for extrapolation outside the database wind conditions;
- Impossible to make a prediction in another geometrical configuration than the one considered in the database;
- CFD database constitution is long and burden: Vendel made 126 flow field calculations representing 10 GB of data during two months continuously.

Faced with these drawbacks, we propose to take the best of both strategies: using a statistical tool instead of solving Navier Stokes equations and create a rich database of CFD calculations in order to give knowledge of fluid mechanics to the statistical tool.

3. Machine learning tools

3.1. Artificial Neural Networks (ANN)

ANN are machine learning models. They are based on the systemic paradigm and are able to identify a nonlinear and dynamic behavior from a database without physical assumption. Thanks to two essential properties: first the universal approximation (Hornik et al., 1989), and second the parsimony (Barron, 1993) the Multilayer perceptron is able to predict efficiently future behaviors on never encountered situations within the variables range of the database. ANNs can be used in classification, in text recognition for example (LeCun et al., 1989). They can also be used to forecast physical phenomenon, presenting powerful models (Kong-A-Siou et al., 2011 & Toukourou et al., 2010). The information about the non-linear phenomenon to simulate or forecast must be provided using a database. As previously presented, ANNs act generally like a black-box: the physics cannot be extracted easily from the models.

Dreyfus (2005) provides a substantial background for the fundamentals on neural networks. In this study, multilayer perceptron is employed: it consists of a feedforward neural network with one hidden layer of N_{nl} non-linear (sigmoid) neurons and a linear output neuron, as shown in Fig. 2.

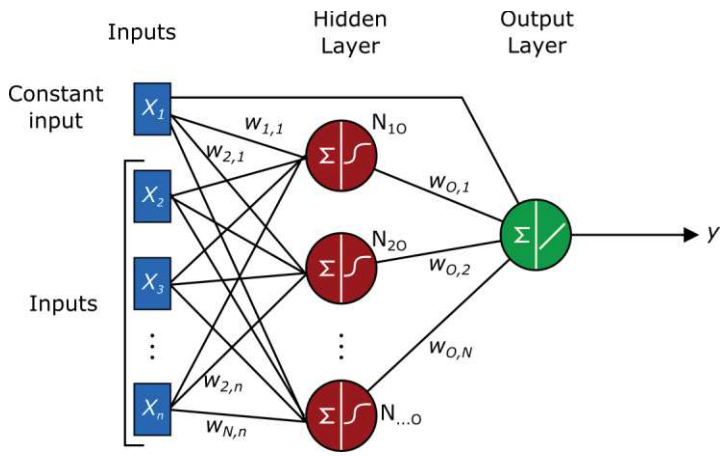

Figure 2: Multilayer perceptron scheme

Parameters $\boldsymbol{w}_{i,j}$ are calculated through a training procedure in order to minimize an error function, comparing y, the estimated value of the variable of interest and y^p, the value of the variable observed on the process. In this work, the least squares error is evaluated and the

parameters modification is done through the use of second order descent method like the Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994).

The function realized by the ANN was continuously tested on a disjoined set of examples, namely the stop set (usually improperly called the validation set). This last set was employed to avoid overfitting using early stopping (Sjöberg et al., 1995, see 4.2.3).

Lastly, performances of the model must be measured on another set, disjoint from previous ones, not used in training neither stopping neither model selection: the test set.

ANN have already been used in atmospheric dispersion for chronic pollution:

The tracer concentrations forecasting, in complex terrain, was made by training an ANN, using databases of values, coming from various sensors spatially distributed (Podnar et al., 2002). In this case, the output variable is the predicted concentration at a specific point. Cao (2007) gives an example of an ANN model forecasting the concentration distribution of non-buoyant aerosols released from transient point sources into the atmosphere. The database was built from a wide range of field experiments: wind velocities from 0.05 to 4.6 $m*s^{-1}$, temperatures from -22 to 28.4 °C, relative humidity from 29 to 100 % and insolation from 0 to 856 $W*m^{-2}$. The model focused on relative dispersion from the puff center. Two methods were developed. The first one deals with direct modeling by the ANN. Input variables are related to the source term (particle shape, initial size of the plume), environment variables (wind speed, solar elevation, air temperature, humidity, Monin-Obukhov length, turbulent kinetic energy, refractivity parameter) and variables relative to the output (dispersion time, relative distance). It results that the neural networks better performed estimations than Gaussian models COMBIC and Slade's puff about 12%. Over the different test cases, FAC2 (fraction of the concentrations predicted into a factor of two of the observed values) evaluated on ANN model is twice the value obtained with Gaussian models. Nevertheless, in this application, the ANN model overestimates the lower concentrations and underestimates the higher ones.

Cao et al. (2007) carried out a study to determine, by ANN method, the dispersion coefficients used in the Gaussian model. The objective was to set these coefficients using different variables (dispersion time, wind speed, solar irradiation, ground heat flux, temperature, humidity, air pressure) as inputs for the neural networks. Generally speaking, Gaussian puff models with dispersion coefficients from ANNs outperform COMBIC and a Gaussian puff model using Slade's parameterizations. Despite these good results, it seems difficult to apply this model for actual emergency situations because of the complexity to collect input data for the neural network.

Artificial Neural Networks have also been used in atmospheric dispersion in case of steady conditions. Previous works (Lauret et al., 2013) shown that using CFD database to learn how dispersion operates gives good results. The only inputs needed are the wind velocity, mass flow rate and location of the point to be evaluated. General behavior is correctly modeled, except for high gradients area where overestimations are noticed.

ANNs are also used in pollution forecasting. Pelliccioni et al. (2010) combine the contributions of Gaussian models to those of static neural networks. Having observed systematic errors of the Gaussian model under specific conditions, the developed method uses neural networks as a filter. The inputs of the neural network are atmospheric stability (mixing height, Monin-Obukhov length, wind speed, friction velocity), distance to emission and the output of a Gaussian model. Model performance is improved with the coupled Gaussian-ANN model compared to single Gaussian model. Direct forecasting of concentrations by ANN is possible on a daily basis. Lauret et al. (2016) developed a model used to forecast particles concentration at given stations from an emission source. This model is trained through meteorological and particles concentration database previously acquired. Mean day concentrations of particles at a station 300 m away from the emission area are forecasted. The global error is maintained under reasonable values despite unexpected peaks.

3.2. Cellular Automata

Cellular automata (CA) are tools used for modeling physical phenomena in discrete space-time coordinates. The impact of local interactions on the evolution of the phenomenon is an important feature that promotes the use of CA. From this statement, Wolfram (1983) designed systematic local rules to study different influences from direct neighborhood. Conclusions of his work were that from simple local rules, it was possible to observe very complex phenomena such as biological systems evolution or structure and patterns development in the growth of organisms. Author demonstrated that CA could emulate specific behavior of biological or physical phenomenon observed in real life. Itami (1994) conceptually formalized cellular automata, defining **Q**, as the global state of the system:

$$\boldsymbol{Q} = \langle \boldsymbol{S}, \boldsymbol{N}, T \rangle \quad (1)$$

Where **S** represents the discrete states accepted for the cellular automaton, **N** represents the neighborhood of cells providing input values for the transition rules, *T* is the transition rule.

Depending on the aim of the study, different neighborhoods can be set such as Moore (sharing at least one node) or Von Neumann (sharing at least one edge) type. The transition rule T defines how a cell updates his state from the current time step to the next one. The

transition rule updates synchronously the states of each cell at each time step. A large number of quantitative mathematical techniques can be used such in the field of Machine Learning: as Artificial Neural Networks (Almeida et al., 2008), genetic algorithms (Ak et al., 2013), self-organizing systems (Elmenreich and Fehérvári, 2011), Markov chain (Balzter et al., 1998), Monte Carlo simulations (Zio et al., 2006), fuzzy logic (Wu, 1998) to implement this transition function.

As in classical numerical simulations, the system is defined on a domain characterized by global dimensions and number of cells. Dimensions of the cells are determined depending on the phenomenon dynamics and the desired accuracy. Stability criterion need to be defined and respect during simulation.

Cellular Automata are also used in atmospheric dispersion. Marin et al. (2000), based on the work of Guariso and Maniezzo (1992), identified the various phenomena involved in atmospheric dispersion of chronic pollution in order to determine transition rules. Calibration parameters (combination of gravity, wind and mass diffusion components) were determined by using measurements from three petrochemical industries. Model is thus able to simulate qualitatively the behavior of dispersion of pollutants, underlining the importance of calibration parameter.

Sarkar and Abbasi (2006) performed a similar method based on the simplification of the advection-diffusion equation (ADE) used as a cellular automata rule. They developed a model to assess consequences of a loss of containment on an industrial site, considering the specific configuration (storages, buildings and type of the area). As for CFD models, the domain is meshed and each cell is influenced by its neighborhood, including the size, nature and position of elements found near the scene of the accident. The transition rule is based on the ADE and simplified by adding calibrated parameters. No convergence is required so this method is potentially faster than CFD models. These propositions are interesting; unfortunately, no comparison was made with respect to actual tests or numerical simulations.

In the following, statistical approximation capacities of Artificial Neural Network are combined with spatiotemporal representation capabilities of cellular automata. The main goal is to produce similar results as CFD associated with small computation time.

