Building and Environment 287 (2026) 113858

Contents lists available at ScienceDirect

Building and Environment

journal homepage: www.elsevier.com/locate/buildenv

Optimal control of ventilation in enclosed spaces using Adversarial Neural Networks

Donghu Guo [a,*], Claire E. Heaney [a,b], Boyang Chen [a], Jieyi Tang [c], Andrea Cammarano [d,e], Prashant Kumar [f], Christopher C. Pain [a,b,g]

[a] Applied Modelling and Computation Group, Department of Earth Science and Engineering, Imperial College London, South Kensington Campus, London, SW7 2AZ, UK
[b] Centre for AI-Physics Modelling, Imperial-X, Imperial College London, White City Campus, London, W12 7SL, UK
[c] Department of Earth Science and Engineering, Imperial College London, South Kensington Campus, London, SW7 2AZ, UK
[d] James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ, UK
[e] School of Engineering, University of Southampton, Southampton, SO16 7QF, UK
[f] Global Centre for Clean Air Research (GCARE), School of Sustainability, Civil and Environmental Engineering, University of Surrey, Guildford, GU2 7XH, Surrey, UK
[g] Data Assimilation Laboratory, Data Science Institute, Imperial College London, South Kensington Campus, London, SW7 2AZ, UK

ARTICLE INFO

Keywords:
Control
Data assimilation
Adversarial neural network
Indoor air quality
Computational fluid dynamics
Machine learning
Thermal comfort

ABSTRACT

Systems that manage and control air quality are the main energy consumers within a building and their design often relies on assumptions that lead to excessive energy usage. This paper proposes a new method based on Computational Fluid Dynamics (CFD) and Artificial Intelligence (AI) that can assimilate observations and control ventilation systems to achieve a certain goal, such as maintaining a particular temperature range in an indoor environment. An AI-based reduced-order model (ROM) is used here because current CFD methods generally have too large a computational expense for real-time control. We use proper orthogonal decomposition (POD) to represent the spatial distributions of the CFD simulations and learn the evolution of the POD coefficients in time with an Adversarial Neural Network. With this AI-based ROM, we can perform 4D Variational Data Assimilation (4D-Var DA) and control with a rapid workflow that incorporates the spatial variation of all the key variables: air flow velocity, CO_2, temperature, relative humidity and viral load. The proposed method is applied to three scenarios: (i) a ventilation scenario in which the aim is to keep the occupants healthy (indicated by ventilation and CO_2 levels) as well as thermally comfortable while minimising energy consumption; (ii) a pandemic scenario in which the priority is to keep people infection-free; (iii) a compromise between (i) and (ii). These three scenarios are developed within a classroom containing 26 children and one teacher, and are combined with an extensive set of measurement data, collected in the classroom of a primary school located in London. Our method successfully met the control objectives in all three scenarios.

1. Introduction

1.1. Background

According to a NHAPS study [1], people spend about 90% of their time in enclosed spaces. Maintaining a suitable temperature, an appropriate level of ventilation and good air quality is very important in such environments and there are calls for indoor air quality in public spaces to be regulated [2]. Furthermore, in the context of the COVID-19 pandemic, achieving a low viral load in indoor environments has become a key part of reducing the risk of exposure to SARS-CoV-2 [3], which is spread mainly by airborne transmission [4–7]. Poorly ventilated spaces are considered high risk [8], and precautionary advice during the pandemic was to over-ventilate buildings [9]. Whilst improving air quality by filtering out particles, so-called particle-filtering heating and cooling systems, such as Dyson fans, can also be used to help achieve thermal comfort [10]. However, such devices, as well as larger building ventilation systems, often have high energy consumption [11,12]. This is a particular issue in winter, when ventilation systems are the largest consumer of energy in cities [13]. In 2023, the IEA reported that operating buildings accounted for about 30% of global final energy

* Corresponding author.
E-mail address: donghu.guo21@imperial.ac.uk (D. Guo).

https://doi.org/10.1016/j.buildenv.2025.113858
Received 15 August 2025; Received in revised form 28 September 2025; Accepted 12 October 2025
Available online 21 October 2025

consumption and 27% of total energy sector emissions [14,15]. One analysis [16] has shown that full automation can save 50%–60% of the HVAC costs (heating, ventilation and air conditioning) of a city's energy consumption, another showed potential savings of 20–45% [17]. Therefore, the challenge is not just to ventilate, but to ventilate adequately, given the competing requirements of air quality, thermal comfort and energy usage. Machine learning has been taken up by some to enable buildings operate in a more "smart" and energy efficient manner. See Das et al. [18] for a general overview of how machine learning can be used in conjunction with traditional methods for this purpose. In this paper we explore the use of machine learning methods to control the air speed and temperature of a fan for optimal effect in three scenarios.

1.2. Related work

Combining observations from sensors with detailed, real-time (or faster) predictions of the indoor air flows and air quality [19] is one way of controlling the balance between ventilation requirements, air quality, thermal comfort and energy use. A range of methods can be used to produce predictions of the state of the indoor environment, such as single or multi-zonal methods [20–22], fast fluid dynamics (FFD) [23] and computational fluid dynamics (CFD), including Navier–Stokes approaches [24] and Lattice Boltzmann approaches [19,25]. Although zonal models and FFD methods are fast, these methods lack the high resolution and accuracy required for predictive control of an indoor environment [26,27], the flows of which often display irregular and difficult-to-explain behaviour [28]. CFD codes have been used successfully as a design tool using static analyses [24,29], however, the computational burden of CFD can be high when such codes are used to solve complex transient indoor flows [26,30]; an issue which is compounded for control, as many predictions are required associated with different values of the control variables.

To address this high computational cost, methods for data assimilation and control have been developed that are based on (1) using AI to accelerate optimisation; (2) surrogate models, many of which benefit from AI techniques; and (3) a combination of (1) and (2) [31]. One type of surrogate model is the reduced-order model [32], which combines dimensionality reduction techniques [33], applied to (CFD) simulation results, and interpolation, in order to model the evolution of the resulting lower-dimensional variables. At the interpolation stage, either neural networks or other techniques can be used for a data-driven approach, or for a projection-based approach, a Galerkin projection of the high-dimensional system onto the low-dimensional space can be performed. The projection-based approach has a stronger link to the discretised governing equations than the data-driven approach, however the latter can be faster and is simpler to construct (not requiring access to the CFD code, for example). Zerihun Desta et al. [34] developed a surrogate model based on a linear transfer matrix which mapped the temperature of the supplied air at inlets to the temperature at a number of sensors within a room. They were able to maintain a desired temperature range in a small room. Li et al. [35] used proper orthogonal decomposition (POD) and Galerkin projection to develop a reduced-order model that was able to maintain a desired temperature distribution within a small room. Oulghelou et al. [36] also developed a reduced-order model based on POD but with Riemannian barycentric interpolation. A genetic algorithm was used for the optimisation, which determined the inflow temperature corresponding to a given temperature distribution in a restricted area of a cavity. Ben Ayed et al. [37] developed a method of control for cooling air in a restaurant which considered temperature and moisture levels using a linear time-invariant surrogate model. Xue et al. [38] developed a genetic algorithm to accelerate the process of finding inlet conditions that would give a desired thermal comfort within an aircraft cabin. Morozova et al. [39] proposed a CFD-based surrogate for predicting comfort-related flow parameters given a number of sensor readings. Cao and Ren [40],Ren and Cao [41],Zhu et al. [42] developed a surrogate model, in which CFD simulations are volume averaged to create a low-dimensional linear ventilation model, which, in turn, is used to train a neural network. The location of the sensors was investigated [41], and with three sensors, energy consumption and CO_2 concentration were reduced within a room when compared with the same scenario without control [42]. Dmitrewski et al. [43] showed that a model for control based on reinforcement learning performed better when trained with CFD data and observations incorporated by data assimilation, being better able to cope with sudden changes to the temperature within the room. Zhuang et al. [44] model occupancy and control a ventilation system by using an autoencoder and a Bayesian Long Short-term Memory neural network (LSTM). In their work, the autoencoder is used to reduce redundant data thereby helping to alleviate model mis-specification when predicting unseen scenarios that are unlike the training data. Analyses of datasets of indoor environments [8,45–47] show that there is a complex relationship between thermal comfort, indoor air quality, outdoor conditions and energy consumption, highlighting the need for investigating control strategies for these indoor environments.

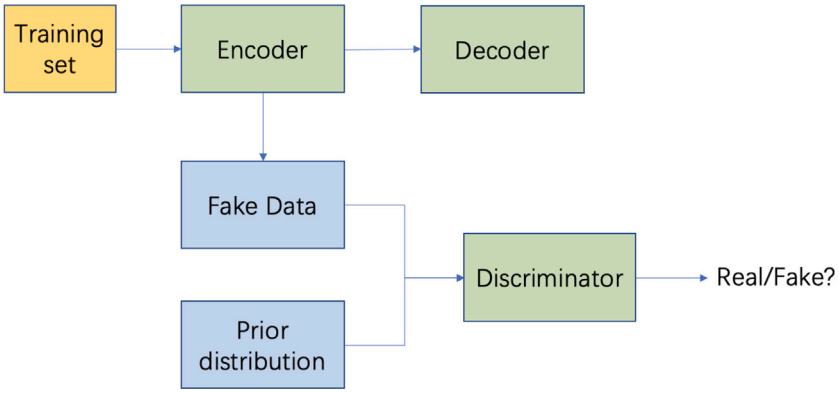
The AI-based reduced-order model presented in this paper has been designed specifically to assimilate data and perform control in a computationally efficient, integrated framework. High-resolution 3D time-dependent CFD simulations provide the training data, which is compressed into a lower-dimensional space using POD [33, chap. 3], and an adversarial autoencoder (AAE) is used to learn the evolution of the variables in the lower-dimensional space. The AAE, with inputs and outputs of POD coefficients (representing state variables), sensor values and control variables, results in a very flexible approach to data assimilation and control which we expand on later. We extend work done by Heaney et al. [48] on using AAEs for prediction and DA and propose a new method for 4D variational data assimilation optimised by backpropagation within a short time window (we refer to this modified DA as 4D-Var DA in this paper). Using the AAE we can rapidly predict the future changes of the indoor air indicators such as temperature, CO_2 concentration and virus load in the air, and quickly determine what the near future conditions will be in the room. We control these indicators in order to meet our near future (e.g. minutes) ventilation requirements such as thermal comfort and air quality. The control of the air quality, energy consumption and comfort indicators is achieved by optimising the ventilation system settings through the AI model. We apply this method to a primary school classroom for which we have a comprehensive set of observations [49].

1.3. Contribution

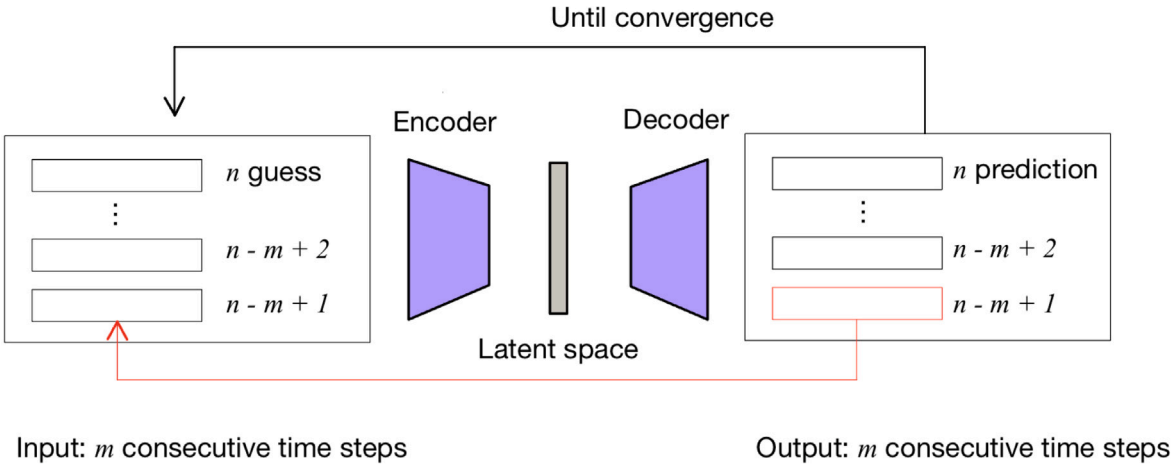
The contribution of this paper is to introduce a flexible AI-based framework that can be used for prediction, data assimilation and control, demonstrated here for an indoor environment. The versatility of the method comes from the architecture of the autoencoder and the choice of input and output variables, which include POD coefficients (representing state variables), sensor observations and control variables. After training, the user can choose whether to make predictions at the sensor locations or to assimilate data at the sensor locations, meaning that if for any reason, sensor observations are missing, the framework still operates without the need to re-train the neural network. Similarly, the user can also decide how many of the control variables to constrain. Again, this decision does not affect the operation of the framework or require the network to be retrained. Another advantage of this framework is that interpolation is not needed to calculate the mismatch between sparse spatial observations or control variables and highly resolved CFD results, as the observations and control variables are predicted directly by the autoencoder.

2. Methodology

In this section, we first present a reduced-order model which can be used for prediction, including a description of proper orthogonal



(a) Training process of the AAE. The encoder and decoder are trained jointly for encoding and reconstruction, while the encoder and discriminator are trained adversarially to regularise the latent space.



(b) Prediction strategy to obtain the solution at time level n using the AAE. Both the input of the encoder and the output of the decoder contain the same m consecutive time steps. The solution at time level n is updated through an iterative process, starting from an initial guess set equal to the values at time level $n-1$, until a convergence is reached.

Fig. 1. Schematic of the training and prediction processes of PredAAE: (a) the training process. (b) the iterative prediction strategy in time.

decomposition (POD) and the adversarial autoencoder (AAE). Next, we introduce the modifications necessary in order to perform both prediction and data assimilation, with two methods for data assimilation: 4D-Var DA with backpropagation and a constrained data assimilation method. Then we propose a ROM that can be used for all the three purposes of prediction, data assimilation and control. Finally, we highlight the flexibility of this AAE-based approach.

2.1. Prediction

2.1.1. Proper orthogonal decomposition

In order to construct a rapid-running method for prediction, data assimilation and control, which is also capable of generating highly resolved spatial fields needed for accuracy, a reduced-order model is proposed. This ROM is made up of two parts, a method of compression to reduce the number of variables, and a method to learn the evolution of the reduced variables in time. For the former, we chose POD to reduce the dimension of data while minimising the resulting loss of information. We use POD to compress the 3D spatial solutions (known as snapshots) of seven fields: velocity (three fields), temperature, CO_2 concentration, relative humidity and viral load. The primary purpose of POD is to decompose physical fields by applying an orthogonal decomposition to a set of solutions gathered over a certain time period. After normalising the snapshots from each of the seven fields, they are combined to form a matrix that has dimension $N^{\text{Fields}} N^{\text{Nodes}}$ by M, where N^{Fields} is the number of fields, N^{Nodes} is the number of nodes and M is the number of snapshots used in building the surrogate model. We decompose the snapshots matrix $\boldsymbol{\Phi}$ using singular value decomposition

$$\left[\boldsymbol{\phi}^1\ \boldsymbol{\phi}^2\ \cdots\ \boldsymbol{\phi}^M\right] =: \boldsymbol{\Phi} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \tag{1}$$

where $\boldsymbol{\phi}^k \in \mathbb{R}^{N^{\text{Fields}} N^{\text{Nodes}}}$ is a column vector containing the normalised solution fields at time level k, $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices containing the left and right singular vectors respectively, and the singular values $\{\sigma_i\}_{i=1}^{M}$ are found on the diagonal of the matrix $\boldsymbol{\Sigma}$. The amount of information captured when using N^{POD} of the M basis functions is given by the following empirical formula

$$\frac{\sum_{i=1}^{N^{\text{POD}}} \sigma_i^2}{\sum_{i=1}^{M} \sigma_i^2}. \tag{2}$$

After setting a value for the amount of information to be captured, N^{POD} can be determined from Eq. (2), and the basis functions are taken as the first N^{POD} columns of $\boldsymbol{U}$ and stored in the matrix $\boldsymbol{R} \in \mathbb{R}^{N^{\text{Fields}} N^{\text{Nodes}} \times N^{\text{POD}}}$. We use M of the total N^s snapshots (where $N^{\text{POD}} \ll M < N^s$) to find the low-dimensional space and we use the same set of snapshots to find the mappings used when normalising the CFD data. The remaining $N^s - M$ snapshots are used as unseen or test data. To project every snapshot onto the low-dimensional space obtained by POD, the following mapping is used:

$$\boldsymbol{\alpha}^k = \boldsymbol{R}^T \boldsymbol{\phi}^k \qquad \forall k\,, \tag{3}$$

where $\boldsymbol{\alpha}^k$ is a vector containing the POD coefficients associated with time level k. When the POD coefficients for the training dataset are obtained, each POD coefficient is normalised, so that, over time, the values lie in the range [0,1]. This is considered good practice for training neural networks to prevent one feature from dominating another due to a difference in magnitude of their values. A more thorough description of the theory of POD and the optimality of the basis functions can be found in Holmes et al. [33].

2.1.2. Predictive adversarial autoencoder

Introduced by Makhzani et al. [50], the adversarial autoencoder (AAE) consists of an autoencoder and a discriminator, with the encoder defining the aggregate posterior distribution over the latent vector (or latent variables) of the autoencoder. The core principle underlying the training of the AAE is to make the encoder and discriminator compete against each other in order to improve the ability of the autoencoder to perform an identity map, whilst also encouraging the latent vector to conform to a predefined distribution. This property has been shown to reduce the likelihood of gaps appearing in the latent space, which, in turn, helps the AAE to interpolate more effectively [50]. The AAE has been shown to perform better than other convolutional neural networks when learning low dimensional representations in urban flow applications [51] and when estimating the conductivity of the subsurface [52]. The input and output of the AAE have the same form, and, in this case, consist of a set of POD coefficients at m consecutive time levels. We represent the input associated with time level n (and the previous $m-1$ time levels) as $\boldsymbol{X}^n$:

$$\boldsymbol{X}^n = \begin{pmatrix} (\boldsymbol{\alpha}^n)^T \\ (\boldsymbol{\alpha}^{n-1})^T \\ \vdots \\ (\boldsymbol{\alpha}^{n-m+2})^T \\ (\boldsymbol{\alpha}^{n-m+1})^T \end{pmatrix}, \tag{4}$$

where $(\boldsymbol{\alpha}^n)^T = \left[\alpha_1^n, \alpha_2^n, \ldots, \alpha_{N^{\text{POD}}}^n\right]$, α_i^n represents the ith POD coefficient at time level n. The dimension of $\boldsymbol{X}^n$ is m by N^{POD} The output of the AAE is represented by $\tilde{\boldsymbol{X}}^n$.