3.3. Transition rule based on Artificial Neural Networks

In the scope of urban land use change and urban growth, several models applied Artificial Neural Networks as transition rules of cellular automata (Li and Yeh 2002, Almeida et al., 2008). In the present work, the aim was to check relevance of coupled ANN and CA to predict concentrations for several study cases. The ANN ability to take over large amount of

data and to represent a specific behavior is engaged through the emulation of transient ADE equation for atmospheric dispersion process in 2D. This model was designed especially for emergency management or prediction situation. It has to be effective and time computation efficient, for near field concentration forecasting in complex terrain.

4. Method

In this study, Cellular Automata are used with an Artificial Neural Network based rule (CA-ANN). At each time step, the neural networks forecast the concentration at the next time step.

Usually modeling a dynamic system is done with models fed by state variables; for example past estimations or measurements of the predicted variable (Dreyfus, 2005). The model proposed herein is intended to simulate concentration of pollutant at a discrete time kT ($k \in N^+$, where T is the sampling period), or more simply at discrete time k. In denoting the estimated concentration value as $C(k)$, the other variables, for example wind velocity, as $\mathbf{v}(k)$, and the nonlinear function implemented by the neural network as $g_{NN}(.)$, the "neural" input-output model has been designed, based on (Nerrand et al., 1993), as follows:

$$C(k) = g_{NN}[C(k-1), v(k), v(k-1), v(k-2), \dots v(k-w+1)] \quad (1)$$

where w is the width of a sliding temporal window conveying the exogenous variables information. Its optimal value is to be determined according to the procedure described in the model selection section.

Such a model is dynamic, given its dependence on previous output value $C(k-1)$, which serves as a state input. It has generally been shown that better performance is obtained from a feedforward model, whose state information is provided by measured values (Artigue et al., 2012). The equation governing this type of model is expressed as follows:

$$C(k) = g_{NN}[C^p(k-1), v(k), v(k-1), v(k-2), \dots v(k-w+1)] \quad (2)$$

where $C^p(k)$ denotes the pollutant concentration measured at time k during the process.

One can remark that the feedforward model that is fed by measured state values is limited to one-step prediction (because it needs measured values) whereas the recurrent model is able to predict over an infinite time horizon as long as exogenous input variables are available. As measured values are not available for the future in case of atmospheric dispersion, the chosen model is the recurrent model, shown in the following figure.

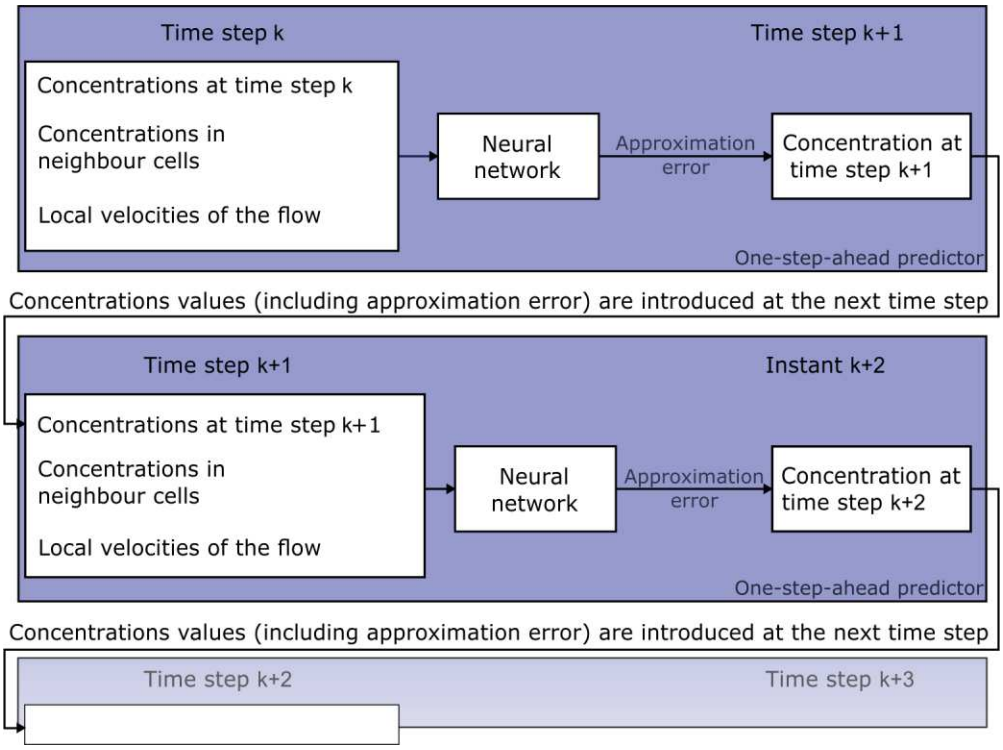

Figure 3: Process diagram of the recurrent neural network

Neural Network estimation introduces an error mainly due to imperfect modeling of the desired function. This error is propagated during the time steps through the feedback and may even lead to instability (divergence, oscillations).

As before discussed, the transition rule uses local variables to produce a forecast of the cellular automata state at the next time step. The CA-ANN is designed in following steps:

- Input selection of the ANN transition rule of the CA
- Create a database representing fully the phenomenon
- Train the ANN and optimize:
    - The sampling of the database
    - The architecture of the Neural Network
- Select the best model
- Assess the quality of the simulation in terms of accuracy and stability.

4.1. Inputs variable selection

Variable selection is a critical task in neural network definition because it reduces the complexity of the model and avoids overfitting (Geman et al., 1992). Variables can usually be selected by empirical methods (Kong A Siou et al., 2011). Nevertheless it is also possible to capitalize on expert knowledge when this information is available as in the present study. The advection-diffusion equation (3) helps to define neural network input variables:

$$\frac{\partial C}{\partial t} + U_x \frac{\partial C}{\partial x} + U_y \frac{\partial C}{\partial y} = D_t \frac{\partial^2 C}{\partial x^2} + D_t \frac{\partial^2 C}{\partial y^2} + S_c \quad (3)$$

Where U_x and U_y are, wind velocity in direction *i,j*; *x* and *y* are length in direction *i,j, t* is the time, *C* is the gas concentration, S_C is the emission source, D_t is the turbulent diffusion coefficient.

Considering a given cell *i,j*, concentration at present time step is $C_{i,j}^t$, both neighbors values of concentrations and flow velocities are used as inputs for the ANN transition rules using the discretization of the advection-diffusion equations terms in the following manner:

- Advection terms of the ADE gives four variables:
  - Velocities: $U_{x_{i,j}}$; $U_{y_{i,j}}$,
  - First derivatives of the concentration: $\frac{C_{i,j}^t - C_{i-1,j}^t}{\Delta x}$; $\frac{C_{i,j+1}^t - C_{i,j-1}^t}{2\Delta y}$
- Diffusion terms of the ADE gives two more variables:
  - Second derivatives of the concentration: $\frac{C_{i+1,j}^t - 2C_{i,j}^t + C_{i-1,j}^t}{\Delta x^2}$; $\frac{C_{i,j+1}^t - 2C_{i,j}^t + C_{i,j-1}^t}{\Delta y^2}$
- Transient term of the ADE gives the last input:
  - Initial concentration in the cell: $C_{i,j}^t$

The output of the transition rule is the concentration in the cell at the next time step: $C_{i,j}^{t+1}$. Variables are thus defined locally on a mesh and at each time-step.

As regard to the convergence, one necessary but not sufficient condition in CFD models is Courant Friedrich Lewy (C_{FL}) number value. It expresses the threshold over which divergence of calculation is observed. It appears if cells dimensions are less than the distance travelled by a particle animated with the fastest velocity of the phenomenon during one time-step:

$$\frac{u.\Delta t}{\Delta x} \leq C_{FL_{max}} \quad (4)$$

Setting the value of Courant number at 1, with a given wind velocity of 20 m.s-1 and a cell length equal to 0.2 meter, time-step has to be inferior or equal to 0.01 second. ANN requires the same setup in order to reach satisfying results in the training phase and thus, in the operating phase.

4.1.1.Setting up the CA-ANN

As mentioned in 3.2, a CA is fully defined by the states (***S***), (***N***) the neighborhood of each automata and (*T*) the transition rule. The domain and the boundary conditions of the CA need to be specified.

The domain is oriented with the wind direction and matches the CFD domain used to create the database. Boundary conditions are defined as follow:

- Left: zero inlet concentrations;
- Up and down: symmetry;
- Right: Neumann outlet boundary condition

4.1.2.Running the CA-ANN

Preliminary, the state of the CA, assumed as defined by the matrix of each output state of automata, namely **Q**, has to be initialized with concentration values. The wind velocity is considered as stationary. Then each automaton obeys to the following steps (figure 4):

1. Read its neighbors states: concentrations and wind values from neighbors cells;
2. Compute its inputs: the discrete first and second concentration derivatives;
3. Normalize input variables with a predefined relation;
4. Estimate output using the neural network;
5. Un-normalize concentrations;
6. Update its state
7. Concentrations estimated are used at the next time step as inputs

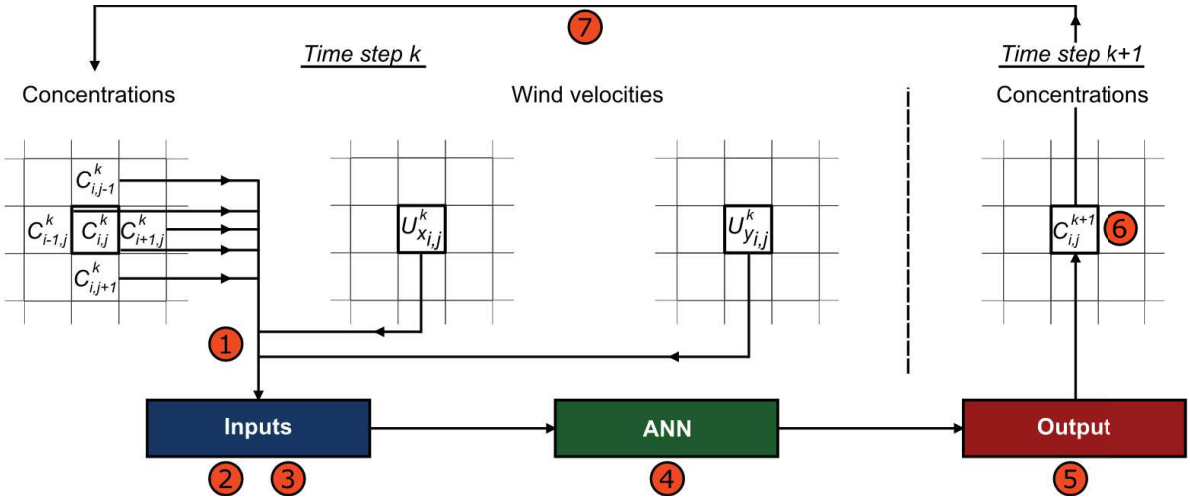

Figure 4: CA-ANN variables and process

The global state $\boldsymbol{Q}(k+1)$ is then updated in a parallel mode using the output of all automata. These steps are repeated until the desired duration time of the simulation is reached.

4.2. ANN Design

Database creation, training and optimization of the ANN are performed using the previous guidelines.

4.2.1.Database creation

The first step of designing a CA-ANN is getting trustful database. It is possible to get data from real size experiments, small scale physical simulations in wind tunnel or CFD modeling. As the database must contain numerous examples of various configurations, it is impossible to obtain in actual configurations, CFD calculations were thus used to build the database as they enabled to get numerous data on thin meshes in various scenarios. In this work, ANSYS Fluent 14 was used to simulate 2D dispersion of methane. The area of interest was defined as a 20 meters large and 30 meters long domain (figure 5). Since the case study was a horizontal 2D dispersion, no gravity effect of methane low density was taken into account. The CFD model was a classical RANS (Reynolds Averaged Navier Stokes equation) model with closure equations on turbulent kinetic energy *k* and its dissipation rate *ε*. It has to be reminded that the accuracy of such a model is not discussed here, since the objective of this study is to use CFD only to provide data considered as a virtual reality, in order to study CA-ANN adequacy to predict atmospheric dispersion. The mesh was composed of 240 000 nodes. Symmetry conditions were applied on both sides of the case. The methane puff was virtually introduced in the domain using velocity inlet boundary conditions. The virtual wind velocity is set using velocity inlet boundary condition too.

Figure 5 represents the case study used:

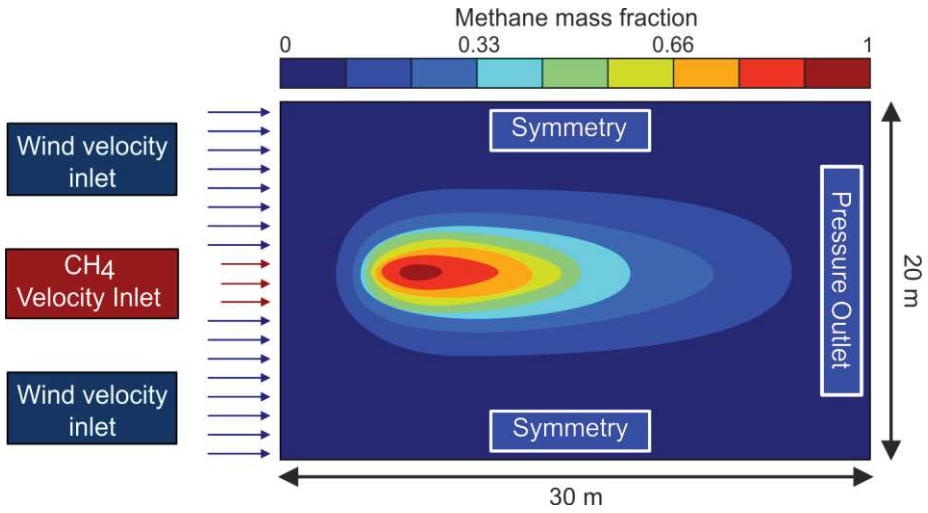

Figure 5: Sketch of methane dispersion in free field simulation by RANS k-ε realizable model

Constant mass of methane for a given mass fraction was injected in the domain with a corresponding duration depending on the wind velocity. Since the aim of the database creation was to be representative of the dynamic dispersion in time and space, the distribution concentration was stored at constant time steps. Model was intended to represent the methane injection in the domain during a short period and the following plume dispersion.

Total number of simulations is 95 decomposed in:

- 19 variations of the inlet velocities in the range of [2-20] $m.s^{-1}$ by step of 1 $m.s^{-1}$
- 5 variations of the initial methane fraction in the range of [0-100] volume percentage (%) by step of 20.

Each one of the 95 simulations is composed of 20 time steps, corresponding to those used in the application of the cellular automaton. Each simulation is stored when the simulation residual value is less than 10^{-5}. Concentration and wind field are computed at the center of each cell. Because CFD and CA-ANN mesh differ slightly, it is necessary to rearrange inputs value. Delaunay triangulation and cubic interpolation on CA-ANN mesh nodes are applied. The database, with values on the structured mesh, represents more than 28 000 000 cells. 32 hours of computation were required to build this database.

4.2.2.Stratified sampling

The initial database generated by CFD solver is sampled in order to diminish the number of examples to avoid useless redundancy and time consumption. The concentration distribution in the database is not homogenous: there are numerous points having a very small concentration value and few examples having a high concentration value. The key point of the study is high concentrations, since it is focused on accidental releases. It is thus important that high concentrations are well represented in the database.

This consideration needs to be considered while sampling. Therefore, a stratified sampling method was chosen to reduce the database size. Stratification considers a set of examples of size N_s divided in n_s strata. In the present work, examples are divided in n_s class of concentration with same step value. The strata are mutually exclusive: each concentration in the initial database must be assigned to only one stratum. The strata should also be collectively exhaustive: no concentration element can be excluded. Simple random sampling of E examples within each stratum is then applied to create the example database containing N_e examples.

4.2.3.ANN Architecture

The structure of the neural network corresponds to a classical two-layer perceptron (Figure 2). Input variables are linked to the neurons of the hidden layer. The output layer contains a unique linear neuron. To avoid over-influence of one specific variable and prevent sigmoid saturations, all variables values are centered and reduced between -0.9 and 0.9 except the concentration, reduced in the range [0 - 0.9] in order to avoid negative concentrations.

4.2.4. Complexity selection

Complexity selection consists in adjusting the best number of hidden neurons. It was done by cross validation (Stone, 1974) through variation of the number of hidden neurons, from 1 to 20. Using the training set of *D* subsets, each subset one at a time is reserved as the validation set. Training is then performed *D* times on *D* subsets. The mean quadratic error is thus calculated *D* times. To assess the model's generalization capability the cross-validation score is compared for the investigated configurations.

$$S_{cv} = \sqrt{\frac{1}{N_e}\sum_{i=1}^{D}\sum_{k\in i}\left(y_k^p - g(x_k, w_i)\right)^2} \quad (5)$$

Where N_e is the number of examples, *D* the number of subsets, y_k^p is the target value, $g(x_k, w_i)$ the value given by the model for a given set of ANN parameters w_i and inputs x_k for an example *k*. Early stopping is used to avoid overtraining: it consists in dividing the database in three parts: one set is the training set and represents 80% of the database. The stop set is used to avoid overtraining: when the mean squared error stops decreasing on it, the training phase is interrupted. The last set: test set is used to assess the model quality.

The determination of the initial parameters is known to influence results of training phase. Once the sampling and architecture are determined, 20 initializations are made in order to get the best model by cross-validation.

4.2.5. Model selection

The training algorithm used in this work is the Levenberg-Marquardt rule (Hagan and Menhaj, 1994). The training step can be optimized thanks to the sampling of the database while model selection (ANN architecture and initialization of parameters) is done by cross-validation (Dreyfus, 2005; Kong A Siou et al., 2012). Lower is the cross-validation score S_{cv}, better is the generalization of the neural network. Cross validation score is computed for each configuration and thus, the best model with the lower S_{CV} is selected.

Then, the training is done using the entire examples database and early stopping. At the end, the mean squared error and the coefficient of determination were computed on the test set assessing the generalization capabilities of the model.

4.3. Performance criteria used

To improve the performance of concentration forecasting evaluation, several criteria were proposed by Chang and Hanna (2004) for air quality. The present study proposes to use the following set of criteria: factor of two (*FAC2*), Normalized Mean Squared Error (*NMSE*), and Fractional Bias (*FB*). Because the coefficient of determination R^2 is widely used to evaluate

performance in the field of artificial neural networks, it replaces the correlation coefficient in the present study. The expression of R^2 is:

$$R^2 = 1 - \frac{\sum_i (y^p - y)^2}{\sum_i \left(y^p - \overline{y^p}\right)^2} \quad (6)$$

$\overline{y^p}$ correspond to the mean of the simulated concentration on the test set.