The AAE first encodes $\boldsymbol{X}^n$ through the encoder into $\boldsymbol{z}^n$ (the latent vector) and then decodes $\boldsymbol{z}^n$ into the output $\tilde{\boldsymbol{X}}^n$ through the decoder. This can be written as

$$\boldsymbol{z}^n = Encoder(\boldsymbol{X}^n) \tag{5}$$

$$\tilde{\boldsymbol{X}}^n = Decoder(\boldsymbol{z}^n)\,, \tag{6}$$

$$\tilde{\boldsymbol{X}}^n = AAE(\boldsymbol{X}^n) = Decoder(Encoder(\boldsymbol{X}^n))\,. \tag{7}$$

Eqs. (8) show the training losses of the individual components of the AAE [50]:

$$\begin{aligned} L_D &= -\mathbb{E}_{z\sim p_{\text{data}}(z)}\left[\log(D(\boldsymbol{z}))\right] - \mathbb{E}_{X\sim p_X(X)}\left[\log(1 - D(Encoder(\boldsymbol{X})))\right], \\ L_E &= \mathbb{E}_{X\sim p_X(X)}\left[\log(1 - D(Encoder(\boldsymbol{X})))\right], \\ L_{AAE} &= \mathbb{E}_{X\sim p_X(X)}\left[\|AAE(\boldsymbol{X}) - \boldsymbol{X}\|^2\right], \end{aligned} \tag{8}$$

where AAE and D represent the autoencoder and the discriminator of the AAE, respectively. The decoder appears in these loss equations through the autoencoder. The structure of the AAE during training can be seen in Fig. 1(a). Minimising the losses in Eqs. (8) will enable an AAE to reconstruct the input variables, as closely as training will allow, whilst also forcing the latent variables to have a predefined distribution.

In order to make predictions for a time series with a trained AAE, we use the PredAAE algorithm, which was introduced in Heaney et al. [48] and is similar to algorithms used previously for predicting time series with adversarial networks [53–56]. To make a prediction for the POD coefficients at time level n ($\boldsymbol{\alpha}^n$), the POD coefficients of the previous $m-1$ time levels are required. We also require an initial guess or approximation for $\boldsymbol{\alpha}^n$. In this case we use $\boldsymbol{\alpha}^n \approx \boldsymbol{\alpha}^{n-1}$. The m POD coefficients are then fed into the AAE, whereupon the output yields a better approximation for $\boldsymbol{\alpha}^n$. The improved approximation is fed back into the AAE with the previous inputs remaining unchanged and this iteration process continues until the solution for $\boldsymbol{\alpha}^n$ does not change to within a given tolerance. This iterative process is depicted in Fig. 1(b). Once convergence has been reached at this time level, the time level and associated time window is incremented by 1, so that the inputs to the AAE for the next time level are $m-m+2, n-m+3, \ldots, n+1$. The process continues until we have solutions for all the desired time levels as described in Algorithm 1, first introduced in [48]. Initially, the algorithm requires $m-1$ solutions (at t^1, t^2, …, t^{m-1}) as 'initial conditions'. These could be taken from CFD results which are mapped onto the low-dimensional space. Once predicting for the POD coefficients at the $(2m-1)$th time level, the algorithm is uses POD coefficients that have been predicted by the AAE (and none from the CFD results).

Algorithm 1 Prediction in time with PredAAE [48]. By constraining the values of the POD coefficients at the previous (known) time levels, we can approximate POD coefficients at the future time level. The quantity represented by $||\cdot||_2$ is the L_2 norm, (the square root of the mean square error).

1: Given the maximum number of iterations (N^{its}); the trained AAE (f^{AAE}); a tolerance for the stopping criterion (ε); and $m-1$ previous solutions ($\boldsymbol{\alpha}^{n-m+1}$, $\boldsymbol{\alpha}^{n-m+2}$, …, $\boldsymbol{\alpha}^{n-1}$), this algorithm will make predictions for the POD coefficients at future time levels ($\boldsymbol{\alpha}^{n-1}$).
2: **for** time level $n \in$ desired range of time levels **do**
3: *!! approximate the solution at time level n*
4: $\boldsymbol{\alpha}^{n,0} = \boldsymbol{\alpha}^{n-1}$
5: **for** iteration $i = 1, 2, \ldots, N^{its}$ **do**
6:

$$\begin{pmatrix} \tilde{\boldsymbol{\alpha}}^{n,i-1} \\ \tilde{\boldsymbol{\alpha}}^{n-1} \\ \vdots \\ \tilde{\boldsymbol{\alpha}}^{n-m+1} \end{pmatrix} = f^{\text{AAE}} \begin{pmatrix} \boldsymbol{\alpha}^{n,i-1} \\ \boldsymbol{\alpha}^{n-1} \\ \vdots \\ \boldsymbol{\alpha}^{n-m+1} \end{pmatrix} \tag{9}$$

7: $\boldsymbol{\alpha}^{n,i} = \tilde{\boldsymbol{\alpha}}^{n,i-1}$
8: **if** $\left(||\tilde{\boldsymbol{\alpha}}^{n,i-1} - \boldsymbol{\alpha}^{n,i-1}||_2^2 < \varepsilon \text{ or } i == N^{its}\right)$ **then**
9: $\boldsymbol{\alpha}^n = \boldsymbol{\alpha}^{n,i}$
10: exit iteration loop
11: **end if**
12: **end for**
13: **end for**

2.2. Data assimilation

2.2.1. 4D-Var data assimilation with backpropagation

In data assimilation, model predictions and observations are combined, resulting in an improved prediction of the state of the system. Using the backpropagation algorithm of neural networks to perform 4D-Var DA is becoming an increasingly popular approach [57–60].

In practice, observations tend to be sparse, as typically, only a limited number of sensors are placed in the domain. To calculate the mismatch between sparse observations and high-resolution spatial fields from CFD, one approach is to interpolate the observations onto the spatial domain used in the CFD simulations [61], and the observations and CFD results can either be compared in the physical space or in low-dimensional space by using the same mapping for observations and CFD data. However, this approach introduces errors as a consequence of the interpolation. In this paper, leveraging the flexibility of neural networks as the surrogate model, we are able to introduce the sparse sensor values explicitly in the input and output of the AAE. This enables the model to predict POD coefficients in a low-dimensional space and sensor values in physical space simultaneously, eliminating the need for interpolation. For data assimilation, the output of the AAE can be written as follows:

$$\boldsymbol{X}^{n}_{4dvar} = \begin{pmatrix} (\boldsymbol{\alpha}^{n})^{T}, (\boldsymbol{s}^{n})^{T} \\ (\boldsymbol{\alpha}^{n-1})^{T}, (\boldsymbol{s}^{n-1})^{T} \\ \vdots \\ (\boldsymbol{\alpha}^{n-m+2})^{T}, (\boldsymbol{s}^{n-m+2})^{T} \\ (\boldsymbol{\alpha}^{n-m+1})^{T}, (\boldsymbol{s}^{n-m+1})^{T} \end{pmatrix}, \quad \tilde{\boldsymbol{X}}^{n}_{4dvar} = AAE\left(\boldsymbol{X}^{n}_{4dvar}\right), \tag{10}$$

where $(\boldsymbol{s}^{n})^{T} = \left[s^{n}_{1}, s^{n}_{2}, \ldots, s^{n}_{N^{\text{Sensors}}}\right]$, s^{n}_{i} represents the ith piece of sensor data at time level n; N^{Sensors} represents the total number of observations collected by the sensors (this number is determined by the number of fields measured multiplied by the number of sensor positions placed in the enclosed space); and $\boldsymbol{X}^{n}_{4dvar}$ and $\tilde{\boldsymbol{X}}^{n}_{4dvar}$ represent the input and output of the AAE respectively. With this specific input and output structure, we can calculate the data mismatch directly, using the model prediction of the sensor data $\tilde{\boldsymbol{s}}^{k}$ and the true observations $\boldsymbol{s}^{k}$, written as:

$$\mathcal{L}_{DA}\left(\boldsymbol{X}^{n}_{4dvar}, \tilde{\boldsymbol{X}}^{n}_{4dvar}\right) = \sum_{k=n-m+1}^{n} \left(\tilde{\boldsymbol{s}}^{k} - \boldsymbol{s}^{k}\right)^{T} \boldsymbol{W}_{s} \left(\tilde{\boldsymbol{s}}^{k} - \boldsymbol{s}^{k}\right) \tag{11}$$

where $\boldsymbol{W}_{s}$ is a weight matrix which controls the relative importance of the sensor values. Here, $\boldsymbol{W}_{s}$ is taken to be the identity matrix, giving the same weight or importance to each observation as we have no information to the contrary. For a single time level n, after calculating the data mismatch between the prediction of the model and the actual observation based on Eq. (11), the data mismatch is minimised by the backpropagation functions which calculate the gradient of the mismatch efficiently using automatic differentiation [62–64]. The whole data assimilation process is described in detail in Algorithm 2.

2.2.2. Constrained data assimilation

Introduced in Heaney et al. [48], constrained data assimilation is an alternative method of implementing DA. This approach fixes certain inputs of a trained AAE, generates an output for variables which are not fixed, and feeds these back into the input, iterating until the values do not change by more than a tolerance. This is similar to how prediction is achieved (described in Section 2.1.2) and will be used to compare our new DA approach (described in Section 2.2.1). For convenience, we recapitulate some of the details here, but for a more in depth discussion, see [48]. If the m sensor values are fixed throughout the time window and the POD coefficients at the previous $m-1$ time levels are fixed, one can iterate by repeatedly feeding the solutions and sensor values through the AAE, changing only the solution at the future time level, $\boldsymbol{\alpha}^{n}$. Once the solution at time level n has converged, one marches forward in time as described in Algorithm 3. Although in some cases, this algorithm achieves a better result than the method based on backpropagation, the data mismatch is not used directly. The data mismatch can be monitored and used to assess convergence, as done in [48]. Here, we use this approach purely for the purpose of comparison: it is not used for control.

Algorithm 2 4D-Var data assimilation with backpropagation. To assimilate observations, we include sensor data in the input and output of PredAAE. After calculating the data mismatch between the prediction and observation of the sensor values within one time level, the mismatch is minimised with respect to the AAE input using the ADAM optimiser. By marching forward through time, we can obtain an improved simulation with observation data assimilated.

This algorithm will make improved predictions for the POD coefficients for each time level in the set of desired values $\mathcal{N}$, given the trained AAE (f^{AAE}); solutions ($\boldsymbol{\alpha}^{n}$) at all time levels from the PredAAE algorithm; observations ($\boldsymbol{s}^{n}$); the learning rate of the Adam optimiser (lr); the maximum number of iterations for data assimilation (N^{its}_{DA}); a tolerance for the stopping data assimilation criterion (ε_{DA}).
initialise $\boldsymbol{\alpha}^{n,0} = \boldsymbol{\alpha}^{n}$ (with results of the PredAAE algorithm)
initialise `optimiser = Adam(lr)`
for time level $n \in \mathcal{N}$ **do**
  !! backpropagate data mismatch and optimise the input within this time level
  for iteration $i = 1, 2, \ldots, N^{its}_{DA}$ **do**
    Form $\boldsymbol{X}^{n,i-1}_{4dvar}$ and pass through the AAE to give $\tilde{\boldsymbol{X}}^{n,i-1}_{4dvar}$
    !! calculate the loss functional between $\boldsymbol{X}^{n,i-1}_{4dvar}$ *and* $\tilde{\boldsymbol{X}}^{n,i-1}_{4dvar}$
    $\mathcal{L}_{DA}\left(\boldsymbol{X}^{n,i-1}_{4dvar}, \tilde{\boldsymbol{X}}^{n,i-1}_{4dvar}\right) = \sum_{k=n-m+1}^{n} \left(\tilde{\boldsymbol{s}}^{k,i-1} - \boldsymbol{s}^{k,i-1}\right)^{T} \boldsymbol{W}_{s}\left(\tilde{\boldsymbol{s}}^{k,i-1} - \boldsymbol{s}^{k,i-1}\right)$
    !! determine whether the loss has been sufficiently reduced
    if $\left(\mathcal{L} < \varepsilon_{DA}$ or $i == N^{its}_{DA}\right)$ **then**
      $\boldsymbol{\alpha}^{n} = \tilde{\boldsymbol{\alpha}}^{n,i-1}$
      exit iteration loop
    end if
    !! backpropagate the mismatch and update the model input
    $\boldsymbol{\alpha}^{n,i}$ =`optimiser.apply_gradients`$(\mathcal{L}, \boldsymbol{\alpha}^{n,i-1})$
  end for
end for

Algorithm 3 Constrained data assimilation. By constraining the values of the sensors and POD coefficients at the previous (known) time levels, and the sensor values at the future time level, we can approximate the POD coefficients at the future time level.

This algorithm will make improved predictions for the POD coefficients for each time level in the set of desired values $\mathcal{N}$, given the maximum number of iterations (N^{its}_{conDA}); the trained AAE (f^{AAE}); a tolerance for the stopping criterion (ε_{conDA}); solutions, $\boldsymbol{\alpha}^{n}$, at all time levels; sensor values, $\boldsymbol{s}^{n}$, at all time levels; and the range of time levels $\mathcal{N}$.
for time level $n \in \mathcal{N}$ **do**
  $\boldsymbol{\alpha}^{n,0} = \boldsymbol{\alpha}^{n-1}$
  for iteration $i = 1, 2, \ldots, N^{its}$ **do**
    form $\boldsymbol{X}^{n,i-1}$ and pass through the autoencoder
    $\tilde{\boldsymbol{X}}^{n,i-1} = f^{\text{AAE}}(\boldsymbol{X}^{n,i-1})$
    !! update the approximation of the POD coefficients using values in $\tilde{\boldsymbol{X}}^{n,i-1}$
    $\boldsymbol{\alpha}^{n,i} = \tilde{\boldsymbol{\alpha}}^{n,i-1}$
    if $\left(||\tilde{\boldsymbol{\alpha}}^{n,i-1} - \boldsymbol{\alpha}^{n,i-1}||^{2}_{2} < \varepsilon_{conDA}$ or $i == N^{its}_{conDA}\right)$ **then**
      $\boldsymbol{\alpha}^{n} = \boldsymbol{\alpha}^{n,i}$
      exit iteration loop
    end if
  end for
end for

2.3. Ventilation control

To simulate scenarios with different Dyson fan settings and achieve ventilation control, we extend the idea of incorporating sensor values in the output of the AAE for data assimilation. Therefore, for control, we include ventilation parameters in the input and output of the AAE model, as shown below:

$$\boldsymbol{X}^n_{ctrl} = \begin{pmatrix} (\boldsymbol{\alpha}^n)^T, (\boldsymbol{s}^n)^T, (\boldsymbol{v}^n)^T \\ (\boldsymbol{\alpha}^{n-1})^T, (\boldsymbol{s}^{n-1})^T, (\boldsymbol{v}^{n-1})^T \\ \vdots \\ (\boldsymbol{\alpha}^{n-m+2})^T, (\boldsymbol{s}^{n-m+2})^T, (\boldsymbol{v}^{n-m+2})^T \\ (\boldsymbol{\alpha}^{n-m+1})^T, (\boldsymbol{s}^{n-m+1})^T, (\boldsymbol{v}^{n-m+1})^T \end{pmatrix}, \quad \tilde{\boldsymbol{X}}^n_{ctrl} = AAE\left(\boldsymbol{X}^n_{ctrl}\right). \quad (12)$$

where $(\boldsymbol{v}^n)^T = \left[v^n_1, v^n_2, \ldots, v^n_{N^{\text{Venparas}}}\right]$, v^n_i represents the ith ventilation parameter at time level n; and N^{Venparas} represents the number of ventilation parameters. During 4D-Var DA, for the purpose of control, these ventilation parameters are used along with observation data to calculate the data mismatch:

$$\mathcal{L}\left(\boldsymbol{X}^n_{ctrl}, \hat{\boldsymbol{X}}^n_{ctrl}\right) = \sum_{k=n-m+1}^{n} \left(\hat{\boldsymbol{s}}^k - \boldsymbol{s}^k\right)^T \boldsymbol{W}_s \left(\hat{\boldsymbol{s}}^k - \boldsymbol{s}^k\right) + \zeta \sum_{k=n-m+1}^{n} \left(\hat{\boldsymbol{v}}^k - \boldsymbol{v}^k\right)^T \boldsymbol{W}_v \left(\hat{\boldsymbol{v}}^k - \boldsymbol{v}^k\right), \quad (13)$$

where $\hat{\boldsymbol{X}}^n_{ctrl}$ represents the model input including $m-1$ desired sensor values $\hat{\boldsymbol{s}}$ and desired ventilation parameters $\hat{\boldsymbol{v}}$ for ventilation control. The scalar ζ controls the relative importance of the sensor mismatch and the mismatch of the control parameters. Here, all scalars and weight matrices are set to 1 and $\boldsymbol{I}$, respectively. To implement control, a similar process is used to that described in Section 2.2.1 and Algorithm 2. Sensor values and control variables are fixed in the input to the AAE, POD coefficients at previous time levels are also fixed and an estimate or guess for those at time level n is used. These values are passed through the AAE. The mismatch measured by the difference in input and output values of the sensor and control variables is minimised with respect to the POD coefficients at time level n and a new approximation to the POD coefficients at time level n is therefore obtained.

2.4. Flexibility of the AAE-based approach

Instead of training a neural network to generate predictions, and then performing data assimilation and control with this trained network, in this paper, we train a single neural network to predict, assimilate data and control the air quality. We have found this to be a particularly flexible approach.

The single network is an AAE with inputs and outputs given in Eq. (12). After training, Algorithm 1 is used to make predictions in time. In addition to the inputs described in that algorithm, the AAE also has inputs of sensor values and control variables, and these are treated exactly the same as the POD coefficients. So, all variables associated with time levels $n-m+1$, $n-m+2$, …, $n-1$, are treated as fixed (or known or pre-determined) and during the iteration process these remain unchanged. Only the variables at the future time level n are updated ($\boldsymbol{\alpha}^n$, $\boldsymbol{s}^n$ and $\boldsymbol{v}^n$) as the output of the AAE for these variables is fed back into the input of the AAE (See the first row of Table 1). To use this network for data assimilation, in addition to the previous variables that are fixed, the sensor values at the future time level n are also fixed, $\boldsymbol{s}^n$ (see the second row of Table 1). The mismatch to be minimised is based on the difference between the input and output of the sensor values at all time levels in the time window. To use this network for control, in addition to the variables that are fixed for data assimilation, the control variables at the future time level are also fixed, $\boldsymbol{v}^n$ (see the third row of Table 1). The mismatch to be minimised is based on the difference between the input and output of the sensor values and the control variables at all time levels in the time window.

Table 1
The separation of the AAE's variables into two groups: those that remain fixed during the iteration procedure and those that are updated.

	fixed or unchanged variables	variables to be updated
prediction	$\{\boldsymbol{\alpha}^k\}^{n-1}_{k=n-m+1}$ $\{\boldsymbol{s}^k\}^{n-1}_{k=n-m+1}$ $\{\boldsymbol{v}^k\}^{n-1}_{k=n-m+1}$	$\boldsymbol{\alpha}^n$, $\boldsymbol{s}^n$, $\boldsymbol{v}^n$
data assimilation	$\{\boldsymbol{\alpha}^k\}^{n-1}_{k=n-m+1}$ $\{\boldsymbol{s}^k\}^{n}_{k=n-m+1}$ $\{\boldsymbol{v}^k\}^{n-1}_{k=n-m+1}$	$\boldsymbol{\alpha}^n$, $\boldsymbol{v}^n$
control	$\{\boldsymbol{\alpha}^k\}^{n-1}_{k=n-m+1}$ $\{\boldsymbol{s}^k\}^{n}_{k=n-m+1}$ $\{\boldsymbol{v}^k\}^{n}_{k=n-m+1}$	$\boldsymbol{\alpha}^n$

Furthermore, a subset of sensor values and/or control variables can be chosen if certain values are not available. For example, if the AAE has been trained to have an input that includes sensor values of CO_2 concentration and relative humidity, but observations of humidity are not available, the sensor values at time level n associated with CO_2 can be put in the 'fixed-variable' group and the sensor values at time level n associated with humidity can be put in the 'variables to be updated' group. The loss function is then modified, Eq. (13), so that only the mismatch between input and output values of CO_2 is considered, omitting values of humidity.

3. Results

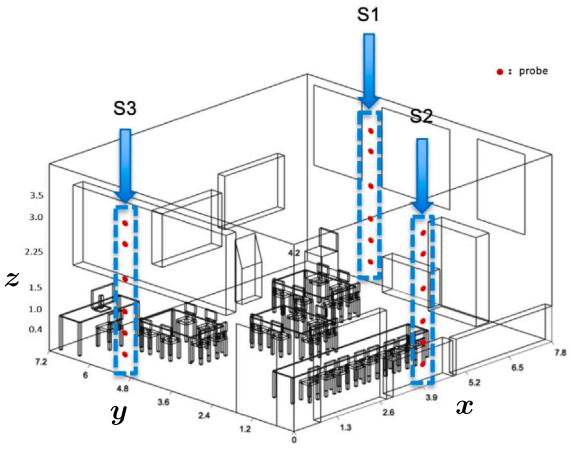
Firstly, we introduce the test case used for this work in Section 3.1. Secondly, we demonstrate the prediction, DA and control ability of our AAE model and 4D-Var DA algorithm in Sections 3.2 and 3.3. Thirdly, we present the application of our methods to the three ventilation control scenarios in Sections 3.4, 3.5 and 3.6. The first scenario is the control of an indoor comfortable living environment, focusing on the changes in indoor temperature and CO_2 concentration. The second scenario focuses on infection risk minimisation, or more specifically, reducing indoor viral loads during a pandemic scenario. The third scenario is based on a combination of the first two, and is designed to simulate a more realistic scenario, where during a pandemic and in the winter, how people still want to heat the classroom while simultaneously achieving a low viral load in the classroom air.

3.1. Test case — a school classroom

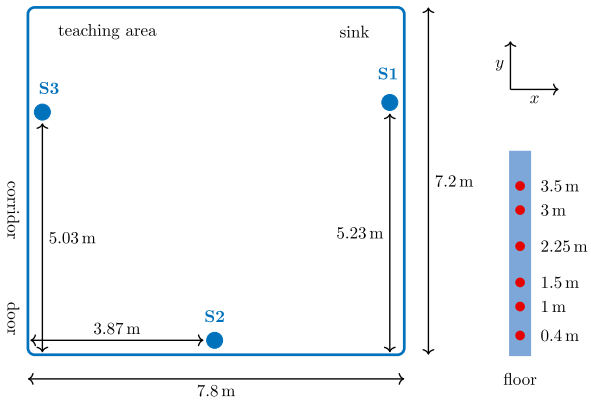
To demonstrate the proposed methods, we chose a classroom in a primary school in London as a test case [49]. Observations of CO_2, relative humidity, and temperature from 18 sensors were collected in one hour while 26 students and a teacher were in the classroom. Heating was provided by radiators and a Dyson fan, which also removed COVID-19 laden particles from the air. Fig. 2 shows a schematic diagram of a classroom with the locations of sensors, the fan, radiators, students and teachers. The door to the interior hallway is open (here on the same wall as the rack of sensors labelled 'S1'), and the top-hinged window on the right is open (here opposite the interior hallway door, next to the rack of sensors labelled 'S3'). It should be noted that during the monitoring period, the classroom had two water-heated radiators. These radiators were used continuously throughout the day to maintain the classroom temperature and were not switched off even of the classroom was unoccupied. For this reason, the classroom temperature was relatively high throughout the monitoring period.

3.1.1. Dyson fan

The fan in this test case is a Dyson TP04 with dimensions (L×B×H) in mm of 204 × 220 × 1050. This type of fan has a dual function; heating or cooling and air purification. It can detect pollutants in real-time and report air quality levels. As an air multiplier, the fan projects a powerful airflow of purified air across the room. The indoor air quality can be adjusted by setting the initial outflow speed and temperature of the Dyson fan, which has a power usage of about 45 watts. This research studies how to use the AI models to control these two parameter settings.

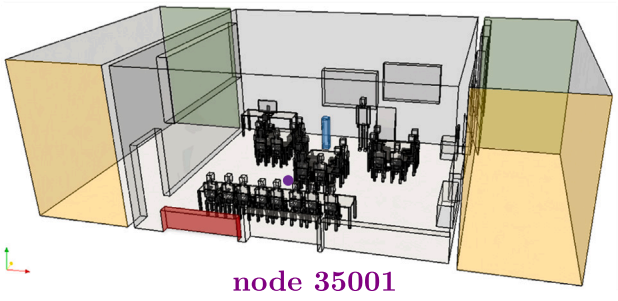


(a) The geometry used for the CFD simulations



(b) Sensor locations

(c) Dyson fan TP04



(d) Computational domain and node 35001 (red arrow, x axis; yellow arrow, y axis and green arrow, z axis)

Fig. 2. Classroom geometry, from [48]. Plots (a) and (b) show the location of the 18 sensors used to collect CO_2, relative humidity and temperature data. The sensors associated with the rack labelled 'S1' share the same x and y coordinates and are sensors 1, 4, 7, 10, 13, 16 (with increasing height). Similarly, sensors 2, 5, 8, 11, 14, 17 are located on rack 'S2' and sensors 3, 6, 9, 12, 15, 18 are located on rack 'S3'. See Kumar et al. [49] for more details. Plot (c) is the schematic of the Dyson fan [65] in the classroom, located at x=3.3 and y= 6.1. Plot (d) shows node 35001 at $(4.18, 0, 1.55)$ where we will show some results. .

3.1.2. CFD simulations

In addition to the observations, we performed a CFD Large Eddy Simulation (LES) to generate high-fidelity numerical data of the air flows and air quality in the classroom using the open source CFD code Fluidity [66,67]. Fluidity has been validated for indoor-outdoor exchanges against water flume experiments [30] and urban air flows against wind tunnel data [68–73]. Detailed information about the set-up of the problem tackled here is given in Heaney et al. [48]. To summarise, the CFD code solves the Navier–Stokes equations using an anisotropic adaptive meshing technique [74] and the finite-element method. An adaptive time stepping scheme is applied in this simulation, which selects time step based on a Courant number of 10. Thermal stratification is modelled by the Boussinesq approximation, which is valid when a change in density affects, primarily, the buoyancy forces. The viral load is modelled by solving the scalar transport equation as a passive tracer with a decay rate, σ, introduced to represent the realistic residence time of the virus:

$$C_t = C_0 e^{-\sigma t} \tag{14}$$

where C_0 and C_t refer to the viral load at the initial state and at time level t. The Indoor Geometry Generator (IGG) [75] was used to generate the classroom geometry. Two buffer zones were attached to the left and right of the classroom with inflow and outflow boundary conditions imposed on the walls of the buffer regions The natural ventilation allows air to pass through doors (left) connecting the classroom with the interior corridor and through top-hinged windows (right) on the exterior wall of the classroom. On the nearest facing planes, inlet velocities of 0.5 m/s (left buffer) and 1.0 m/s (right buffer) are specified. The non-hydrostatic pressure is set to zero by an open boundary condition on the far wall of the buffer zone. A natural boundary condition is applied on all other external boundaries and internal walls, by imposing a normal velocity of zero — effectively a shear stress of zero. A no-slip boundary condition is applied to any remaining surfaces (e.g., the surface of desks, objects and people). Regarding the temperature, zero heat flux is applied on the walls, floor, furniture and ceiling and inside the classroom, a Dirichlet temperature boundary condition is applied to the radiator enforcing a temperature of 32 °C. To model the heat transfer from the students to the surrounding air, we used a Robin boundary condition which balances the heat flux across the boundary with difference between the air temperature and the skin temperature (taken to be 34 °C [76]). For CO_2, relative humidity and virus concentration, Dirichlet boundary conditions are adopted at the 'mouth' and 'nose' of people.

Over the one hour simulation and observation period, every 5 s we sampled the solution fields that might affect the air quality (CO_2 concentration, three directional components of the velocity field, temperature, relative humidity and viral load, so $N^{\text{Fields}} = 7$). Although the solutions were generated using mesh adaptation, in order to apply POD, all solutions were interpolated onto the unstructured grid generated at the end of the simulation, which had $N^{\text{Nodes}} = 54{,}363$ nodes. The solutions or snapshots are divided into two groups: the training or visible dataset corresponding to the time period of $[0, 3000]$ seconds

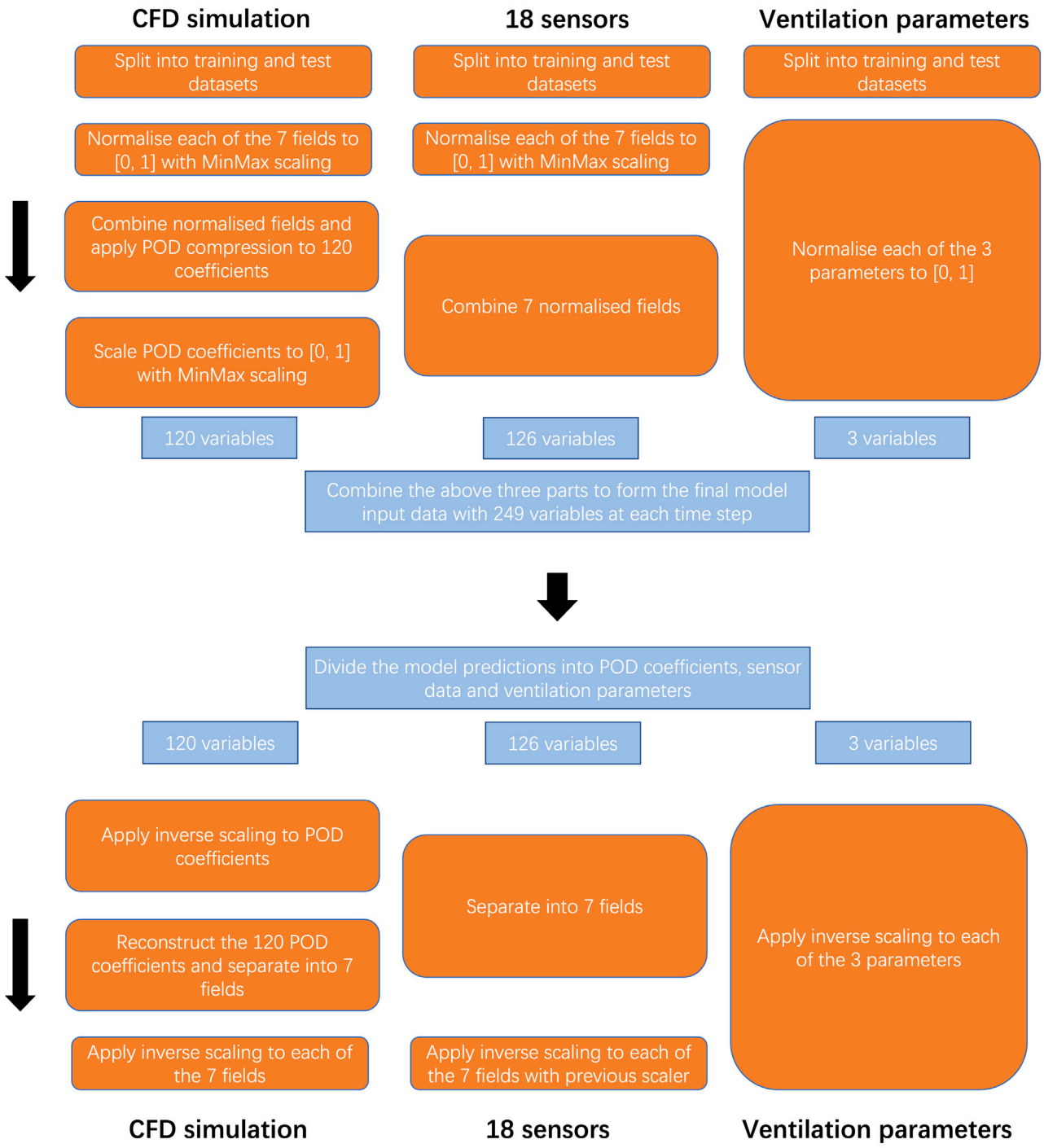


Fig. 3. Data processing workflow. The upper part of the figure illustrates the preprocessing which prepares the data for training. The lower part of the figure relates to the process of recovering the output of the models in the original primitive variables (velocities, temperature, etc.). Here, the three ventilation parameters are outflow velocity, flow rate and temperature, respectively, of which velocity and temperature correspond to the Dyson fan settings, see Table 2.

(600 time levels); and the time period corresponding to $[3000, 3600]$ seconds (120 time levels) is used to form the test or unseen dataset.

By setting different initial temperatures and air speeds of the Dyson fan for the CFD simulations, 7 different cases were performed, of which the final one (Case 7) is a simulation in which the fan is turned off throughout. When generating these different simulations, in addition to temperature (K) and outflow speed (m/s) v, we also include flow rate (m^3/h) $Q = v \times A$ as a ventilation parameter, where A refers to the cross-sectional area and is calculated as $0.13\,m^2$ for all the cases mentioned here. This was included to give an option for monitoring/controlling the energy (which depends on the flow rate), although we do not investigate energy control in this paper. The specific ventilation parameter settings are shown in Table 2. For these 7 cases, we used the first 6 cases to train and test the PredAAE model, and Cases 1 and 7 to

Table 2
Specific parameter settings of the Dyson fan ventilation for 7 cases. The first 6 cases are used to train and test the PredAAE model. In Case 7, the fan is turned off, and this case is used for optimal control.

CASE	Outflow velocity (m/s)	Flow rate (m^3/h)	Temperature (K)
1	1.0	468	298
2	1.0	468	293
3	1.0	468	303
4	2.0	936	298
5	0.5	234	298
6	0.5	234	293
7	–	–	–

test our ventilation control capability. We consider the outflow velocity and temperature to achieve ventilation control. By default, the optimal control is performed on Case 7 by assimilating the values and settings from Case 1. In the three scenarios, some of these values and settings are changed to match corresponding desired control targets.

3.1.3. Data processing and network architecture

We store CFD simulations as snapshots (in .vtu file format, see [67]) from all the cases for the 7 solution fields every 5 seconds for one hour (720 time levels in total). We then follow these steps. (1) Extract the solutions at all the spatial nodes of the chosen 7 fields from these .vtu files and convert them to .npy python format. (2) Extract the values of these fields at the 18 sensors from the .vtu files (by using the coordinates of the sensors) and convert them to .npy python format. (3) Divide the extracted data into the training dataset (the first 600 time levels) and test dataset (the last 120 time levels). (4) Process the two datasets separately; the first dataset being used to find normalisation mappings which are then applied to both datasets. (5) After the prediction result (for POD coefficients) is obtained, the seven fields are reconstructed in physical space and written into a new .vtu file. The specific data processing steps are shown in Fig. 3.

For this classroom test case, we specifically set the structure of the PredAAE model as shown in Table 3. As can be seen in the 'workflow' diagram shown in Fig. 3, we processed the input into 249 dimensions at each time level, including 120 POD coefficients to approximate the numerical simulation model, 126 coefficients representing 7 fields of 18 sensors and 3 ventilation parameters. We aggregated the model samples from 9 consecutive time levels. Therefore, the input and output sizes of the autoencoder are both 9 × 249. The sequences of 9 time levels are chosen at random from within the training dataset.

For the autoencoder network, we chose to use 2D convolutional (Conv2D) layers to make use of the correlation through time, avoiding a potential loss of information from data slicing and data stacking which is required when using fully connected layers [77]. The number of Conv2D layers of the encoder and decoder was set to 3 and 2, respectively. Using hyperparameter tuning, we set latent space to 512. For the discriminator, we chose to use 3 fully connected layers to distinguish the real or fake data, with dimensions of 256, 128 and 64, respectively. By tuning and optimising with a grid search, all final hyperparameters of the AAE model were obtained as shown in Table 4.

3.2. Predicting and assimilating data with a single set of Dyson fan parameters

3.2.1. Predicting for case 1

We build a reduced-order model in the manner described using the CFD data from Case 1, which has settings for the Dyson fan of an outflow air speed of 1 m/s and temperature of 298 K. The sensor values for the input of the AAE needed during training are obtained from the CFD simulations. We make predictions using PredAAE and then assimilate data (again from the CFD simulation) using the 4D-Var DA algorithm, comparing both sets of results with the CFD simulation.

Table 3
Architecture of the PredAAE. The right side is the autoencoder and the left side is the discriminator. The input of the autoencoder (9 × 249) represents the 249 elements from 9 consecutive time levels; the input of the discriminator (512) represents the latent space. The other numbers in the table represent the output dimensions of the corresponding layers. The output size of the conv2D layers in the table gives the two dimensions of the feature maps and the third number denotes the number of 3 × 3 filters used.

autoencoder			discriminator	
input	9 × 249			
conv2D	7 × 247 × 128			
conv2D	5 × 245 × 64			
conv2D	3 × 243 × 32			
reshape	23 328			
dense	8964			
dense	512	⟶	input	512
dense	8964		dense	249
reshape	9 × 249 × 4		dense	128
conv2D	9 × 249 × 64		dense	64
conv2D	9 × 249 × 32		output	1
output	9 × 249			

Fig. 4 shows the prediction results and from this, it can be seen that the PredAAE prediction (without 4D-Var DA), shown with blue lines, approximates well the ground truth from the CFD model. PredAAE makes very good predictions, not only for the first 3000 s (which corresponds to the training data), but also for the following 1200 s (corresponding to the test data). After applying 4D-Var DA, the results (orange lines in the graphs) remain close to the (CFD) observations. Visually, the data assimilation process does not have much impact on the PredAAE results, and, after DA, the results remain close to the CFD simulation ('or ground truth'), as the predictions are already so good. However, the RMSE values in Table 5 reveal measurable improvements, confirming the overall good performance of both approaches and the additional gains achieved with 4D-Var DA.

3.2.2. Dual twin experiment with the 4D-Var DA algorithm

In order to test the effectiveness of the 4D-Var DA algorithm further, we experiment with predicting from time 0 s to 1800 s whilst simultaneously assimilating data from time 1800 s to 3600 s. This type of dual-twin experiment will reveal if, by assimilating data from this later time period, the results will become consistent with the predictions from the later times or remain similar to the time from which the simulation was initiated. We use the ROM trained with CFD data from Case 1 only, still, and initialise the model using the results from the beginning of Case 1 (i.e. by using POD coefficients and sensor values (obtained from the CFD results) for the first 8 time levels of Case 1). The results are shown in Fig. 5. It can be seen that the results match very well with the predictions from time 1800 s and onwards, so the algorithm has started from time 0 s, yet made the predictions conform to time 1800 s and onwards in response to the data that has been assimilated. This verifies the ability of this 4D-Var DA method to assimilate observation data.