The target values for these criteria are as following: R^2 and *FAC2*=1; and *FB* and *NMSE*=0. *FB* measures systematic errors which lead to always underestimate or overestimate measured values. *FB* values ranges between -2 (extreme underprediction) to 2 (extreme overprediction). Therefore, matching perfect target *FB* value does not mean perfect modeling, because of possible cancelling errors. *NMSE* measures systematic and random errors. Acceptable values are within +/- 30% of the mean fractional bias ($|FB| < 0.3$), random scatter is about a factor of two to three of the mean ($NMSE < 1.5$), coefficient of determination is superior to 0.9, the factor of two is superior to 0.5. For this reason it is necessary to use simultaneously several criteria.

5. Results and discussion

The following paragraph proposes a discussion in terms of performance of the ANN training. Then, once the best model is selected, concentration forecasting capabilities of the CA-ANN are evaluated.

5.1. Training of the ANN

To select the best ANN model, the influence of important characteristics of the ANN is investigated:

- number of examples in the database,
- number of neurons in hidden layer,
- initialization

Three databases were generated by sampling with three pairs (n_s; *E*) providing respectively 18 060 examples, 125 300 examples and 418 160 examples. For each database the number of hidden neurons was varied from 1 to 20. Moreover, the computing time is less than 20 minutes for the small database, about 7 hours for the medium one and more than 21 hours for the large one. So, only one initialization is performed for the medium and the large database. 10 initializations were made with the 18 060 examples database. In the following, influence of the number of neurons in hidden layer, number of examples in the database and initialization is evaluated. Results are discussed through the S_{CV} value. Possibility to extent this method to real measurements is evaluated through the use of a noisy database.

5.1.1. Influence of the sampling process and the architecture

Considering the influence of the number of examples and the influence of the number of neurons in hidden layer, figure 7 gives important information. First, an increase of the complexity of the ANN (number of neurons in hidden layer) involves a continuous decrease of the Cross-Validation Score S_{cv} reflecting improvement of generalization ability. Using more than 20 neurons do not significantly improve model performances. This behavior is characteristic of a database without noise.

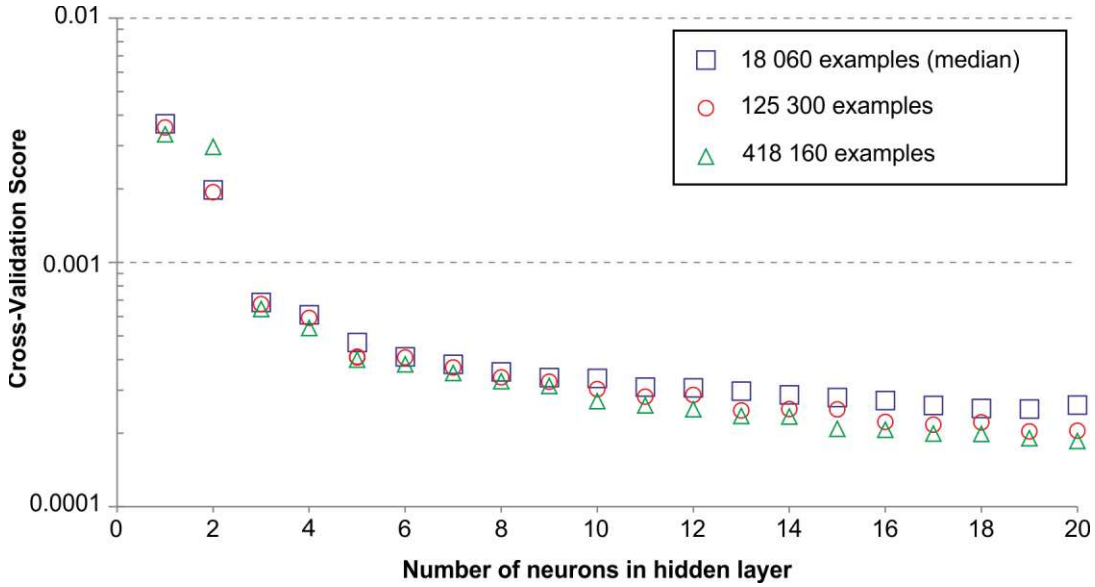

Figure 6: Cross validation score and number of neurons in hidden layer for three database dimensions

Secondly, training with databases having high number of examples provides better results than those obtained with lower number of examples. Obviously, there is a difference between medium and small database (blue and green symbols). The more representative of the phenomenon the data are, the better the ANN fits the data, even on new examples (test).

To evaluate the influence of parameters initialization, 10 initializations are done before training of the 18 060 examples database. Results of training for a 20 hidden neurons neural networks are reported on figure 8:

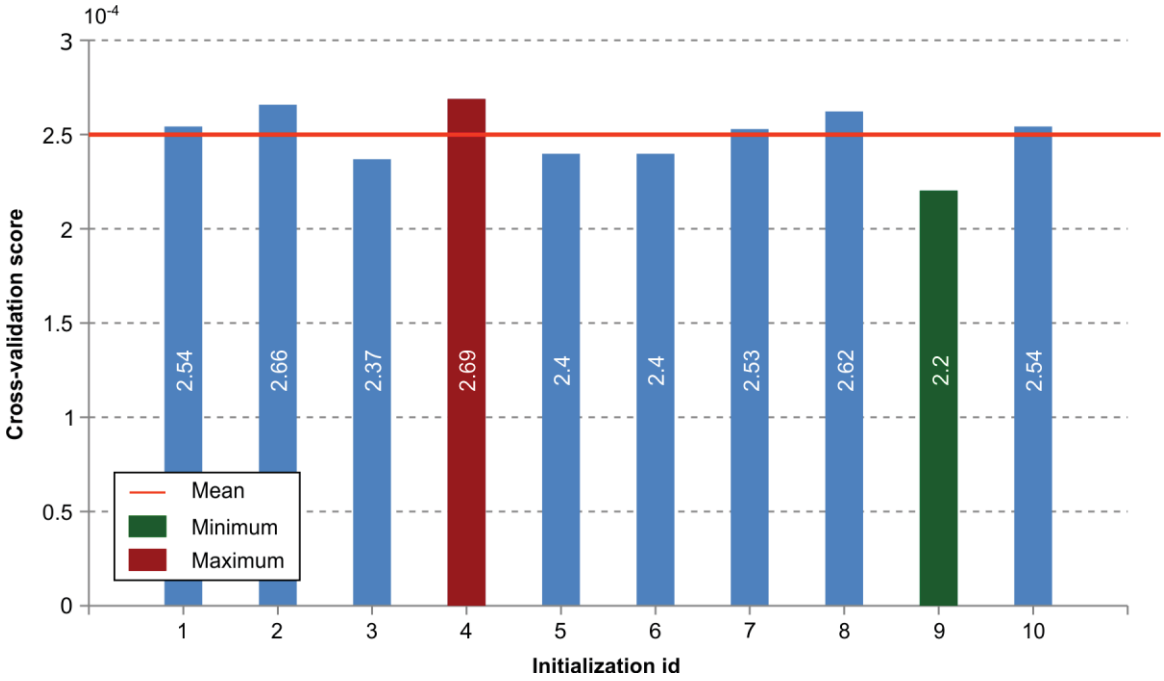

Figure 7: Variability of the cross validation score (18 060 examples) versus initialization

The minimum and maximum of the S_{cv} show that generalization performances are dependent of the initialization. These results sustain the need of testing several parameters vectors in order to select the best model once the complexity of the neural network is set. Nevertheless, S_{cv} values score are not widely scattered from the average. The standard deviation is less than 10% of the average value. Influence of initialization is not major and the value of 10 initializations will be retained.

Increase in generalization performance induced by large databases is not significant. Therefore, in order to evaluate correctly the limits of this method while keeping reasonable computation time, small database is used both in case of noiseless and noisy databases training.

Considering noiseless database, specifications hereafter are used in the following:

- 18 060 examples database (small);
- 20 neurons in the hidden layer;
- Selection of best initialization (from 10 different initializations).

5.1.2. Influence of a noisy database

In the case of the initial database, the only noise identified is the numerical noise due to interpolation of ADE on the mesh. As shown previously, it is negligible and does not disturb the training of neural network.

Nevertheless, considering the bias-variance tradeoff this would not be the case in a real environment with noisy data. For this reason it seems important to evaluate the sensitivity of the generalization capability of the used neural network to a more intense noise. In order to consider this eventuality, noisy databases were created from the initial noiseless database: a white noise was applied to the original data, with a Signal on Noise Ratio (S_{NR}) increasing from 10 to 30 dB:

$$S_{NR_{dB}} = 10log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \quad (7)$$

Comparison is made between models derived from the previous experiences, and from noisy databases of 18 060 examples. The S_{cv} are calculated and drawn in figure 8 versus the number of hidden neurons.