3.2.3. Assimilating observations from the classroom in the primary school

Here we assimilate observations taken from sensors in the classroom into our simulations. The sensors measure three fields only: temperature, CO_2 concentration and humidity. Therefore, the 4D-Var DA is performed using only the data of these three fields. The performance of the DA method at the one of the sensor locations is shown in Fig. 6. It can be seen that after DA, the results of these three fields are close to the observed values; the results of the other fields, as no corresponding data was assimilated, are similar to the prediction results from PredAAE. This demonstrates the ability of the 4D-Var DA algorithm to assimilate the observational data collected within the classroom. We note that the ground truths of velocity and virus load are taken from the CFD model as these were not measured by the sensors.

Table 4
Hyperparameter settings for the predictive AAE. The values over which a grid search was performed are on the right; the optimal values are on the left.

	optimal values	grid search
latent space	512	128, 256, 512, 1024
batch size	64	32, 64, 128
encoder conv2D layers	3	2, 3, 4, 5
decoder conv2D layers	2	2, 3, 4, 5
filter size	(3, 3)	(2,2), (3,3), (5,5)
optimiser	Adam	Adam, Adamax, RMSProp, SGD
activation functions:		LeakyReLU, ReLU, sigmoid
hidden layers in encoder and decoder	LeakyReLU	
hidden layers in discriminator	ReLU	
output layers	sigmoid	
epochs	12,000	6,000, 12,000, 18,000
learning rate:		10^{-2}, 10^{-3}, 5×10^{-3}, 9×10^{-3}, 10^{-4}
autoencoder	10^{-3}	
encoder-discriminator	5×10^{-3}	

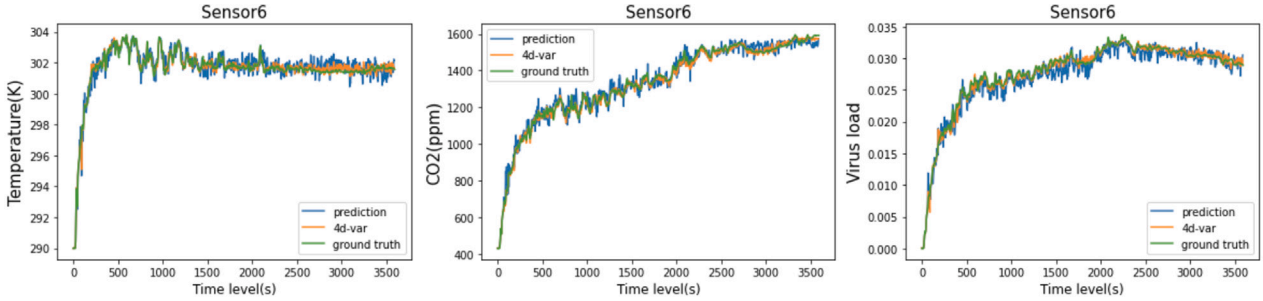


Fig. 4. The prediction of PredAAE and 4D-Var DA results after training on one set of parameter values (those corresponding to Case 1). Here, we choose the 6th sensor and show values of temperature, CO_2 concentration and viral load. The blue and green lines represent the PredAAE prediction, without 4D-Var DA, and the result after 4D-Var DA, respectively; the orange line represents the original CFD simulation of Case 1.

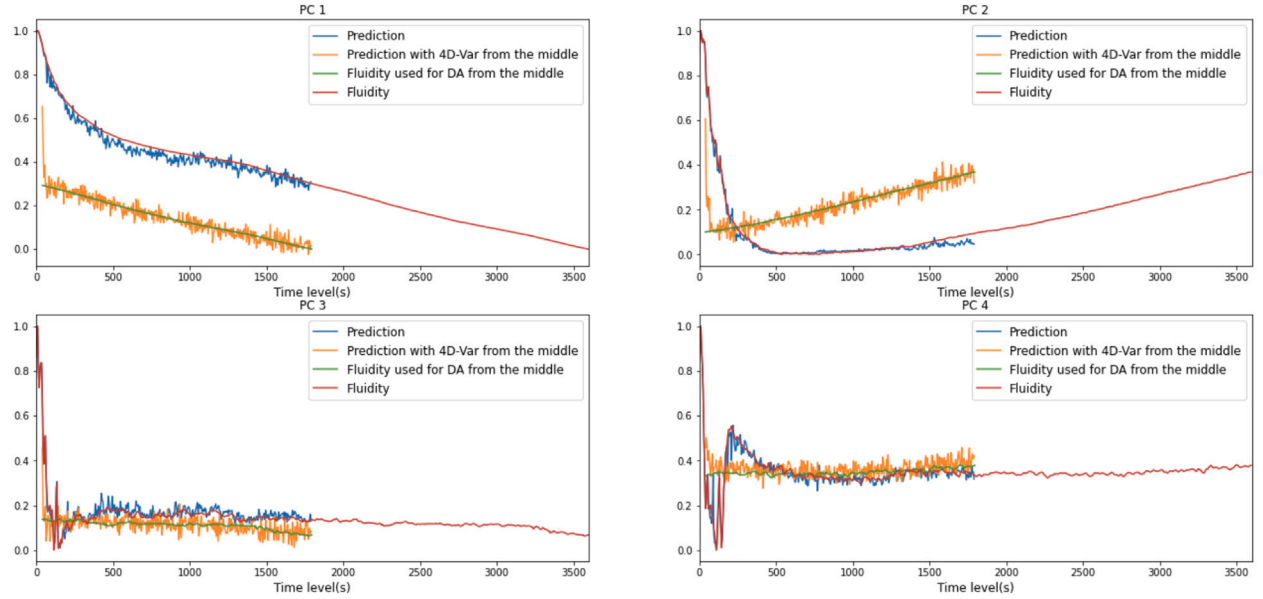


Fig. 5. Results of predicting from time 0 s whilst assimilating data from 1800 s (the temporal mid-point of the data we have) for the dual twin experiment. The magnitudes of the four most dominant POD modes are shown (normalised and labelled PC1, PC2 etc.). They can be viewed as indicators of the overall predictive ability of the DA method. The blue line ("Prediction") represents prediction with PredAAE, the red line represents the original CFD simulation ("Fluidity"), the green line indicates the observed value taken from 1800 s and onwards in the original CFD simulation ("Fluidity used for DA from the middle"), and the orange line is the result of 4D-Var DA process ("Prediction with 4D-Var from the middle").

3.3. Predicting for different settings of the Dyson fan

Here we construct a reduced-order model trained with simulations spanning a range of air speeds and temperature values for the Dyson fan (see Cases 1 to 6 in Table 2) in order to test the ability of our model to predict for different sets of parameter values. Here, we use PredAAE and 4D-Var DA to make predictions for the first 6 cases outlined in Table 2. The predicted results are shown in Fig. 7. By looking carefully at the figures, one can see that the 4D-Var DA can produce more accurate predictions than the PredAAE method without DA. These

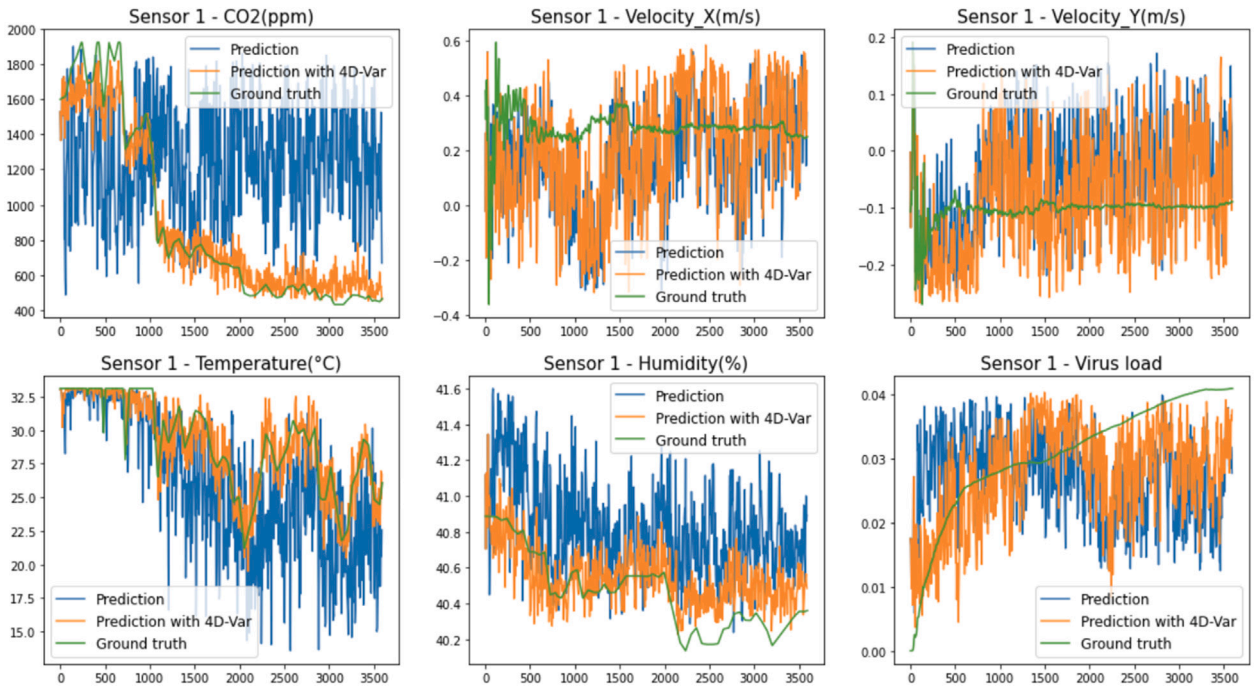


Fig. 6. 4D-Var DA results after assimilating sensor readings from the classroom. The results at Sensor 1 are shown. The blue line is the prediction using PredAAE, orange line is the result after 4D-Var DA and the green line is the ground truth. For temperature, CO_2 concentration and humidity, the ground truth refers to the observations from the classroom; for other fields, the ground truth refers to the Case 1 values .

Table 5
RMSE errors corresponding to Fig. 4. The errors are calculated for both the prediction and 4D-Var results against the ground truth across the three fields.

Method	Temperature (K)	CO_2 (ppm)	Virus load
Prediction	0.5476	3.8747	0.0012
4D-Var	0.1713	1.5360	0.0004

results suggest that PredAAE can make reasonable predictions for a range of parameter settings and that the 4D-Var DA algorithm can improve the match to the observations.

3.4. Scenario 1: control to achieve a comfortable environment

Applying the proposed control algorithm to achieve a comfortable classroom temperature and CO_2 concentration is our first test. Due to the flexibility of the AAE in incorporating observations and ventilation parameters as both inputs and outputs, in this scenario, we can specify the desired temperature and CO_2 concentration at the 4th to 12th sensors (the heights of these sensors are close to the head heights of the students) and attempt to find the speed of the Dyson fan that gives rise to these values. The desired value of the temperature setting of the Dyson fan and the temperature at the sensors is 21 °C; the desired value of the CO_2 concentration at the sensors is 800 ppm. The outflow velocity of the fan was initialised with a speed of $1\,m\,s^{-1}$ and then optimised using the proposed control algorithm. Fig. 8 shows speed of the outflow of the Dyson fan which is required to achieve the desired temperature and CO_2 level. We can see that, for this scenario, an outflow velocity setting of approximately $0.75\,m\,s^{-1}$ would be sufficient as there is no need to adjust the setting at every time level. By lowering the outflow velocity setting further, we could reduce energy consumption, but the desired temperature and CO_2 levels might be reached more slowly. Fig. 9 shows values of temperature and CO_2 at two of the sensor locations during this experiment. It can be seen from the line plots in Figs. 9(a) and 9(b) that, after applying our control algorithm, both the temperature and CO_2 concentration are much closer to the desired values of 21 °C and 800 ppm. In addition, we also conducted the same experiment using the constrained data assimilation method mentioned in Section 2.2.2 and the results are shown in Fig. 9(c). Although this method produces predictions that were close to the original CFD simulation, when using control the method struggled to find values of the control variables that lead to the desired temperature and CO_2 level. Results corresponding to those in Fig. 9 but at all the sensor locations can be seen in Figs. A.16, A.17 and A.18 in the appendix.

Fig. 10 shows the comparisons of iso-surfaces of humidity coloured with temperature both with and without control. The humidity iso-surface with control at 1200 s has a similar profile to that without control at 2400 s. Also, the humidity iso-surface with control at 2400 s has a similar profile to that without control at 3600 s. This is because without control, the fan speed is $1\,m\,s^{-1}$, whereas with control, the fan speed is reduced to $0.75\,m\,s^{-1}$ and the humidity will disperse less quickly through the domain.

3.5. Scenario 2: COVID-19 pandemic

Since the outbreak of COVID-19 in 2019, the global pandemic has changed people's perceptions of viruses. There is an increased desire to achieve good air quality with low viral load, in order to reduce, as much as possible, the likelihood of infections spreading from person-to-person in enclosed spaces. For the second scenario, we focus on analysing whether a specified level of viral load can be reached and maintained by certain settings of the Dyson fan. Since we cannot intuitively judge how to use the Dyson fan to control the viral load directly, we perform control through 4D-Var DA on the viral load fields of the 4th to 12th sensors (which are closest to the location of the students' heads) with the aim of minimising the values of viral load. In this case, the corresponding desired values were set to a nominal low value of $0.001\,particles/(m^3\,s)$. The actual values of viral load are

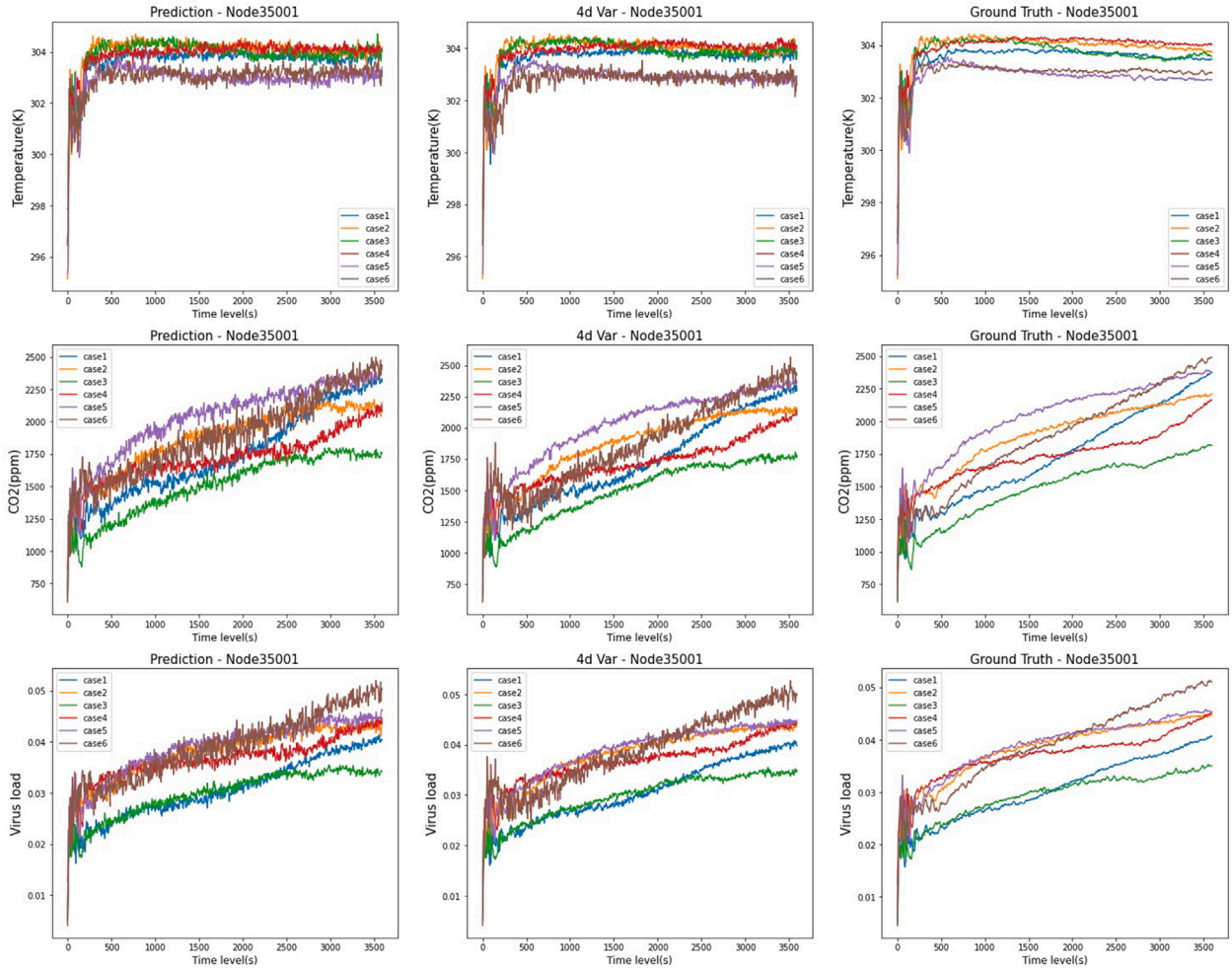


Fig. 7. The prediction of PredAAE and 4D-Var DA results for Cases 1 to 6 outlined in Table 2. Each row represents temperature, CO_2 concentration and viral load from top to bottom. Each column represents the prediction without data assimilation (PredAAE), the 4D-Var DA and the corresponding original CFD simulations (ground truth) from left to right. The subplots are generated at node 35001 in the FEM solution domain. See Fig. 2(d) for the location of the node.

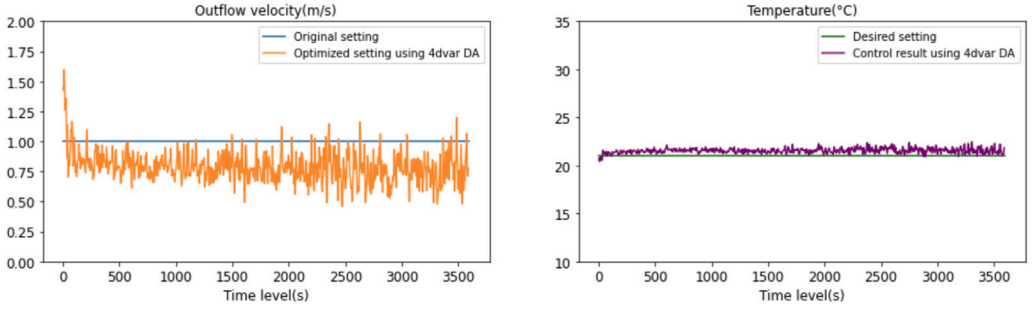
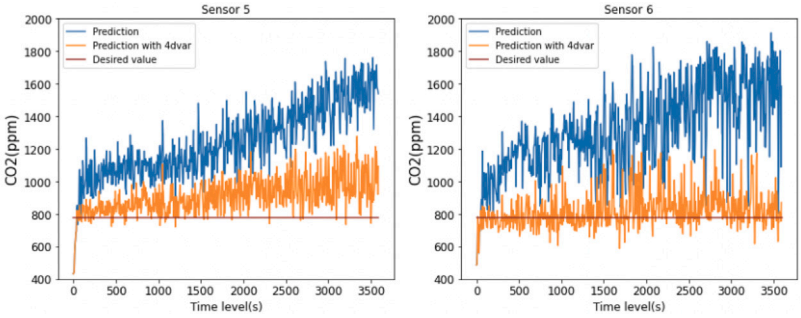


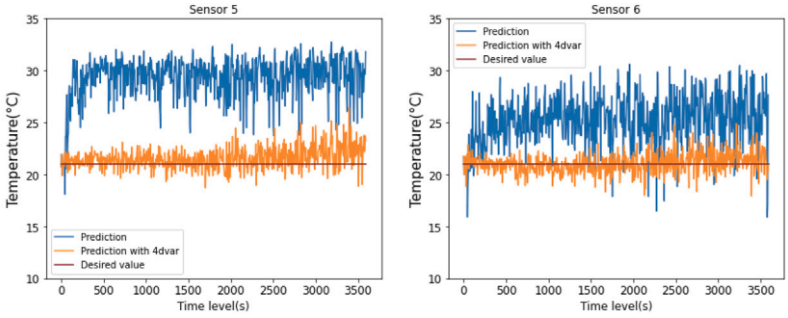
Fig. 8. Scenario 1 control results: Dyson fan ventilation settings. In this scenario, we wish to obtain a temperature of 21 °C at the fan and 4th to 12th sensors, and a CO_2 concentration of 800 ppm at the 4th to 12th sensors. The control algorithm attempts to find a fan speed which gives rise to this. The initial fan speed (blue) and the speed obtained by the control algorithm (orange) can be seen on the left, and the desired (green) and actual (purple) temperatures of the fan can be seen on the right.

arbitrary as they are defined by the viral load concentrations at the mouths of the people within the room. The temperature and outflow speed of the Dyson fan were initialised with values of 25 °C and 1 $m\,s^{-1}$, respectively (the same setting as in Case 1), and then optimised through the proposed method of control.

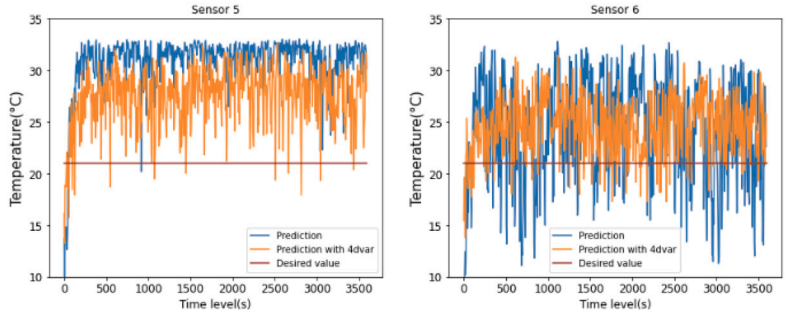
Fig. 11(a) shows the optimised values for the control variables of the Dyson fan. We can see that, in this scenario, we could gradually reduce the outflow velocity setting for the first half of the simulation and stabilise it around 0.25 m/s, while setting the temperature at 20 °C. It can be seen from the line plots in Fig. 11(b) that, with control,



(a) Predictions of CO_2 concentration at sensors 5 and 6 with control using the 4D-Var DA method



(b) Predictions of temperature at sensors 5 and 6 with control using the 4D-Var DA method



(c) Predictions of temperature at sensors 5 and 6 with control using the constrained DA algorithm

Fig. 9. Scenario 1 control results: (a) CO_2 concentration. (b) Temperature. (c) Temperature results of using the constrained DA algorithm for comparison. Each subplot represents the change over time of a particular variable at sensors 5 and 6. Blue lines represent predictions made with the PredAAE algorithm without control. The orange lines represent predictions with control (using either 4D-Var DA or constrained DA algorithms). Purple lines in the subplots represent desired control values. Results for all the sensor locations can be found in the appendix (Figs. A.16, A.17 and A.18) .

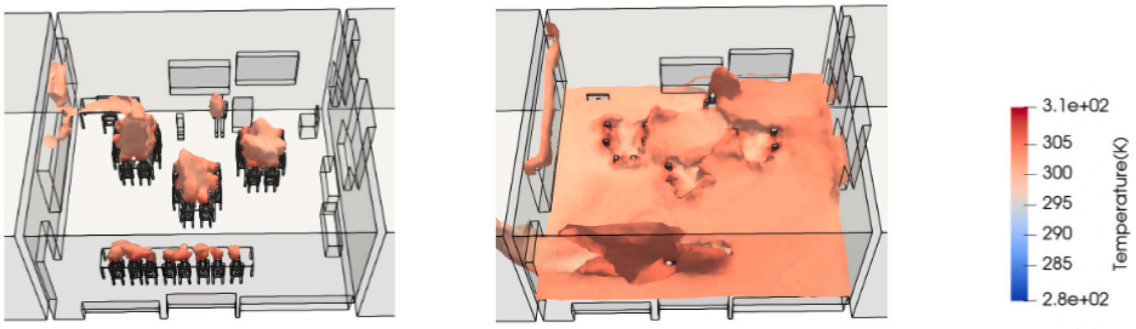
the viral load at sensors 5 and 6 is reduced to $0.005\,\text{particles}/(\text{m}^3\,\text{s})$, close to the desired value of $0.001\,\text{particles}/(\text{m}^3\,\text{s})$. Fig. 11(c) shows how temperature changes as a result of controlling the viral load. Results at all the sensors locations can be seen in Figs. A.19 and A.20 in the appendix. Fig. 12 shows a comparison of iso-surfaces of CO_2 concentration coloured by viral load after virus control against the simulation without control — Case 1. It can be seen that at the same time level, the shapes of the CO_2 concentration contour plots at the same value have a large difference between the simulations with and without control.

3.6. Scenario 3: heating in winter during the pandemic scenario

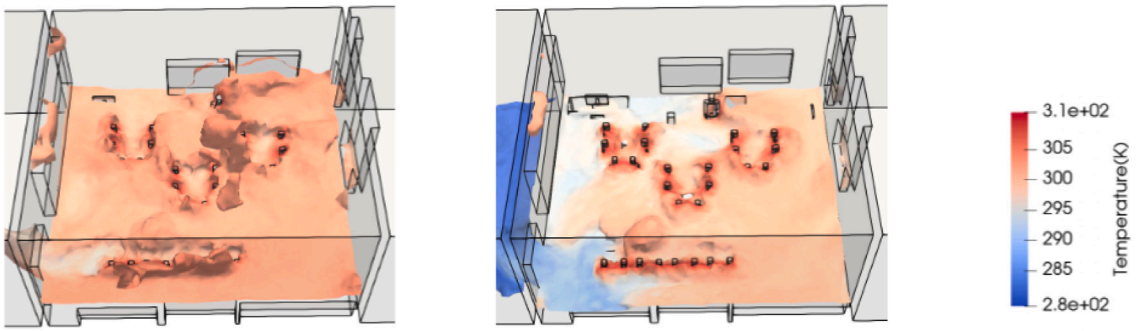
Scenario 3 combines elements of the previous two scenarios and occurs in the winter during the COVID-19 pandemic. In this scenario, we wish to maintain a low viral load indoors and simultaneously a warm temperature in the room. We performed control using the same algorithm as for the 4D-Var DA method but with a functional that attempts to control both of the viral load and temperature fields at the 4th to 12th sensors (the heights of these sensors are around the students), see Eq. (13). The control algorithm attempts to find a setting of the outflow velocity (initialised at $1\,\text{m}\,\text{s}^{-1}$) and the temperature of the fan that give rise to a low viral load of $0.001\,\text{particles}/(\text{m}^3\,\text{s})$ and a temperature at the sensors of 30 °C. It is worth noting that the CFD simulations used in this study have a relatively high temperature anyway, so to test whether the control algorithm can raise the temperature in the classroom, we set a relatively high value for our desired temperature.

Fig. 13(a) shows the optimised Dyson fan settings after applying the control algorithm. The figure reveals that to achieve our goals we could reduce the outflow velocity from its initial value of $1\,\text{m}\,\text{s}^{-1}$ to $0.8\,\text{m}\,\text{s}^{-1}$, at the beginning, and reduce the temperature setting from 30 °C to 25 °C after 1800 s. This could significantly reduce energy consumption by comparison with keeping the temperature of the fan at 30 °C and the speed at $1\,\text{m}\,\text{s}^{-1}$.

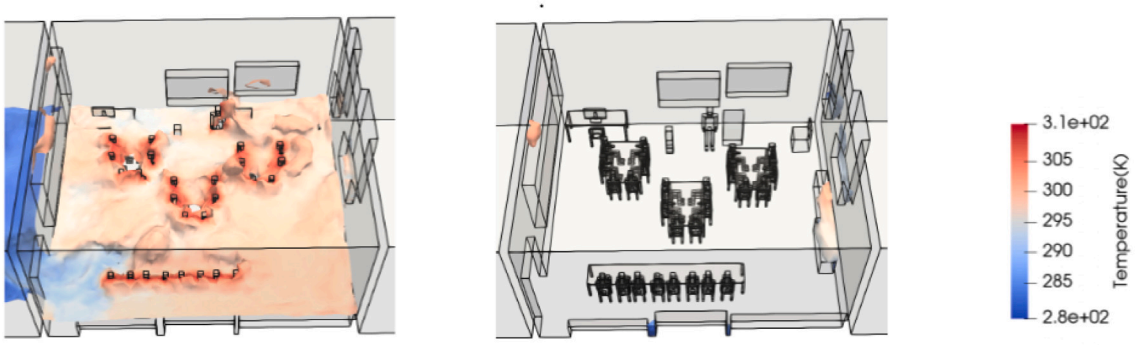
Fig. 13(b) shows that, with the proposed control algorithm, the viral load at sensors 8 and 9 is reduced significantly. The control result is



(a) Predictions with control (left) and without control (right) at 1200 s



(b) Predictions with control (left) and without control (right) at 2400 s



(c) Predictions with control (left) and without control (right) at 3600 s

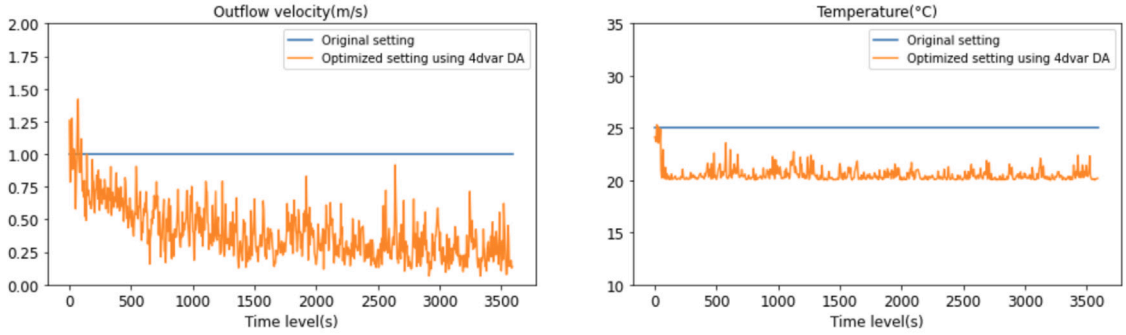
Fig. 10. 3D contour plot comparison of Scenario 1 using the control algorithm (left) and the same experiment (Case 1) using no control (right). Each subplot is a humidity iso-surface at the value 0.408 and coloured by temperature. Each row represents one time level.

similar to that of Scenario 2 (see Fig. A.19) in that the algorithm is able to reduce the viral load to a level approaching that of the desired value. Fig. 14 shows a comparison of Scenarios 2 and 3, highlighting the temperature at the sensors that we are aiming to control. Results show that the temperature near the students stable, at around 30 °C, as a result of applying the control algorithm. Figs. A.21 and A.22 in the appendix show the viral load and temperatures for all sensors locations. Fig. 15 shows a comparison of 3D CO_2 concentration contour plots coloured by viral load before (original state — case 1) and after control. The coloured surfaces of the contour plots show that the viral load has been reduced effectively in the entire classroom. A consequence of this viral load control is that the CO_2 level within the classroom has also been reduced.

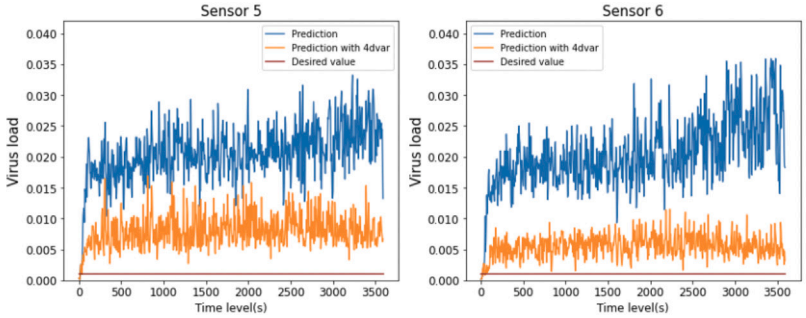
In these three scenarios, we have obtained the optimised settings of the air speed and air temperature for a Dyson fan to achieve certain desired levels of temperature and viral load at the sensors nearest to the students.

4. Conclusions and future work

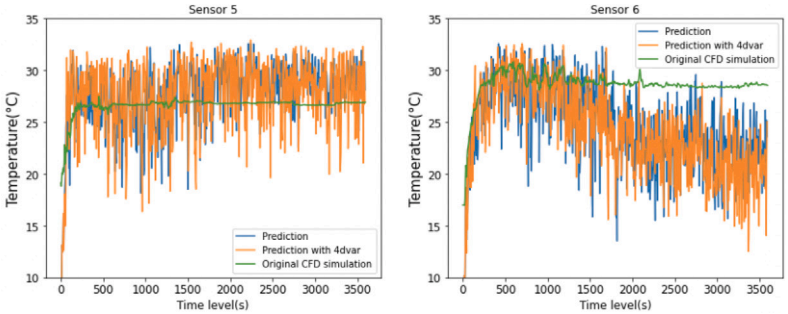
We propose a machine learning approach for control of ventilation in enclosed spaces based on Computational Fluid Dynamics (CFD) simulations and neural networks. We use the Adversarial Autoencoder (AAE) to predict, to assimilate data, and to build an accurate and efficient control workflow. Along with proper orthogonal decomposition (POD), the AAE is used as a surrogate model of the CFD to make the method computationally tractable. POD coefficients substantially reduce the size of the inputs and outputs of the AAE. After training, the framework can be used for all three tasks (prediction, data assimilation and control). All or selected sensor values can be used for the



(a) Results of Dyson fan ventilation settings. The blue and orange lines represent a prediction for the settings of Case 1 and a prediction for the optimised results, respectively.



(b) Results of viral load at sensors 5 and 6. Blue lines: predictions without control (using PredAAE); orange: with control (using 4D-Var DA); purple: desired values.



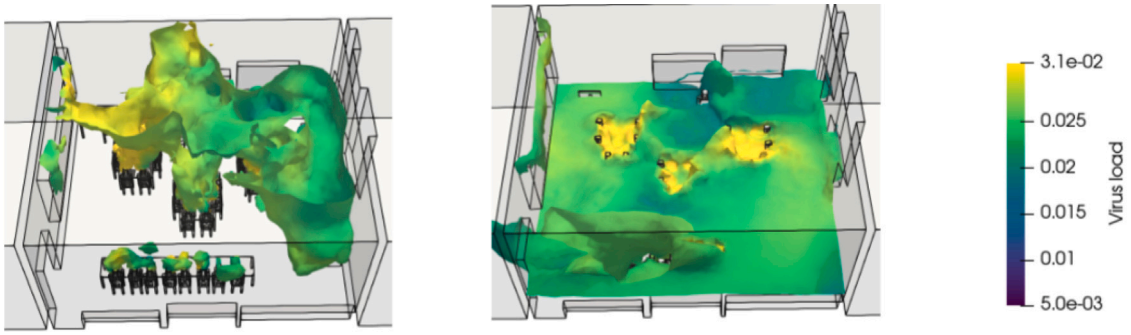
(c) Results of temperature at sensors 5 and 6. Blue lines: predictions without control (using PredAAE); orange: with control (using 4D-Var DA); green: original CFD simulation (Case 1).

Fig. 11. Scenario 2 control results: (a) Dyson fan ventilation settings. (b) Viral load at sensors 5 and 6. (c) Temperature at sensors 5 and 6. In this scenario, we let both the outflow velocity and temperature settings be optimised by our method based on the original Case 1 ventilation settings. Results of the viral load and temperature for all sensor locations are provided in the appendix (Figs. A.19 and A.20).

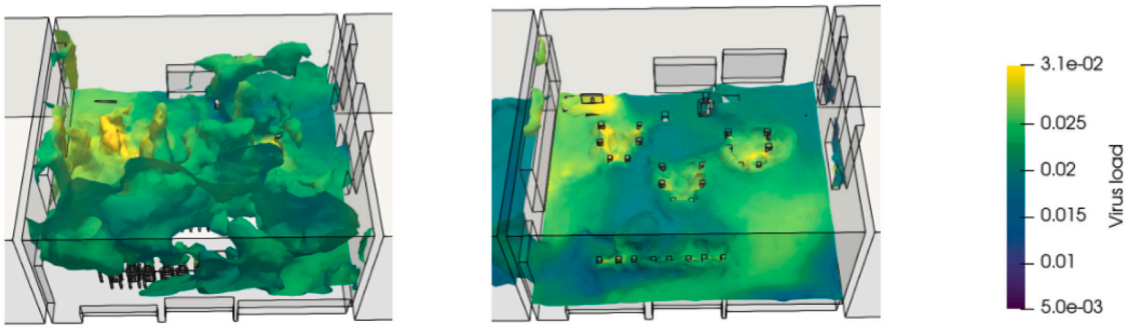
assimilation and control without the need to re-train the network, which gives flexibility to the method, making it robust in the case of missing data. We use a primary school classroom in London as a test case. Three ventilation control scenarios were investigated here, using the proposed approach.

In this paper, we focused on controlling the relevant indicators (e.g. temperature, viral load) near students within a classroom through time by finding optimised settings of a Dyson fan (its temperature and outflow velocity). In Scenario 1, we aimed to achieve a comfortable heating environment; in Scenario 2, we aimed to achieve lower viral loads during a pandemic situation; in Scenario 3, we aimed to achieve both comfort control (gauged by temperature) and lower viral load in the winter. We were able to control the environment in all three scenarios to meet the control objectives (comfort, health CO_2, infection risk minimisation).

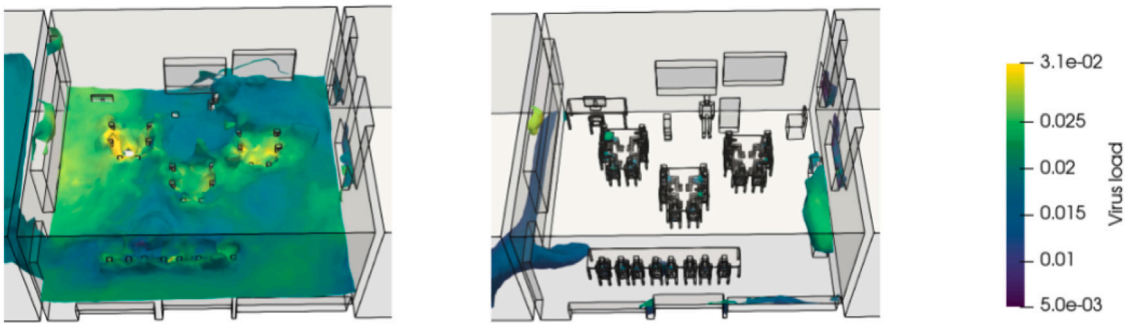
The resulting framework is highly flexible. For example, after training, the user can choose whether to make predictions at the sensor locations or to assimilate data at the sensor locations, meaning that if for any reason, sensor observations are missing, the framework still operate without the need to re-train the neural network. Similarly, the



(a) Predictions with control (left) and without control (right) at 1200 s



(b) Predictions with control (left) and without control (right) at 2400 s



(c) Predictions with control (left) and without control (right) at 3600 s

Fig. 12. 3D contour plot comparison of Scenario 2 with and without control (Case 1). Each subplot shows an iso-surface of CO_2 concentration with value 1200 ppm coloured by viral load. Each row represents one time level.

user can also decide how many of the control variables to constrain. Again, this decision does not affect the operation of the framework or require the network to be retrained. Another advantage of this framework is that interpolation is not needed to calculate the mismatch between sparse spatial observations or control variables and highly resolved CFD results, as the observations and control variables are predicted directly by the autoencoder.