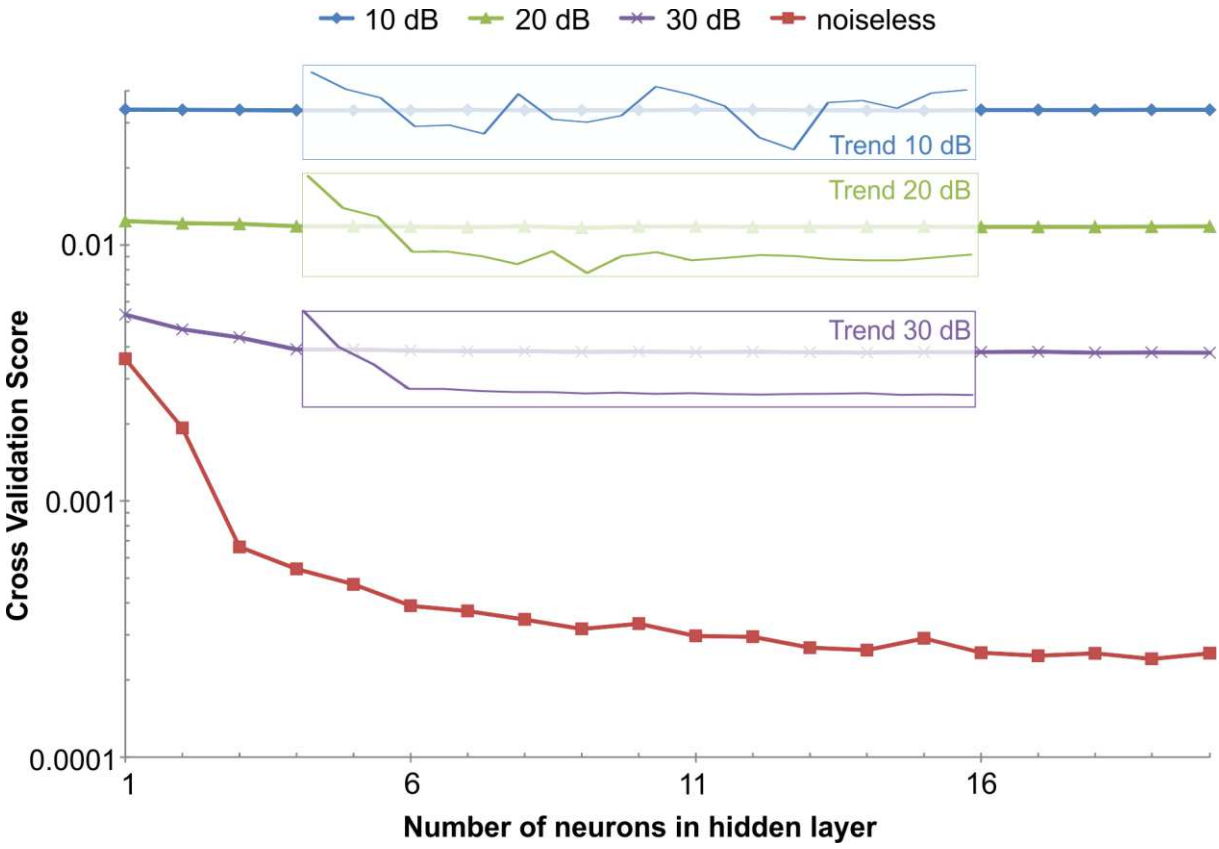

Figure 8 : Evolution of cross validation score versus the number of hidden neurons

Curves from noisy databases show a small decrease with the number of neurons until an asymptotic line is obtained; this means that the cross validation is efficient to prevent overfitting (Kong-A-Siou, 2012). Noisier database (S_{NR}=10dB) has the biggest cross validation score. It noticed the increase of hidden neurons does not correspond to a better performance: too much noise is harmful to the training. Noiseless database shows a constant diminution of the S_{cv}. The S_{CV} decreases until the magnitude of the noise is reached (Dreyfus, 2005). In cases of noisy database, this minimum is reached earlier.

In the following the performances related to the noisy and the noiseless databases will be provided. The best S_{cv} of each one gives the initial parameters vector to use. For the noiseless database, the best model configuration is:

- 18 060 examples database;
- 12 neurons hidden layer;
- Best initialization within 20 tries.

5.1.3.Final training

After the selection of the best model by cross-validation (best database sampling, number of hidden neurons and initialization), the most appropriate configuration is used to perform training on the whole training set. The network previously presented in table 1 was used to implement the transition rule of an automaton. When evaluated on the test set, obtained errors are 5.82×10^{-4} for the noiseless and 0.0113 for the 10dB S_{NR} noisy databases.

It is not easy to assess the quality of this result when the transition rule is applied in parallel automata synchronously, in an iterative way. This analysis is presented in the following section.

5.2. Cellular automata for concentration estimation

5.2.1.CA-ANN Evaluation

The model is evaluated in terms of convergence towards the state observed with the CFD simulation. The domain of cellular automata is composed of 15 251 identical automata, in a matrix of 101 cells in *y* direction and 151 in *x* direction. The transition rule is identical for all the automata and implemented by the previously designed neural network. 15 251 identical neural networks are thus run in parallel with different input variables (concentration and wind from neighbors).

To get a global estimation of the model performance, the coefficient of determination is calculated over the whole set of automata at each time step. In the following calculations, only predicted and CFD concentrations superior to 450 ppm are compared, in order to avoid errors made on concentration inferior to 1% of the lower explosive limit. Moreover, each criterion is calculated on values paired both in space and time.

Test cases generated to validate the CA-ANN are selected on the full range values: three different flow rates {3.2 $m.s^{-1}$, 10.2 $m.s^{-1}$, 18.8 $m.s^{-1}$} and three different initial mass fractions {0.26, 0.5, 0.89} as mentioned in table 2:

Table 1: Test cases characteristics

Id	1	2	3	4	5	6	7	8	9
Velocity ($m.s^{-1}$)	3.2			10.2			18.8		
Initial mass fraction	0.26	0.5	0.89	0.26	0.5	0.89	0.26	0.5	0.89

They are evaluated twice:

- At a chosen time: time step 46. This time step corresponds to 90% of the total mass remaining in the domain in cases with high velocity (18.8 $m.s^{-1}$). In other cases, mass is exiting the domain at a higher time step value.
- At a fixed value of total mass remaining in the domain for each case (approximatively the same distance from release). As enounced before, it corresponds to time step 46 in case of high velocity. For medium and low velocities, it corresponds respectively to time step 85 and 272.

If we consider the same time step (46), figure 9 represents values of R^2 and FAC2 for each test cases and two ANNs (noiseless and 30dB databases):

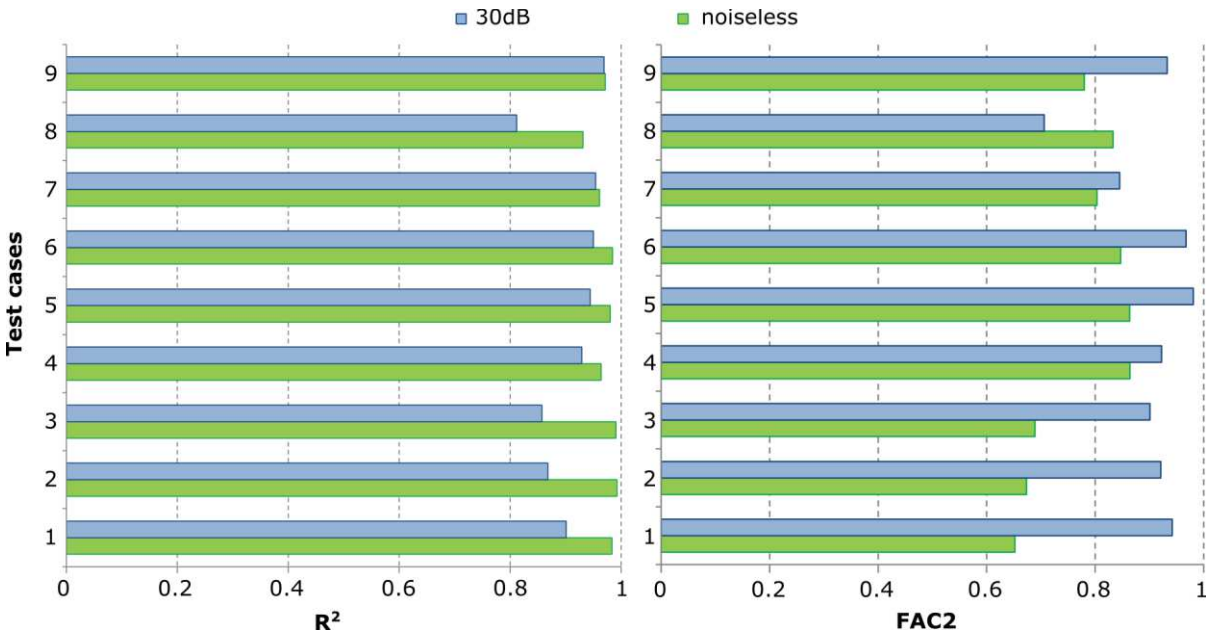

Figure 9: *R^2* and *FAC2* for test cases with two models: based on 30dB noisy database (blue) and on noiseless database (green)

R^2 and *FAC2* criteria show values respectively superior to 0.8 and 0.6 involving correct forecasting for all the test cases. Some variability is observed depending on the considered test case. For example, low wind velocities cases (1, 2 and 3) are well forecasted if we considered the *FAC2* on the noiseless trained CA-ANN.