Further work would examine a wider range of conditions including occupancy, outside weather and personal preferences in terms of comfort. Also, as the health status of individuals is a factor in their reaction to pollution [78], we could bring this into the model, so that asthmatics could be provided with less polluted, filtered air, for example.

CRediT authorship contribution statement

Donghu Guo: Writing – review & editing, Writing – original draft, Software, Methodology. **Claire E. Heaney:** Writing – review & editing, Supervision, Software, Methodology. **Boyang Chen:** Writing – review & editing, Supervision, Software. **Jieyi Tang:** Writing – review & editing, Software, Methodology. **Andrea Cammarano:** Writing – review & editing, Funding acquisition. **Prashant Kumar:** Writing – review & editing, Funding acquisition, Data curation. **Christopher C. Pain:** Writing – review & editing, Supervision, Software, Methodology, Funding acquisition, Conceptualization.

Funding sources

The authors would like to acknowledge the following UKRI grants: D-XPERT: AI-Powered Total Building Management System (Innovate UK, TMF 10097909); AI-Respire, "AI for personalised respiratory health and pollution, UK (EP/Y018680/1); RELIANT, Risk EvaLuatIon fAst iNtelligent Tool for COVID19, UK (EP/V036777/1); CO-TRACE, COvid-19 Transmission Risk Assessment Case Studies — education Establishments, UK (EP/W001411/1); INHALE, Health assessment across biological length scales, UK (EP/T003189/1); the PREMIERE programme, UK grant (EP/T000414/1).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We are very grateful to Dr Laetitia Mottet for allowing us to use the Indoor Geometry Generator [75], developed as part of the MAGIC (EP/N010221/1) and RAMP projects, in order to be able to generate geometries for objects and people sitting, standing or lying, in indoor spaces such as offices, schools, shops and trains. We also thank the University of Surrey's Global Centre for Clean Air Research (GCARE) team members Dr Arvind Tiwari, Nidhi Rawat, Dr Sarkawt Hama, Dr Gopinath Kalaiarasan and Dr Ana Paula Mendes Emygdio for collecting the classroom data [49]. Finally, thanks to Imperial College Research Computing Service (http://doi.org/10.14469/hpc/2232) for providing computational resources.We would also like to thank the reviewers for their constructive comments which have improved the manuscript.

Appendix. Results for control experiments at all the sensor locations

Results for scenario 1

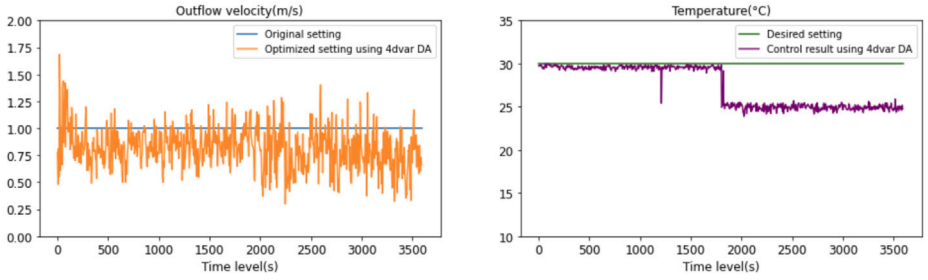
See Figs. A.16–A.18.

Results for scenario 2

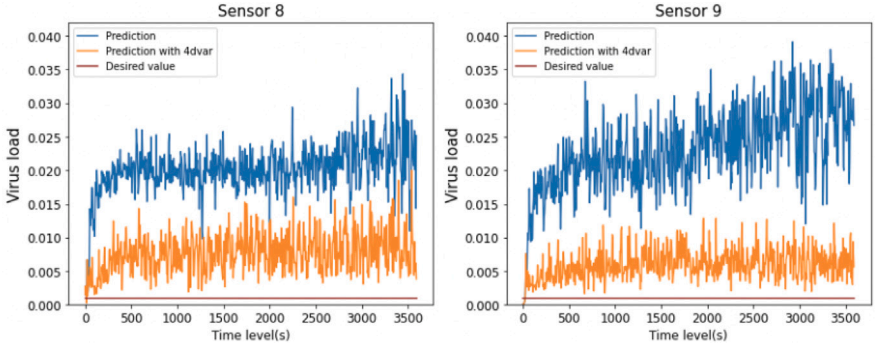
See Figs. A.19 and A.20.

Results for scenario 3

See Figs. A.21 and A.22.

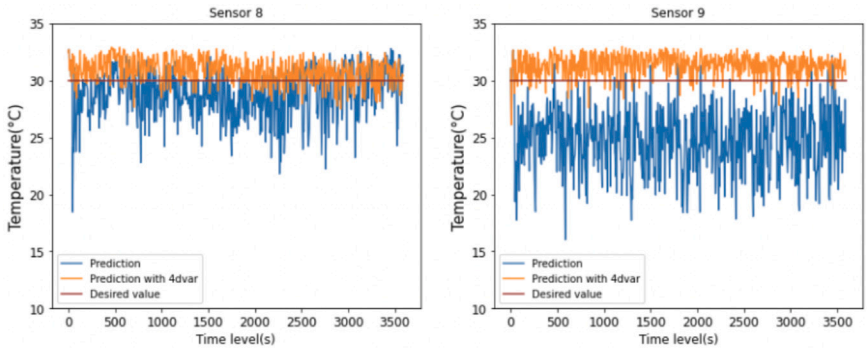


(a) Results of Dyson fan ventilation settings. Blue and orange lines represent the original Case 1 outflow velocity setting and optimised result, respectively; Green and purple lines represent the desired control value and control result of the temperature setting, respectively.

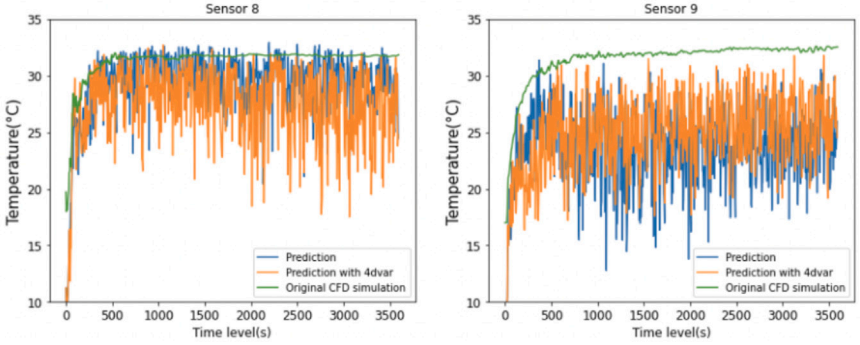


(b) Results of the viral load at sensors 8 and 9. Blue lines: predictions without control (using PredAAE); orange: with control (using 4D-Var DA); purple: desired values.

Fig. 13. Scenario 3 control results: (a) Dyson fan ventilation settings and (b) Viral load at sensors 8 and 9. In this scenario, we set the desired air temperature and optimise the outflow velocity of the fan (the same as in Scenario 1). Results of the viral load for all sensor locations can be found in the appendix (Fig. A.21).

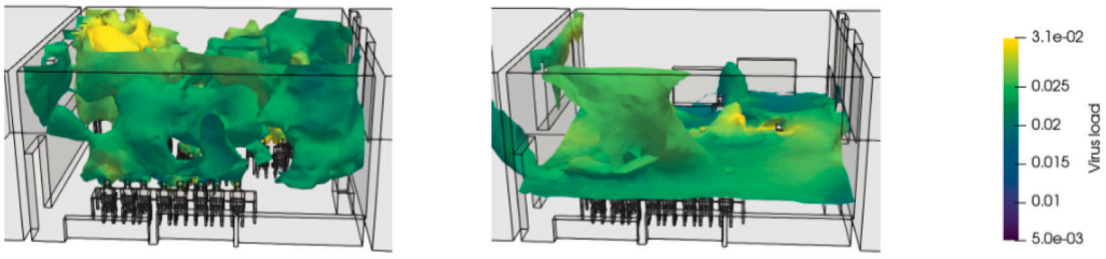


(a) Scenario 3 control results on the temperature at sensors

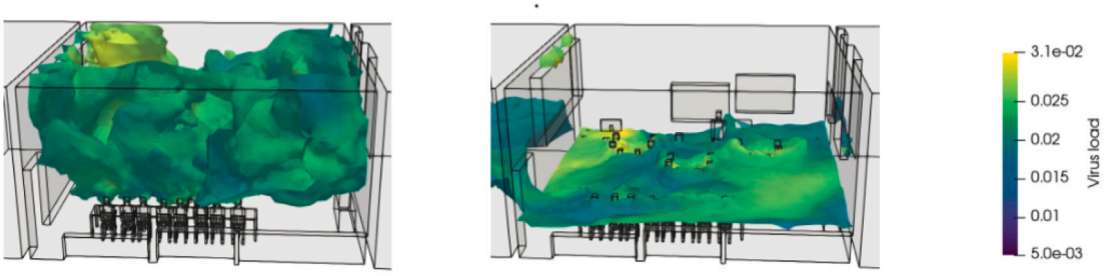


(b) Scenario 2 control results on the temperature at sensors

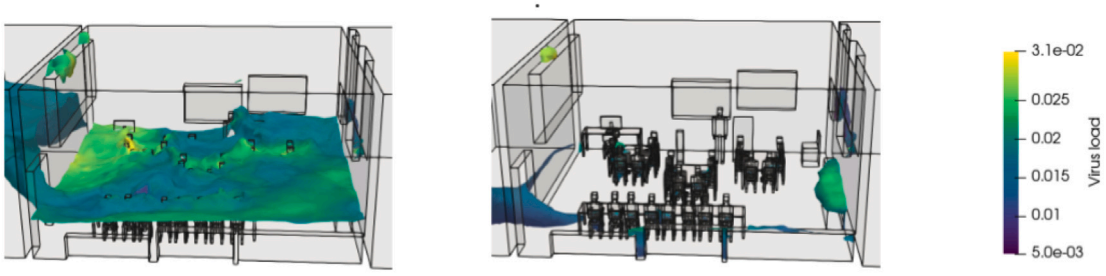
Fig. 14. Comparison of Scenario 3 and Scenario 2 control results for the temperature at two selected sensors (at which control is applied). Blue and orange lines represent a prediction without control (PredAAE) and prediction with control using 4D-Var DA, respectively. Plot (a) shows the results for Scenario 3, where purple lines represent desired values we set to control. Plot (b) shows the corresponding results for Scenario 2, where green lines represent the original CFD simulation (Case 1). Results for all sensor locations can be found in the appendix (Fig. A.22.)



(a) Predictions with control (left) and without control (right) at 1200 s



(b) Predictions with control (left) and without control (right) at 2400 s



(c) Predictions with control (left) and without control (right) at 3600 s

Fig. 15. 3D contour plots of Scenario 3 with control and without control (Case 1). Each subplot is an iso-surface of CO_2 at 1200 ppm and coloured by the viral load. Each row represents one time level.

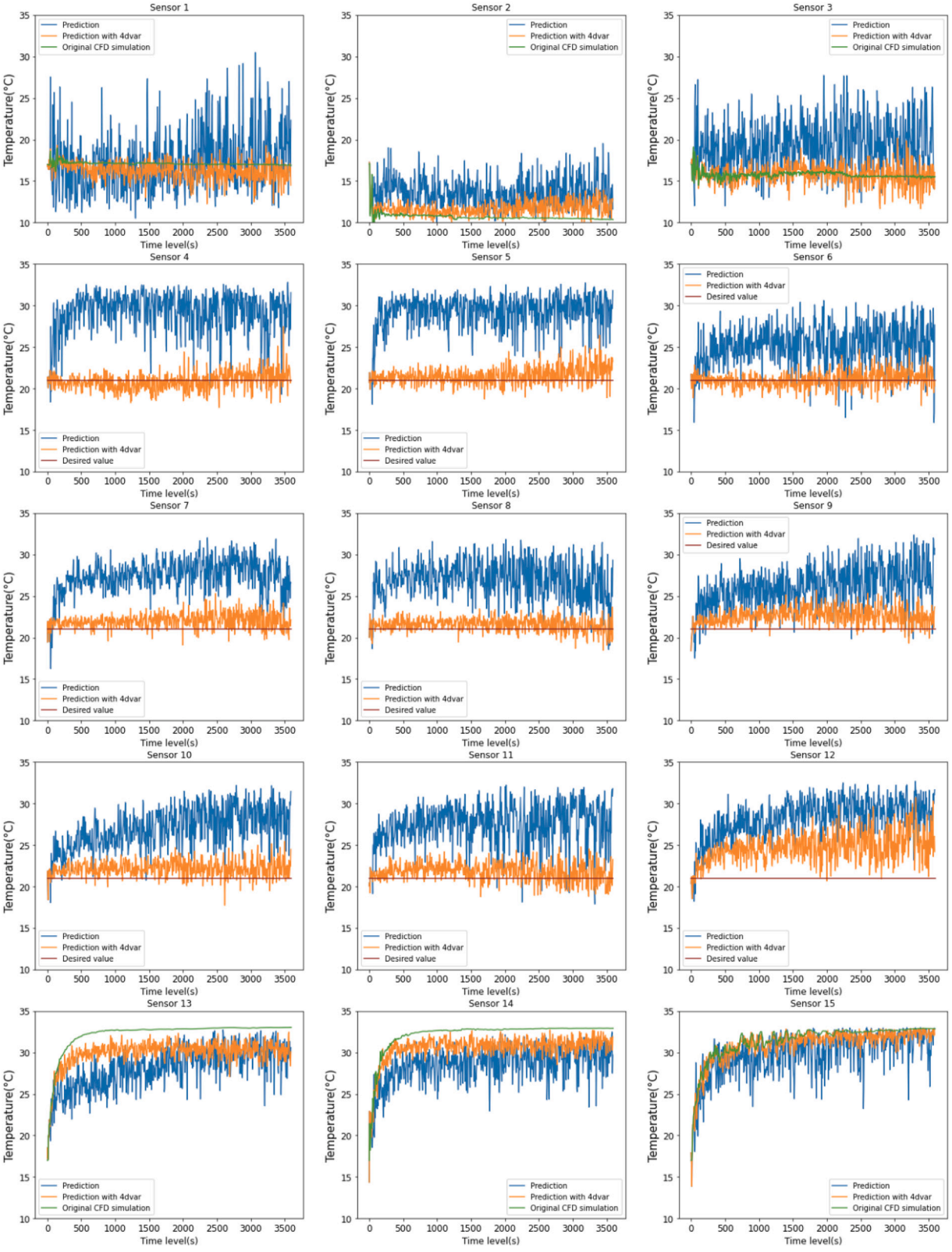


Fig. A.16. Scenario 1 control results: the temperature at sensors. Each subplot represents the temperature changing over time at a particular sensor. Blue and orange lines represent prediction with PredAAE and 4D-Var DA, respectively. Purple lines in the subplots for Sensor 4 to Sensor 12 (close to the head heights of the students) represent desired values we wish to control. Green lines in the subplots for other sensors represent the original CFD simulation (Case 1).

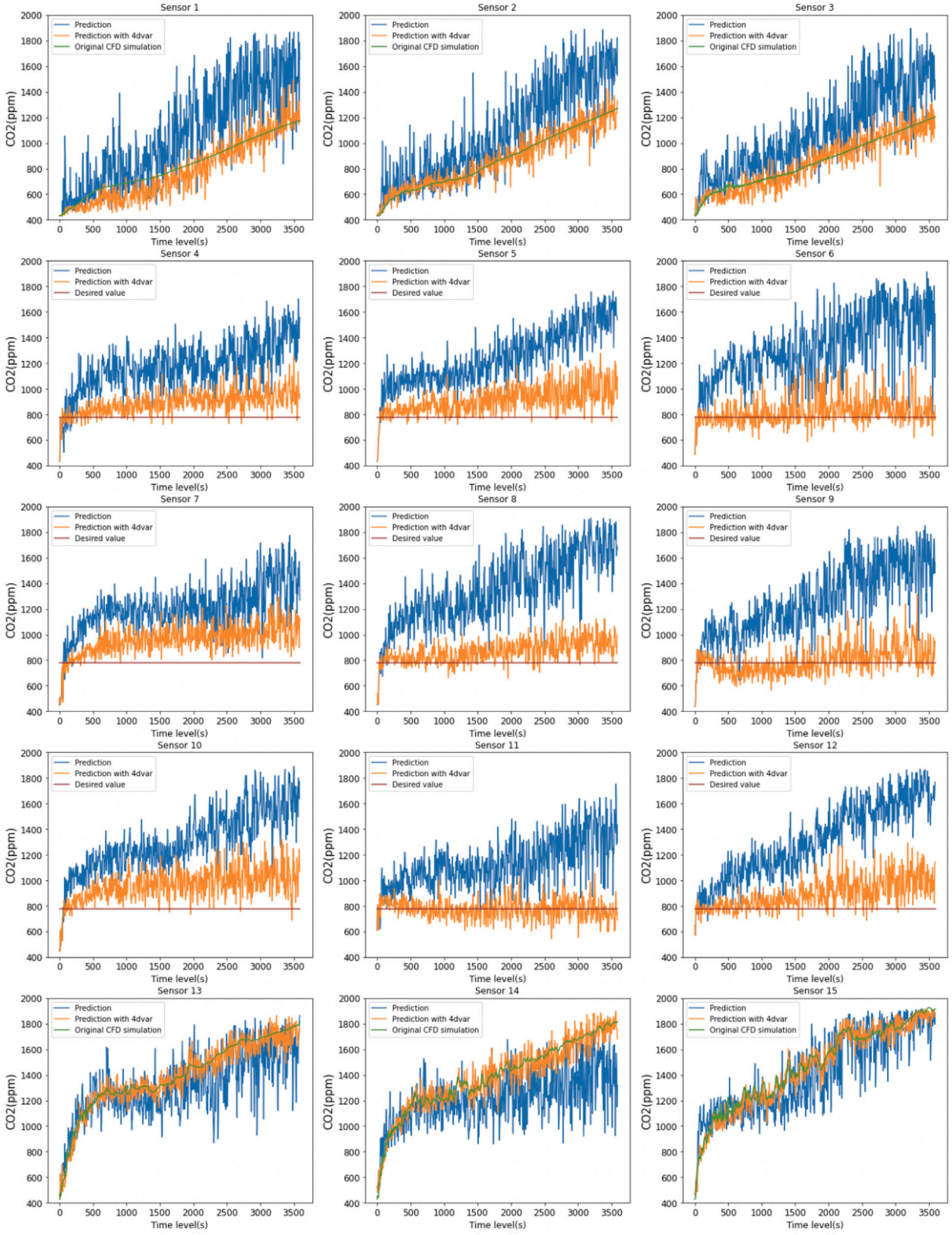


Fig. A.17. Scenario 1 control results: the CO_2 concentration at sensors. Each subplot represents the CO_2 concentration changing over time at a particular sensor. Blue and orange lines represent prediction from PredAAE and 4D-Var DA, respectively. Purple lines in the subplots for Sensor 4 to Sensor 12 (around the head height of the students) represent values we wish to control. Green lines in the subplots for other sensors represent the original CFD simulation (Case 1).

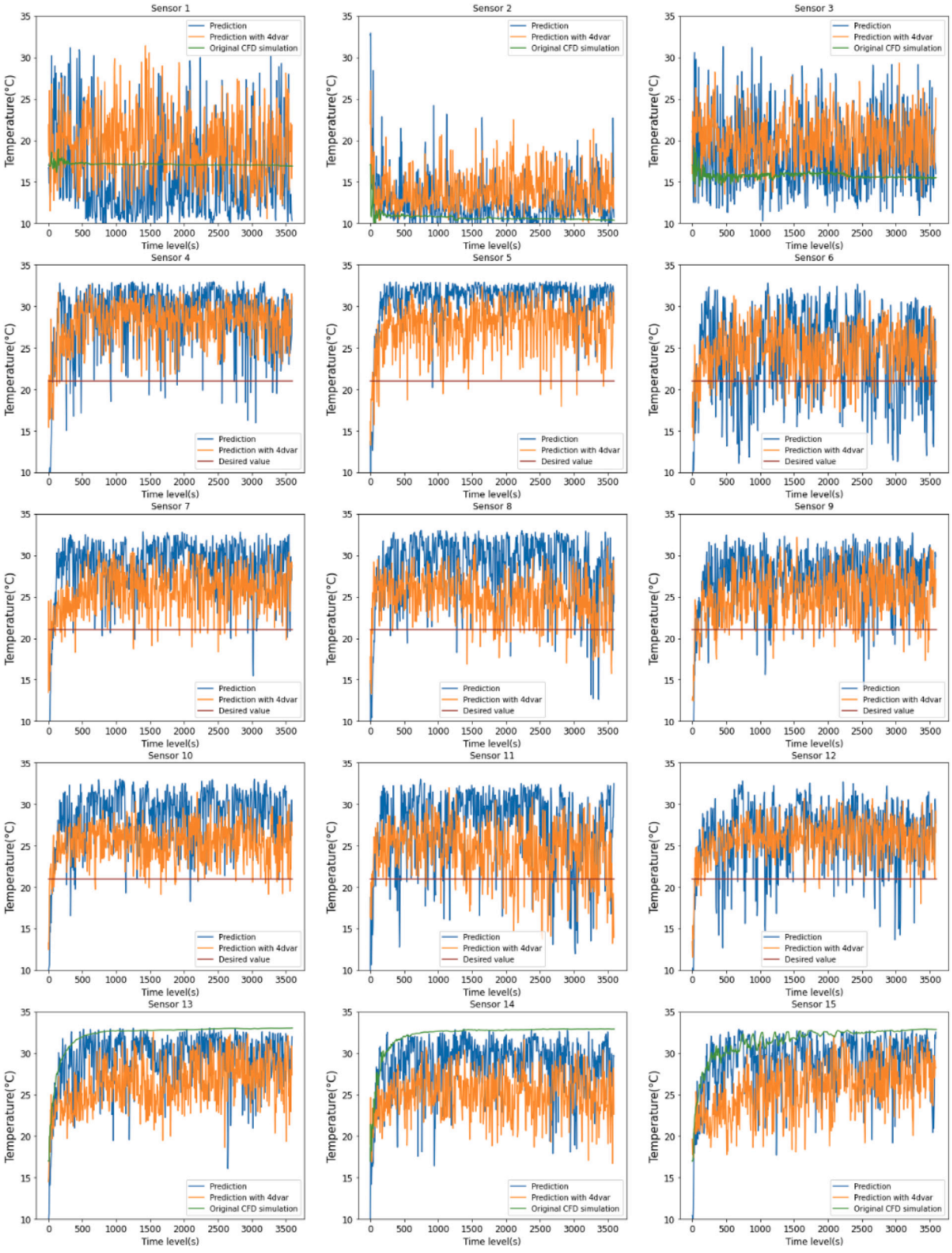


Fig. A.18. Scenario 1 control results with constrained DA and prediction method showing the resulting temperature at the sensors for algorithm comparison. Each subplot represents the temperature changing over time at a particular sensor. Blue and orange lines represent predictions from PredAAE and 4D-Var DA, respectively. Purple lines in the subplots for Sensor 4 to Sensor 12 (close to the head height of the students) represent values we wish to control. Green lines in the subplots for other sensors represent the original CFD simulation (Case 1).

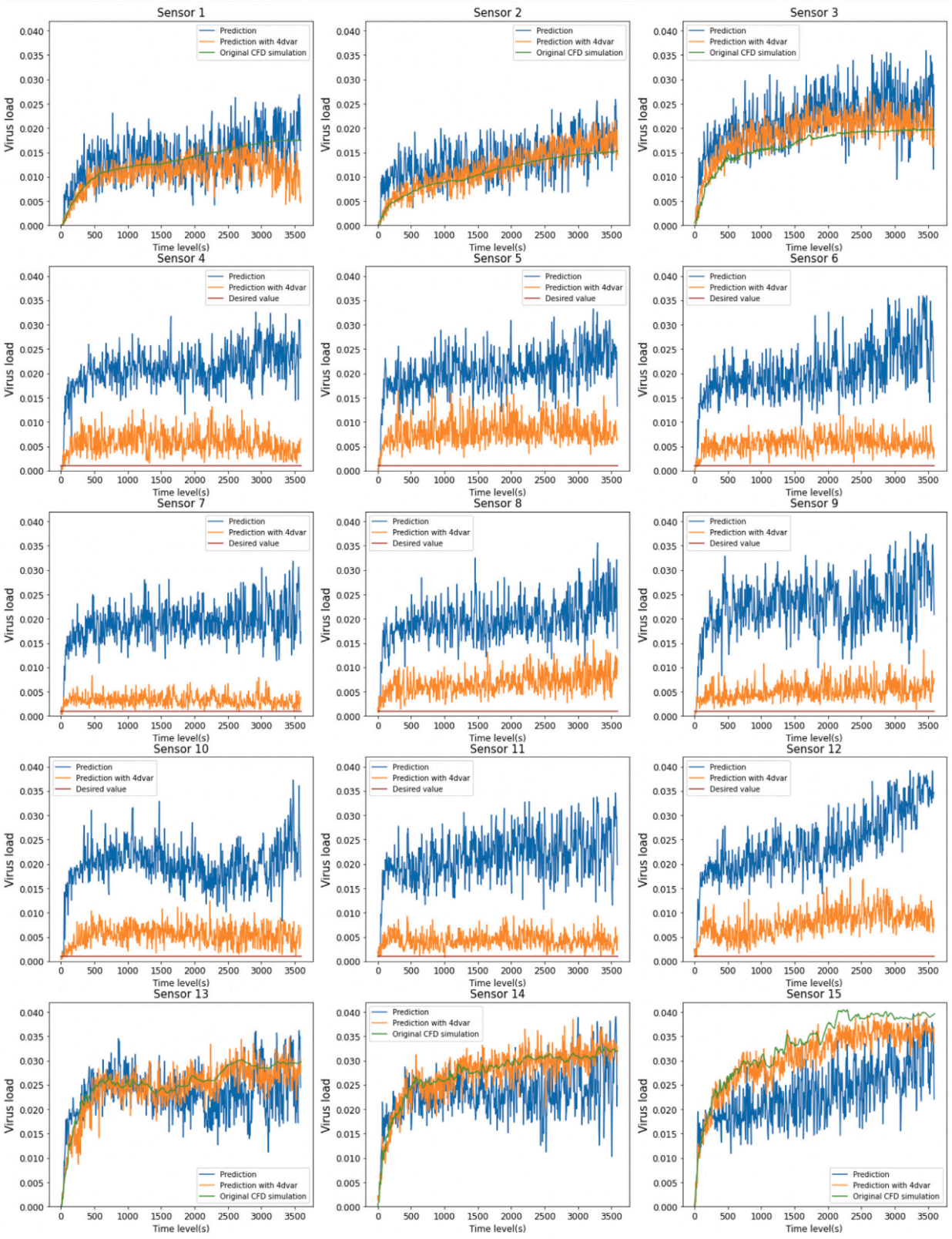


Fig. A.19. Scenario 2 control results: the viral load at the sensor locations. Each subplot represents the viral load changing over time at a particular sensor. Blue and orange lines represent predictions by PredAAE and 4D-Var DA, respectively. Purple lines in the subplots for Sensor 4 to Sensor 12 (around students) represent desired values we would like to achieve. Green lines in the subplots for other sensors represent the original CFD simulation — Case 1.

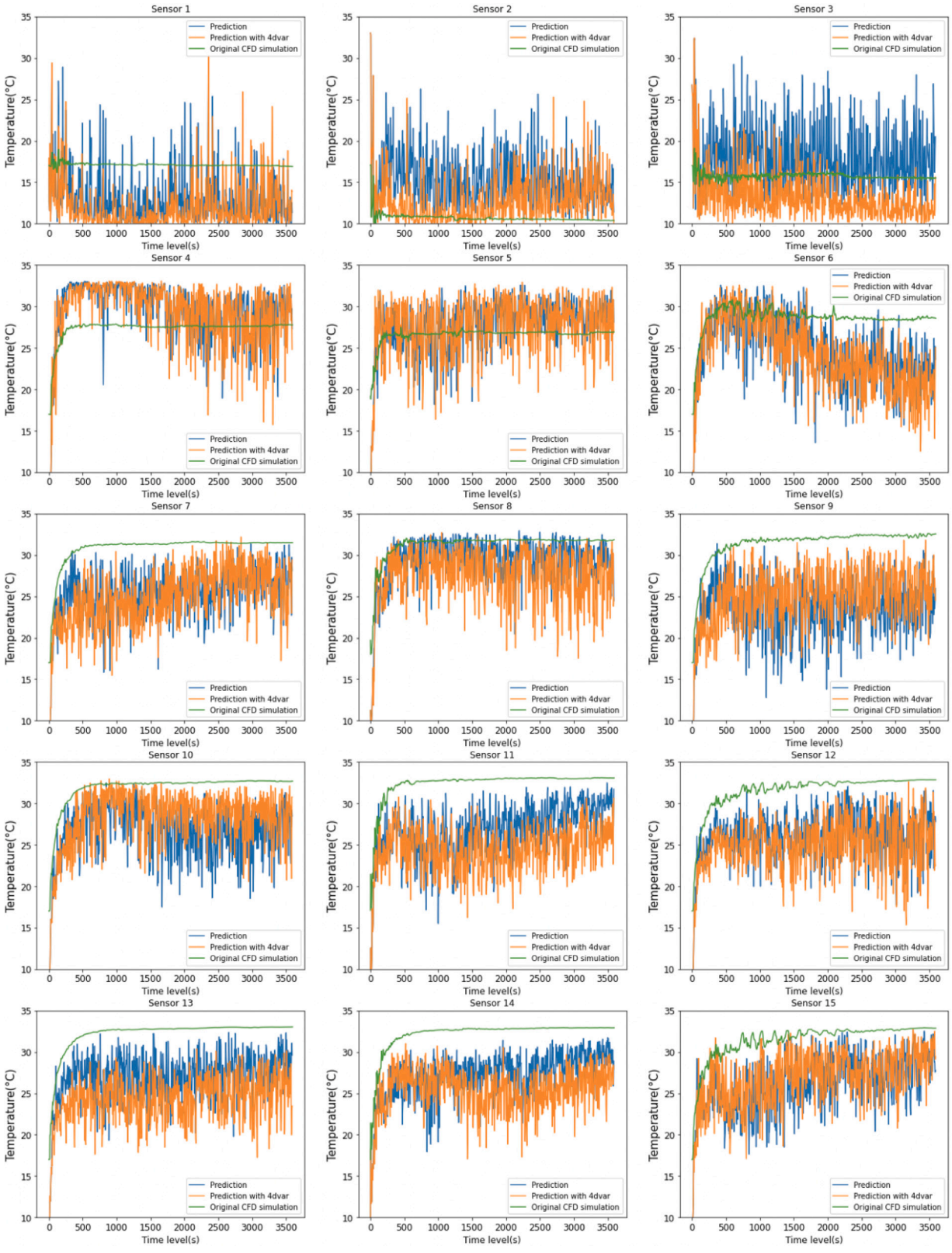


Fig. A.20. Scenario 2 controlling influence on the temperature at sensors after only controlling the viral load. Each subplot represents the temperature changing over time at a particular sensor. Blue, orange and green lines represent predictions from PredAAE, 4D-Var DA and the original CFD simulation — Case 1, respectively.

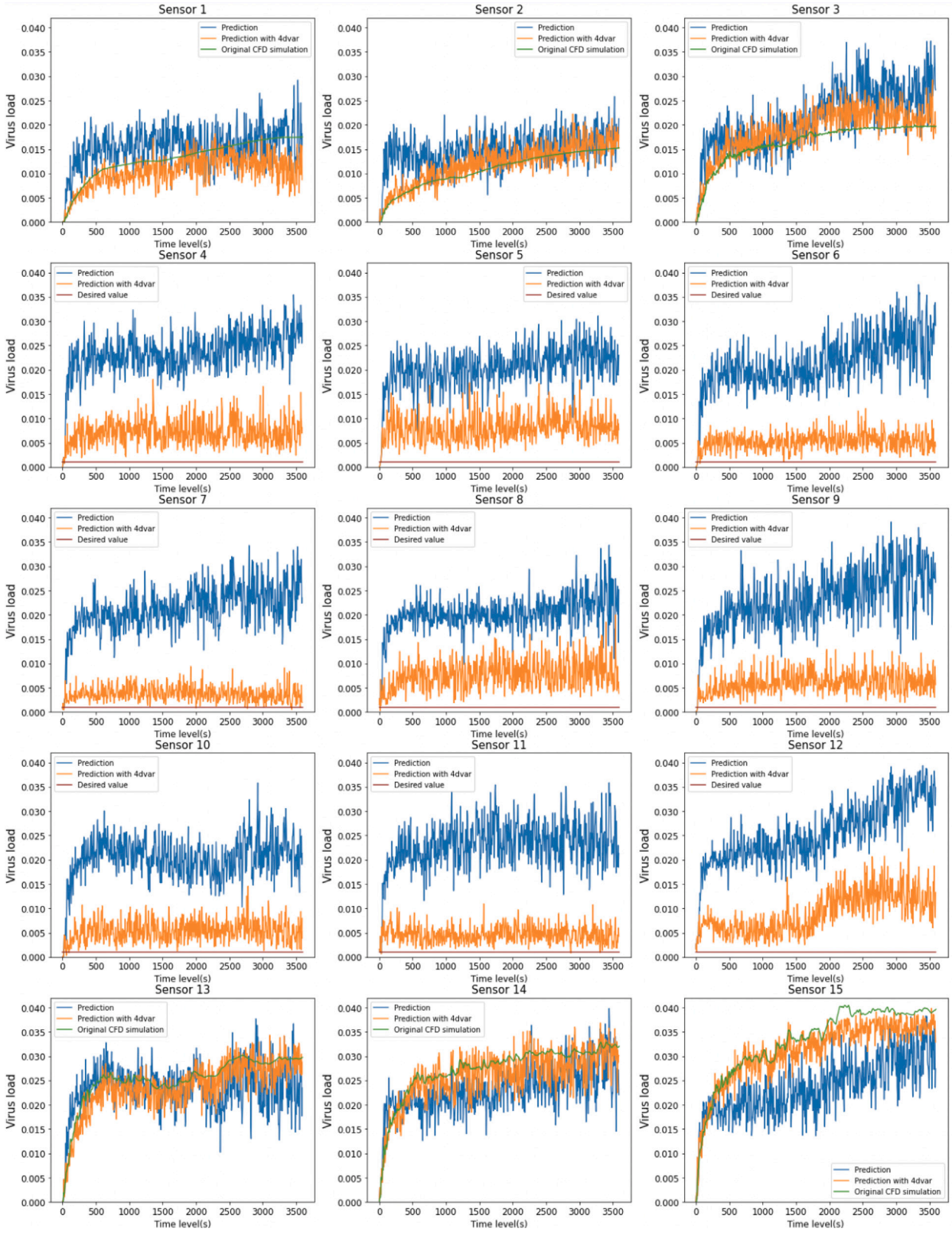
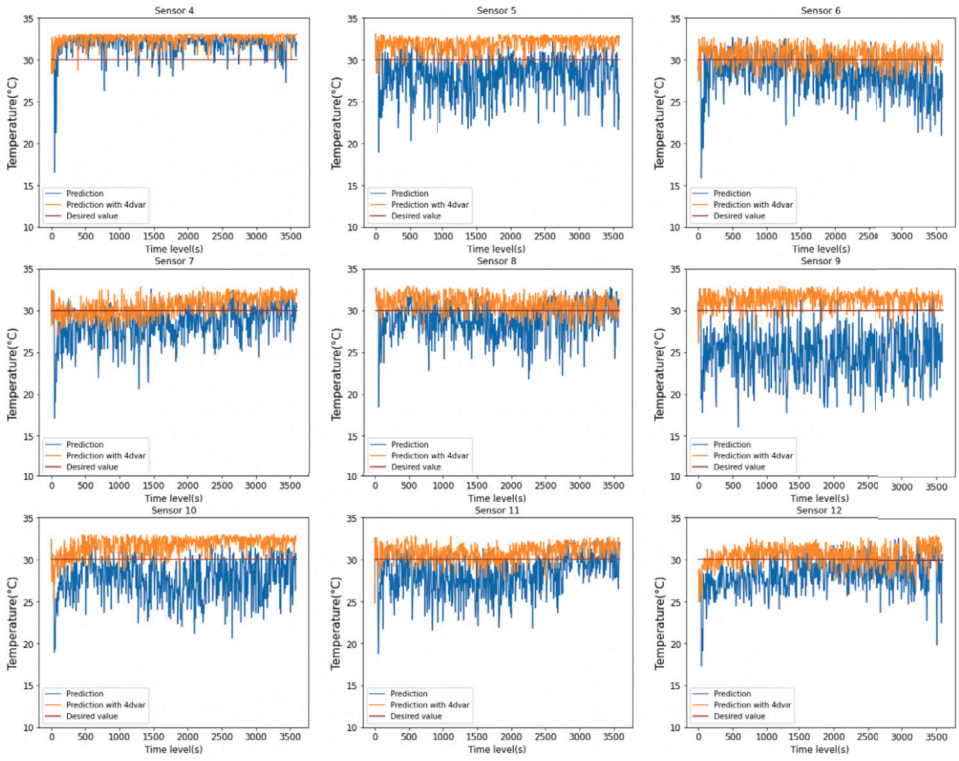
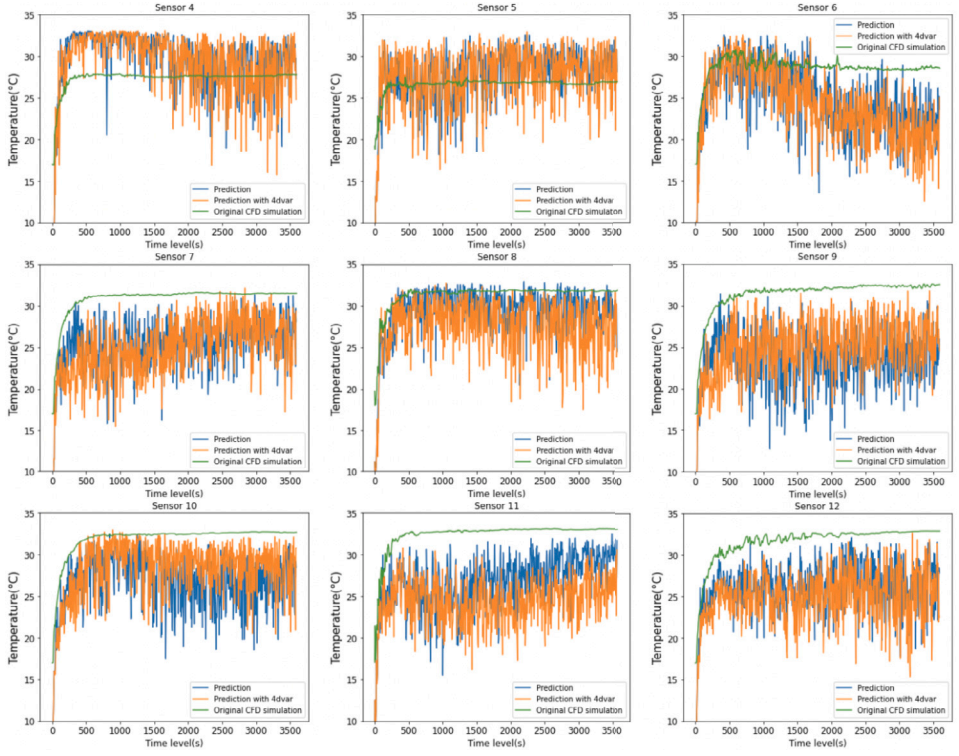


Fig. A.21. Scenario 3 control results: the viral load at sensors. Each subplot represents the viral load changing over time at a particular sensor. Blue and orange lines represent normal AAE prediction and 4D Var DA, respectively. Purple lines in the subplots for Sensor 4 to Sensor 12 (around students) represent desired values we set to control. Green lines in the subplots for other sensors represent the original CFD simulation — Case 1.