When comparing noiseless and 30dB noisy trained CA-ANN, R^2 criterion shows a better performance for all cases for the noiseless training. FAC2 shows a different trend, with better values for noisy training. As said before, it is crucial to use several criteria to judge quality of a model. Fractional bias and normalized mean square error are thus shown in the following graph.

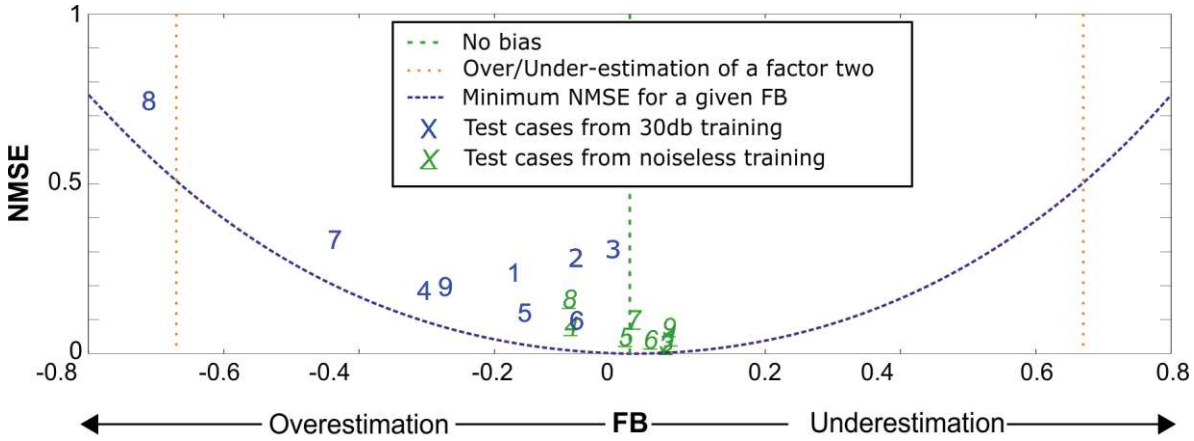

Figure 10: Noiseless and 30 dB trained models with systematic (FB) and random (NMSE) errors

Differences between the two models are significant when considering *NMSE* and *FB* criterion. Model based on noiseless training is less biased and random error is lower than 30db training based model. Test case 8 is the worst case for both models with an important overestimation of the concentrations. This figure illustrates the difficulty to model dispersion using CA-ANN from a noisy database.

Figure 11 is a scatter plot of test cases 8 and 2 for both 30 dB and noiseless model. Both test cases 8 and 2 based on noiseless model (A and B) have trend lines close to the perfect fitting. Nevertheless, one can point out that in A, lower concentrations are underestimated while higher concentrations are overestimated. In B, the standard deviation seems to be lower, with an important majority of examples close to the perfect fitting. When comparing test cases 8 and 2 based on 30 dB model (C and D), results are less fitted to CFD model. In C, the model has difficulties to forecast near zero concentrations. Again, an overestimation appears on the higher concentrations. D modeling shows similar deviation as A, with underestimation of lower concentrations and overestimation of higher concentrations. This analysis shows that the ANN has difficulties to correctly interpret the noise included in the training data while keeping good general forecasting trend. Evaluation of model based on noiseless training, A and B, shows the capacity of the model to maintain errors under an acceptable level.

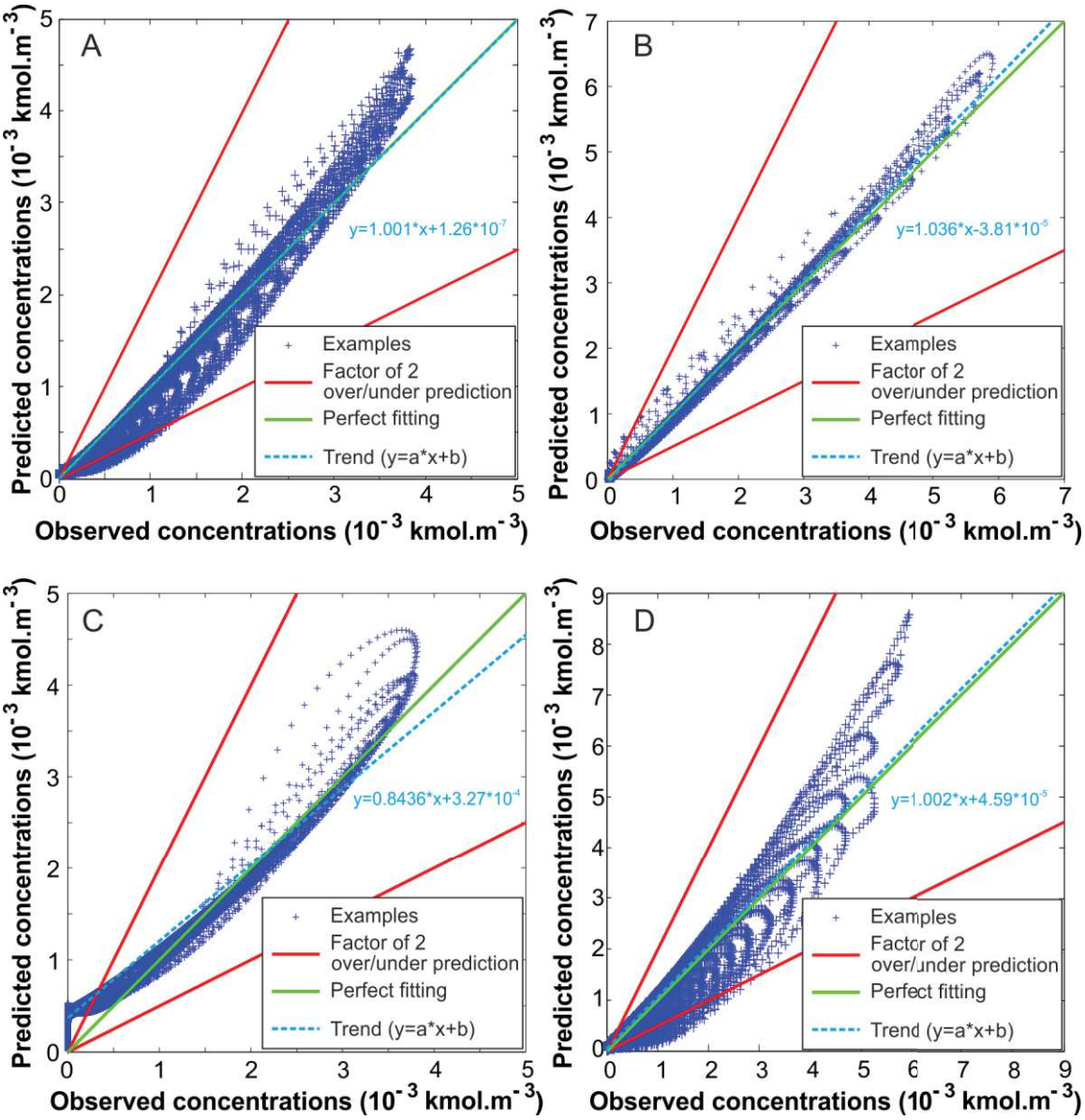

Figure 11: Scatter plot from: (A) test case 8 (noiseless model), (B) test case 2 (noiseless model), (C) test case 8 (30 dB model) and (D) test case 2 (30 dB model) – Red lines indicate factor two over/under estimation – Green line is perfect match.

In the following, model based on noiseless training only are considered. Test cases are compared at same distance (90% of initial mass still remaining into the domain) corresponding to different time steps: 46 for high velocity, 85 for medium velocity and 272 for low velocity. Figure 12 presents results for R^2 and *FAC2* for each test case. Except for low velocities cases, R^2 and *FAC2* are over 0.7, corresponding to well modeling.

Figure 12: R^2 and *FAC2* for test cases from noiseless training model

Considering systematic and random error confirms the trend on R^2 and *FAC2*. Low velocities cases are predicted worse as illustrated on figure 13:

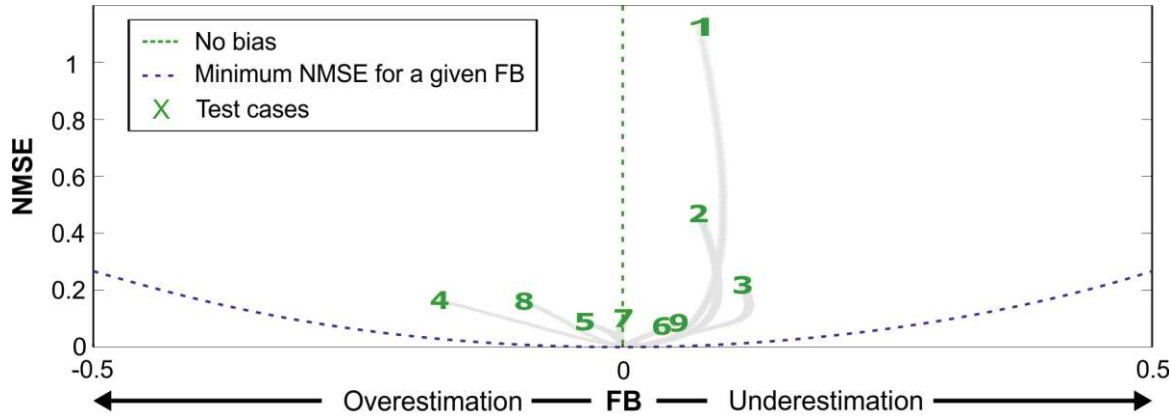

Figure 13: Systematic (*FB*) and random errors (*NMSE*) comparison between *CFD* and noiseless training based model for the nine test cases

Paths corresponding to values at previous time steps are indicated in grey. Medium and high velocities are better forecast. Nevertheless, trajectories seem to be similar to those observed for low wind velocities. Test cases 4, 5 and 8 show different behavior than other by overestimating concentrations from a global point of view. Moreover, it is observed that systematic error on cases 1, 2 and 3 reach a maximum early in the process. Clearly, the main result from these simulations is that error is increasing with time steps. There is no compensation of the errors. This can be observed on the different criteria on the figure 14:

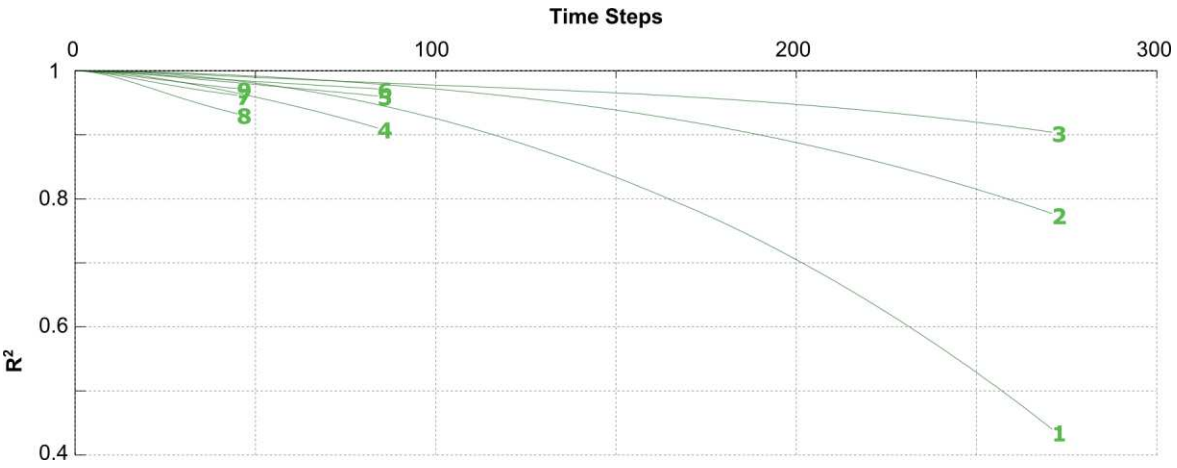

Figure 14: R^2 evolution with time steps for test cases

Values of the coefficient of determination slightly decrease for each test case as the number of time steps is increasing. The previous behavior is also identified here. It is noticed that cases with high initial mass fraction have greater values of R^2 through the time steps. However the main point is the decrease of performance while time increases.

This evolution is induced by the nature of the cellular automata computation. Even if the performance of the training is great, a little error is made for each cell at each time step. This forecast concentration is used as an input at the next time step. This error is propagated with no compensation as the number of time steps increases as illustrated on figure 14.

Otherwise, CA-ANN method does not take into account for mass balance. Considering total mass included in the domain, it is possible to compare modeled to observed data. Figure 15 shows that the error varies through the time steps.

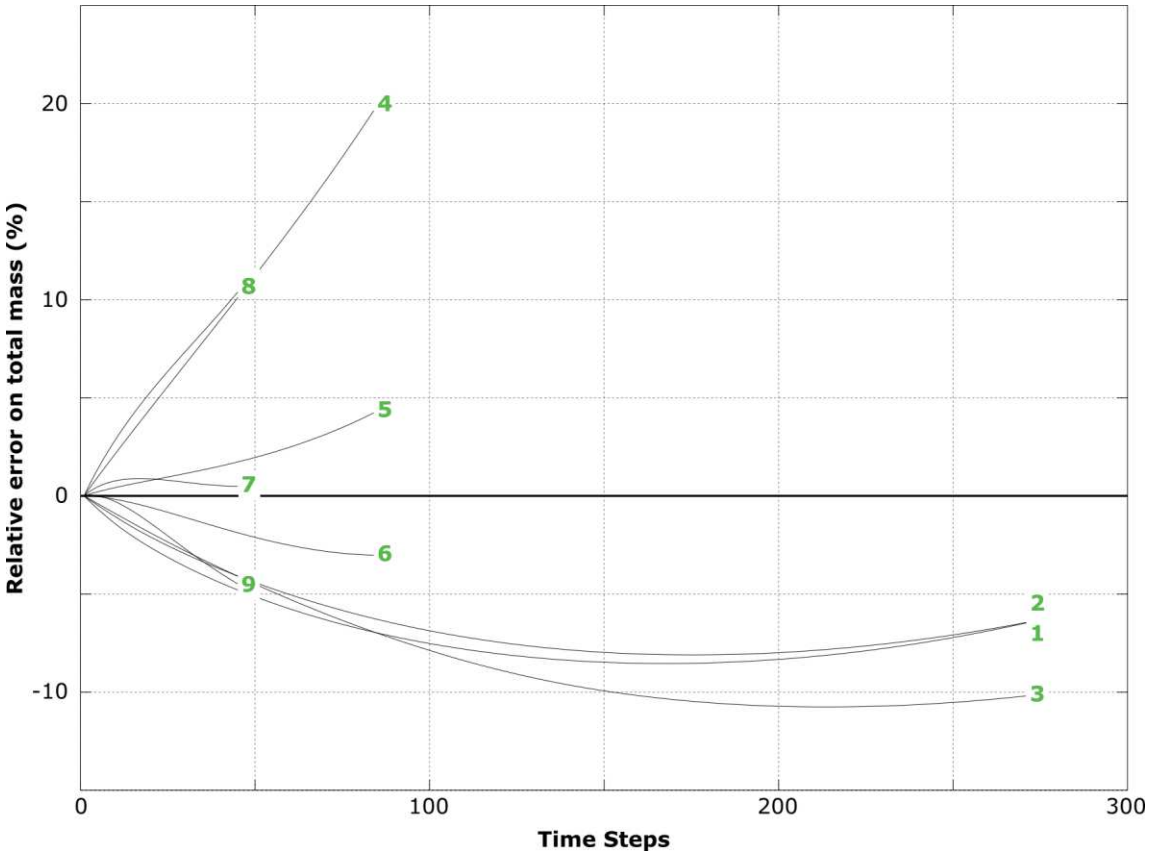

Figure 15: Relative error on total mass between observed and model data for nine test cases

It is possible to observe, as the time increases, the quality of criteria values slightly decreases except for cases 4 and 8, exceeding 10% on the relative error. This is a consequence of the model choice, with the use of recurrent neural network. A method to consider mass balance needs to be implemented to ensure respect of physical law.

CFD models have an implicit formulation for temporal discretization. Thus, it allows them to avoid calculation of the evolution of the plume for each time step. CA-ANN have an explicit formulation and each time step has to be calculated. Comparison between CFD and CA-ANN models based on the computing time can be therefore performed in two different manners:

- Compare the computation time for each time step used by the cellular automaton
- Compare the computation time for a set of dispersion duration.

In the first comparison, the CA-ANN computes the results in less than half a second. The CFD calculation is performed in about one minute for a time step. Improvement in computing time is more than 120. In this case, the use of CA-ANN is extremely interesting for operational situation assuming adjustments in spatial value of cells and thus in time step duration.

In the second comparison, improvement in computing time is less, since the CA-ANN must compute each time step, instead of a CFD model implicit temporal discretization. Thus, for the same domain and using the same workstation, CA-ANN model is 1.5 times faster than CFD calculation.

6. Conclusion

The model developed here is a combination of Cellular Automata controlled by an Artificial Neural Networks rule. Artificial Neural Networks were widely used as forecasting models while cellular automata were used to represent spatiotemporal phenomena. CFD simulations were computed and considered as “simulated reality”. Several levels of noise were added in the database in order to better reproduce what an actual database could be. As this work was innovative, several original steps were conducted for the model design: first the constitution of an optimized database by a specific sampling procedure and second the selection of the complexity thanks to noisy and noiseless databases. Comparison of this method to CFD cases using several criteria showed its great interest, either using noiseless or noisy data. Good agreement was demonstrated using classical air dispersion quality performance criteria. The coefficient of determination was over 0.7 for most cases. The model was better on high wind velocities cases. Low wind velocities cases were worse represented because of the increasing number of time-steps. Indeed, few estimation errors were made by the ANN rule at each time step. These errors were accumulated as the process go on. Method to reduce these errors has to be promoted, for example adapt the mesh dimension to wind velocity by using several CA-ANN or integrate a mass balance control during the training. Future work will focus on correcting the difference between expected and computed mass. Also, CA-ANN current model uses wind field as an input. To determine this global wind field, a model could be implemented, recovering flow perturbations behind an obstacle. The final goal could be providing model able to deal with atmospheric dispersion in an obstructed area. Cylindrical and spherical obstacles should be the shapes to consider, because of the hazardous materials stored in.

7. Software and data availability

CFD simulations database was realized on ANSYS FLUENT 14.0 (2011). Workstation used is a Dell Precision T7400 with Intel Xeon E5440 processor. The CA-ANN methodology was implemented with Matlab R2013 code for a total size of 236 Ko. CFD database is available in

Matlab format for a total size of 570 Mo at https://institutdessciencesdesrisques.wp.mines-telecom.fr/personnel/pierre-lauret/.

8. Acknowledgements

This research was supported by the French Alternative Energies and Atomic Energy Commission (CEA, Commissariat à l'énergie atomique et aux énergies alternatives).

9. Bibliography

Artigue, G., Johannet, A., Borrell, V., Pistre, S., 2012. Flash flood forecasting in poorly gauged basins using neural networks: case study of the Gardon de Mialet basin (southern France). Nat. Hazards Earth Syst. Sci., 12, 3307-3324, doi:10.5194/nhess-12-3307-2012

Ak, R., Li, Y., Vitelli, V., Zio, E., 2013. A Genetic Algorithm and Neural Network Technique for Predicting Wind Power under Uncertainty. Chem. Eng. Trans. 33, 925–930. doi:10.3303/CET1333155

Almeida, C.M., Gleriani, J.M., Castejon, E.F., Soares-Filhow, B.S., 2008. Using neural networks and cellular automata for modeling intra-urban land use dynamics. Int. J. Geogr. Inf. Sci. 22, 943–963.

Ansys Inc, 2011. ANSYS Fluent 14.0 user's guide.

Balzter, H., Braun, P.W., Köhler, W., 1998. Cellular automata models for vegetation dynamics. Ecol. Modell. 107, 113–125. doi:10.1016/S0304-3800(97)00202-0

Barad, M.L., 1958. Project Prairie Grass, A Field Program In Diffusion. Geophys. Res. Pap. 1, 299.

Barron, A.R., 1993. Universal approximation bounds for superpositions of a sigmoidal function. IEEE Trans. Inf. Theory 39, 930–945. doi:10.1109/18.256500

Brambilla, S., Totaro, R., Manca, D., 2010. Simulation of the LPG release, dispersion, and explosion in the Viareggio railway accident. Chem. Eng. Trans. 19, 195–200. doi:10.3303/CET1019032

Cao, X., 2007. Modelling the Concentration Distribution of Non-Buoyant Aerosols Released from Transient Point Sources into the Atmosphere. Queen's University, Ontario, Canada.

Caputo, M., Giménez, M., Schlamp, M., 2003. Intercomparison of atmospheric dispersion models. Atmos. Environ. 37, 2435–2449. doi:10.1016/S1352-2310(03)00201-2

Castellani, F., Astolfi, D., Burlando, M., Terzi, L., 2015. Numerical modelling for wind farm operational assessment in complex terrain. J. Wind Eng. Ind. Aerodyn. 147, 320–329. doi:10.1016/j.jweia.2015.07.016

Chang, J.C., Hanna, S.R., 2004. Air quality model performance evaluation. Meteorol. Atmos. Phys. 87, 167–196. doi:10.1007/s00703-003-0070-7

Chopard, B., Dupuis, A., Masselot, A., Luthi, P., 2002. Cellular automata and lattice boltzmann techniques: an approach to model and simulate complex systems. Adv. Complex Syst. 5, 1–144.

Dreyfus, G., 2005. Neural Networks. Springer-Verlag, Berlin/Heidelberg. doi:10.1007/3-540-28847-3

Elmenreich, W., Fehérvári, I., 2011. Evolving self-organizing cellular automata based on

neural network genotypes. Proc. 5th Int. Conf. Self-organizing Syst. 16–25. doi:10.1007/978-3-642-19167-1_2

Geman, S., Doursat, R., Bienenstock, E., 1992. Neural Networks and the Bias/Variance Dilemma. Neural Comput. 4, 1–58. doi:10.1162/neco.1992.4.1.1

Guariso, G., Maniezzo, V., 1992. Air quality simulation through cellular automata. Environ. Softw. 7, 131–141.

Hagan, M.T., Menhaj, M.B., 1994. Training Feedforward Networks with the Marquardt Algorithm. IEEE Transations Neural Networks 5, 989–993.

Hanna, S.R., Egan, B.A., Purdum, J., Wagler, J., 1999. Evaluation of the ADMS, AERMOD, and ISC3 dispersion models with the optex, duke forest, kincaid, indianapolis, and lovett field data sets. Int. J. Environ. Pollut. 3.

Hardy, J., Pomeau, Y., De Pazzis, O., 1973. Time evolution of two-dimensional model system. I. Invariant states and time correlation functions. J. Math. Phys. 14, 1746–1759.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer Feedforward Networks are Universal Approximators. Neural Networks 2, 359–366.

Itami, M., 1994. Simulating spatial dynamics : cellular automata theory. Landsc. Urban Plan. 30, 27–47.

Kong A Siou, L., Johannet, A., Borrell, V., Pistre, S., 2011. Complexity selection of a neural network model for karst flood forecasting: The case of the Lez Basin (southern France). J. Hydrol. 403, 367–380. doi:10.1016/j.jhydrol.2011.04.015

Kong A Siou, L., Johannet, A., Valérie, B.E., Pistre, S., 2012. Optimization of the generalization capability for rainfall–runoff modeling by neural networks: the case of the Lez aquifer (southern France). Environ. Earth Sci. 65, 2365–2375. doi:10.1007/s12665-011-1450-9

Launder, B.E., Spalding, D.B., 1974. The Numerical Computation of Turbulent Flows. Comput. Methods Appl. Mech. Eng. 3, 269–289.

Lauret, P., Heymes, F., Aprin, L., Johannet, A., Munier, L., Lapébie, E., 2013. Near Field Atmospheric Dispersion Modelling on an Industrial Site Using Neural Networks. Chem. Eng. Trans. 31, 151–156. doi:10.3303/CET1331026

Lauret, P., Perrin, M., Heymes, F., Aprin, L., Slangen, P., Pey, A., Steinkrauss, M., 2016. Atmospheric powder dispersion in an urban area 48.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., 1989. Backpropagation applied to handwritten Zip Code recognition. Neural Comput. 1, 541–551.

Li, X., Yeh, A.G.-O., 2002. Neural-network-based cellular automata for simulating multiple land use changes using GIS. Int. J. Geogr. Inf. Sci. 16, 323–343. doi:10.1080/13658810210137004

Marin, M., Rauch, V., Rojas-Molina, A., Lopez-Cajun, C.S., Herrera, A., Castano, V.M., 2000. Cellular automata simulation of dispersion of pollutants. Comput. Mater. Sci. 18, 132–140.

Nerrand, O., Roussel-Ragot, P., Personnaz, L., Dreyfus, G., Marcos, S., 1993. Neural networks and nonlinear adaptive filtering: Unifying concepts and new algorithms. Neural Computation 5, 165–199.

Pelliccioni, a., Gariazzo, C., Tirabassi, T., 2010. A neural net-air dispersion model validation study using the Indianapolis urban data set. Int. J. Environ. Pollut. 40, 70. doi:10.1504/IJEP.2010.030884

Podnar, D., Koračin, D., Panorska, A., 2002. Application of artificial neural networks to modeling the transport and dispersion of tracers in complex terrain. Atmos. Environ. 36,

561–570. doi:10.1016/S1352-2310(01)00446-0

Sarkar, C., Abbasi, S. a, 2006. Cellular automata-based forecasting of the impact of accidental fire and toxic dispersion in process industries. J. Hazard. Mater. 137, 8–30. doi:10.1016/j.jhazmat.2006.01.081

Sharan, M., Gopalakrishnan, S.G., 1997. Bhopal gas accident: a numerical simulation of the gas dispersion event. Environ. Model. Softw. 12, 135–141. doi:10.1016/S1364-8152(96)00054-0

Sjöberg, J., Zhang, Q., Ljung, L., Benveniste, A., Delyon, B., Glorennec, P.Y., Hjalmarsson, H., Juditsky, A., 1995. Nonlinear Black-box Modeling in System Identification : Unified Overview *. Automatica 31, 1691–1724.

Stone, M., 1974. Cross-Validatory Choice and Assessment of Statistical Predictions. J. R. Stat. Soc. Ser. B 36, 111–147.

Toukourou, M., Johannet, A., Dreyfus, G., Ayral, P.-A., 2011. Rainfall-runoff modeling of flash floods in the absence of rainfall forecasts: the case of “Cévenol flash floods.” Appl. Intell. 35, 178–189. doi:10.1007/s10489-010-0210-y

Vendel, F., 2011. Modélisation de la dispersion atmosphérique en présence d’obstacles complexes : application à l’étude de sites industriels. Ecole Centrale de Lyon.

Wolfram, S., 1983. Statistical mechanics of cellular automata. Rev. Mod. Phys. 55, 601–644.

Wu, F., 1998. Simulating urban encroachment on rural land with fuzzy-logic-controlled cellular automata in a geographical information system 53, 293–308. doi:10.1006/jema.1998.0195

Zio, E., Podofillini, L., Zille, V., 2006. A combination of Monte Carlo simulation and cellular automata for computing the availability of complex network systems. Reliab. Eng. Syst. Saf. 91, 181–190. doi:10.1016/j.ress.2004.12.002