(a) Scenario 3 control results on the temperature at sensors.



(b) Scenario 2 control results on the temperature at sensors

Fig. A.22. Comparison of scenario 3 and scenario 2 control results on the temperature at controlled sensors. Each subplot represents the temperature changing over time at a particular sensor from Sensor 4 to Sensor 12. Blue and orange lines represent normal AAE prediction and 4D Var DA, respectively. (a) shows the corresponding results in scenario 3, where purple lines represent desired values we set to control. (b) shows the corresponding results in scenario 2, where green lines represent the original CFD simulation — Case 1.

Data availability

Data will be made available on request.

References

[1] N.E. Klepeis, W.C. Nelson, W.R. Ott, J.P. Robinson, A.M. Tsang, P. Switzer, J.V. Behar, S.C. Hern, W.H. Engelmann, The national human activity pattern survey (NHAPS): a resource for assessing exposure to environmental pollutants, J. Expo. Anal. Environ. Epidemiology 11 (3) (2001) 231–252, http://dx.doi.org/10.1038/sj.jea.7500165.

[2] L. Morawska, J. Allen, W. Bahnfleth, B. Bennett, P.M. Bluyssen, A. Boerstra, G. Buonanno, J. Cao, S.J. Dancer, A. Floto, F. Franchimon, T. Greenhalgh, C. Haworth, J. Hogeling, C. Isaxon, J.L. Jimenez, A. Kennedy, P. Kumar, J. Kurnitski, Y. Li, M. Loomans, G. Marks, L.C. Marr, L. Mazzarella, A.K. Melikov, S.L. Miller, D.K. Milton, J. Monty, P.V. Nielsen, C. Noakes, J. Peccia, K.A. Prather, X. Querol, T. Salthammer, C. Sekhar, O. Seppänen, S. ichi Tanabe, J.W. Tang, R. Tellier, K.W. Tham, P. Wargocki, A. Wierzbicka, M. Yao, Mandating indoor air quality for public buildings, Science 383 (6690) (2024) 1418–1420, http://dx.doi.org/10.1126/science.adl0677.

[3] M. Ciotti, M. Ciccozzi, A. Terrinoni, W.-C. Jiang, C.-B. Wang, S. Bernardini, The COVID-19 pandemic, Crit. Rev. Clin. Lab. Sci. 57 (6) (2020) 365–388, http://dx.doi.org/10.1080/10408363.2020.1783198.

[4] R.K. Bhagat, M.D. Wykes, S.B. Dalziel, P. Linden, Effects of ventilation on the indoor spread of COVID-19, J. Fluid Mech. 903 (2020) http://dx.doi.org/10.1017/jfm.2020.720.

[5] R. Wölfel, V.M. Corman, W. Guggemos, M. Seilmaier, S. Zange, M.A. Müller, D. Niemeyer, T.C. Jones, P. Vollmar, C. Rothe, et al., Virological assessment of hospitalized patients with COVID-2019, Nature 581 (7809) (2020) 465–469, http://dx.doi.org/10.1038/s41586-020-2196-x.

[6] D. Lewis, Is the coronavirus airborne? Experts can't agree, Nature 580 (7802) (2020) 175, http://dx.doi.org/10.1038/d41586-020-00974-w.

[7] K.P. Fennelly, Particle sizes of infectious aerosols: implications for infection control, Lancet Respir. Med. 8 (9) (2020) 914–924, http://dx.doi.org/10.1016/S2213-2600(20)30323-4.

[8] H.C. Burridge, S. Bontitsopoulos, C. Brown, H. Carter, K. Roberts, C. Vouriot, D. Weston, M. Mon-Williams, N. Williams, C. Noakes, Variations in classroom ventilation during the COVID-19 pandemic: Insights from monitoring 36 naturally ventilated classrooms in the UK during 2021, J. Build. Eng. 63 (2023) 105459, http://dx.doi.org/10.1016/j.jobe.2022.105459.

[9] M.S. Cattaruzza, V. Zagà, S. Gallus, P. D'Argenio, G. Gorini, Tobacco smoking and COVID-19 pandemic: old and new issues. a summary of the evidence from the scientific literature, Acta Bio Medica: Atenei Parm. 91 (2) (2020) 106, http://dx.doi.org/10.23750/abm.v91i2.9698.

[10] J. Li, G. Fan, Y. Ou, Q. Deng, Characteristics and control strategies of indoor particles: An updated review, Energy Build. 294 (2023) 113232, http://dx.doi.org/10.1016/j.enbuild.2023.113232.

[11] A. Pantazaras, M. Santamouris, S.E. Lee, M.N. Assimakopoulos, A decision tool to balance indoor air quality and energy consumption: A case study, Energy Build. 165 (2018) 246–258, http://dx.doi.org/10.1016/j.enbuild.2018.01.045.

[12] M.J. Risbeck, M.Z. Bazant, Z. Jiang, Y.M. Lee, K.H. Drees, J.D. Douglas, Modeling and multiobjective optimization of indoor airborne disease transmission risk and associated energy consumption for building HVAC systems, Energy Build. 253 (2021) 111497, http://dx.doi.org/10.1016/j.enbuild.2021.111497.

[13] EnergyPlus 8.7 — Input Output Reference, US Department of Energy, Washington, DC, USA, 2017.

[14] International Energy Agency, Buildings: Tracking buildings, 2023, https://www.iea.org/energy-system/buildings#tracking.

[15] International Energy Agency, Breakthrough agenda report, 2023, https://www.iea.org/reports/breakthrough-agenda-report-2023/buildings.

[16] S. Schiavon, A.K. Melikov, C. Sekhar, Energy analysis of the personalized ventilation system in hot and humid climates, Energy Build. 42 (5) (2010) 699–707, http://dx.doi.org/10.1016/j.enbuild.2009.11.009.

[17] A. Rackes, M.S. Waring, Using multiobjective optimizations to discover dynamic building ventilation strategies that can improve indoor air quality and reduce energy use, Energy Build. 75 (2014) 272–280, http://dx.doi.org/10.1016/j.enbuild.2014.02.024.

[18] H.P. Das, Y.-W. Lin, U. Agwan, L. Spangher, A. Devonport, Y. Yang, J. Drgoňa, S. Chong, C.J. Spanos, Machine learning for smart and energy-efficient buildings, Environ. Data Sci. 3 (2024) e1, http://dx.doi.org/10.1017/eds.2023.43.

[19] N. Salman, A. Khan, A.H. Kemp, C.J. Noakes, Indoor temperature forecast based on the lattice Boltzmann method and data assimilation, Build. Environ. 210 (2022) 108654, http://dx.doi.org/10.1016/j.buildenv.2021.108654.

[20] E.J. Teshome, F.F. Haghighat, Zonal models for indoor air flow — A critical review, Int. J. Vent. 3 (2) (2004) 119–129, http://dx.doi.org/10.1080/14733315.2004.11683908.

[21] A. Megri, F. Haghighat, Zonal modeling for simulating indoor environment of buildings: Review, recent developments, and applications, Sci. Technol. Built Environ. 13 (2007) 887–905, http://dx.doi.org/10.1080/10789669.2007.10391461.

[22] Y. Yao, K. Yang, M. Huang, L. Wang, A state-space model for dynamic response of indoor air temperature and humidity, Build. Environ. 64 (2013) 26–37, http://dx.doi.org/10.1016/j.buildenv.2013.03.009.

[23] W. Zuo, Q. Chen, Fast and informative flow simulations in a building by using fast fluid dynamics model on graphics processing unit, Build. Environ. 45 (2010) 747–757, http://dx.doi.org/10.1016/j.buildenv.2009.08.008.

[24] B. Zhao, P. Guan, Modeling particle dispersion in personalized ventilated room, Build. Environ. 42 (3) (2007) 1099–1109, http://dx.doi.org/10.1016/j.buildenv.2005.11.009.

[25] M.A.I. Khan, N. Delbosc, C.J. Noakes, J. Summers, Real-time flow simulation of indoor environments using lattice Boltzmann method, Build. Simul. 8 (2015) 405–414, http://dx.doi.org/10.1007/s12273-015-0232-9.

[26] N. Morozova, F.X. Trias, R. Capdevila, C.D. Pérez-Segarra, A. Oliva, On the feasibility of affordable high-fidelity CFD simulations for indoor environment design and control, Build. Environ. 184 (2020) 107144, http://dx.doi.org/10.1016/j.buildenv.2020.107144.

[27] H. Gao, L. Zhuang, C. Li, W. Qian, J. Dong, L. Liu, J. Liu, Fast prediction of three-dimensional indoor flow fields by a reduced dimensional deep-learning approach, Build. Environ. 270 (2025) 112517, http://dx.doi.org/10.1016/j.buildenv.2024.112517.

[28] M. Trčka, J.L. Hensen, Overview of HVAC system simulation, Autom. Constr. 19 (2) (2010) 93–99, http://dx.doi.org/10.1016/j.autcon.2009.11.019.

[29] O. Asfour, M. Gadi, A comparison between CFD and network models for predicting wind-driven ventilation in buildings, Build. Environ. 42 (2007) 4079–4085, http://dx.doi.org/10.1016/j.buildenv.2006.11.021.

[30] L. Mottet, J. Song, A.C. Short, S. Chen, J. Wu, W. Yu, J. Xiong, Q. Zhang, J. Ge, M. Liu, R. Yao, B. Li, The hot summer-cold winter region in China: Challenges in the low carbon adaptation of residential slab buildings to enhance comfort, Energy Build. 223 (2020) 110181, http://dx.doi.org/10.1016/j.enbuild.2020.110181.

[31] F.-X. Le Dimet, I.M. Navon, R. Ştefănescu, Variational data assimilation: Optimization and optimal control, in: S.K. Park, L. Xu (Eds.), Data Assimilation for Atmospheric, Oceanic and Hydrologic Applications (Vol. III), Springer International Publishing, Cham, 2017, pp. 1–53, http://dx.doi.org/10.1007/978-3-319-43415-5_1.

[32] B.R. Noack, M. Morzynski, G. Tadmor, Reduced-order Modelling for Flow Control, vol. 528, Springer Science & Business Media, 2011.

[33] P. Holmes, J. Lumley, G. Berkooz, C. Rowley, Turbulence, Coherent Structures, Dynamical Systems and Symmetry, Cambridge University Press, 2012, http://dx.doi.org/10.1017/CBO9780511919701.

[34] T. Zerihun Desta, K. Janssens, A. van Brecht, J. Meyers, M. Baelmans, D. Berckmans, CFD for model-based controller development, Build. Environ. 39 (2004) 621–633, http://dx.doi.org/10.1016/j.buildenv.2004.01.001.

[35] K. Li, H. Su, J. Chu, C. Xu, A fast-POD model for simulation and control of indoor thermal environment of buildings, Build. Environ. 60 (2013) 150–157, http://dx.doi.org/10.1016/j.buildenv.2012.11.020.

[36] M. Oulghelou, C. Beghein, C. Allery, A surrogate optimization approach for inverse problems: Application to turbulent mixed-convection flows, Comput. & Fluids 241 (2022) 105490, http://dx.doi.org/10.1016/j.compfluid.2022.105490.

[37] S. Ben Ayed, D. Kim, J.T. Borggaard, E.M. Cliff, Optimal control of indoor-air cooling in buildings using a reduced order model, Energy 116 (2016) 1191–1204, http://dx.doi.org/10.1016/j.energy.2016.10.022.

[38] Y. Xue, Z.J. Zhai, Q. Chen, Inverse prediction and optimization of flow control conditions for confined spaces using a CFD-based genetic algorithm, Build. Environ. 64 (2013) 77–84, http://dx.doi.org/10.1016/j.buildenv.2013.02.017.

[39] N. Morozova, F.X. Trias, R. Capdevila, E. Schillaci, A. Oliva, A CFD-based surrogate model for predicting flow parameters in a ventilated room using sensor readings, Energy Build. 266 (2022) 112146, http://dx.doi.org/10.1016/j.enbuild.2022.112146.

[40] S.-J. Cao, C. Ren, Ventilation control strategy using low-dimensional linear ventilation models and artificial neural network, Build. Environ. 144 (2018) 316–333, http://dx.doi.org/10.1016/j.buildenv.2018.08.032.

[41] J. Ren, S.-J. Cao, Incorporating online monitoring data into fast prediction models towards the development of artificial intelligent ventilation systems, Sustain. Cities Soc. 47 (2019) 101498, http://dx.doi.org/10.1016/j.scs.2019.101498.

[42] H.-C. Zhu, C. Ren, S.-J. Cao, Dynamic sensing and control system using artificial intelligent techniques for non-uniform indoor environment, Build. Environ. 226 (2022) 109702, http://dx.doi.org/10.1016/j.buildenv.2022.109702.

[43] A. Dmitrewski, M. Molina-Solana, R. Arcucci, CntrlDA: A building energy management control system with real-time adjustments. Application to indoor temperature, Build. Environ. 215 (2022) 108938, http://dx.doi.org/10.1016/j.buildenv.2022.108938.

[44] C. Zhuang, R. Choudhary, A. Mavrogianni, Probabilistic occupancy forecasting for risk-aware optimal ventilation through autoencoder Bayesian deep neural networks, Build. Environ. 219 (2022) http://dx.doi.org/10.1016/j.buildenv.2022.109207.

[45] L. López-Pérez, J. Flores-Prieto, C. Ríos-Rojas, Comfort temperature prediction according to an adaptive approach for educational buildings in tropical climate using artificial neural networks, Energy Build. 251 (2021) 111328, http://dx.doi.org/10.1016/j.enbuild.2021.111328.

[46] S. Haddad, A. Synnefa, M. Ángel Padilla Marcos, R. Paolini, S. Delrue, D. Prasad, M. Santamouris, On the potential of demand-controlled ventilation system to enhance indoor air quality and thermal condition in Australian school classrooms, Energy Build. 238 (2021) 110838, http://dx.doi.org/10.1016/j.enbuild.2021.110838.

[47] G. Chiesa, M. Vigliotti, Comparing mechanical ventilation control strategies for indoor air quality: Monitoring and simulation results of a school building in northern Italy, Energy Build. 322 (2024) 114665, http://dx.doi.org/10.1016/j.enbuild.2024.114665.

[48] C.E. Heaney, J. Tang, J. Yan, D. Guo, J. Ipock, S. Kaluvakollu, Y. Lin, D. Shan, B. Chen, L. Mottet, P. Kumar, C.C. Pain, Data assimilation with machine learning for dynamical systems: Modelling indoor ventilation, Phys. A 643 (2024) 129783, http://dx.doi.org/10.1016/j.physa.2024.129783.

[49] P. Kumar, N. Rawat, A. Tiwari, Micro-characteristics of a naturally ventilated classroom air quality under varying air purifier placements, Environ. Res. (2022) 114849, http://dx.doi.org/10.1016/j.envres.2022.114849.

[50] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, B. Frey, Adversarial autoencoders, 2015, http://dx.doi.org/10.48550/arXiv.1511.05644, ArXiv Preprint, arXiv:1511.05644.

[51] S. Masoumi-Verki, F. Haghighat, N. Bouguila, U. Eicker, The use of GANs and transfer learning in model-order reduction of turbulent wake of an isolated high-rise building, Build. Environ. 246 (2023) 110948, http://dx.doi.org/10.1016/j.buildenv.2023.110948.

[52] S. Mo, N. Zabaras, X. Shi, J. Wu, Integration of adversarial autoencoders with residual dense convolutional networks for estimation of non-Gaussian hydraulic conductivities, Water Resour. Res. 56 (2) (2020) http://dx.doi.org/10.1029/2019WR026082, e2019WR026082.

[53] V.L. Silva, C.E. Heaney, Y. Li, C.C. Pain, Data assimilation predictive GAN (DA-PredGAN): applied to a spatio-temporal compartmental model in epidemiology, J. Sci. Comput. 94 (2023) 25, http://dx.doi.org/10.1007/s10915-022-02078-1.

[54] M. Jolaade, V.L. Silva, C.E. Heaney, C.C. Pain, Generative networks applied to model fluid flows, in: 22nd International Conference on Computational Science, ICCS, Springer, 2022, pp. 742–755.

[55] C.E. Heaney, Z. Wolffs, J.A. Tómasson, L. Kahouadji, P. Salinas, A. Nicolle, I.M. Navon, O.K. Matar, N. Srinil, C.C. Pain, An AI-based non-intrusive reduced-order model for extended domains applied to multiphase flow in pipes, Phys. Fluids 34 (5) (2022) 055111, http://dx.doi.org/10.1063/5.0088070.

[56] C.E. Heaney, X. Liu, H. Go, Z. Wolffs, P. Salinas, I.M. Navon, C.C. Pain, Extending the capabilities of data-driven reduced-order models to make predictions for unseen scenarios: Applied to flow around buildings, Front. Phys. 10 (2022) 910381, http://dx.doi.org/10.3389/fphy.2022.910381.

[57] M. Bocquet, J. Brajard, A. Carrassi, L. Bertino, Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization, Found. Data Sci. 2 (2020) 55–80, http://dx.doi.org/10.3934/fods.2020004.

[58] A.J. Geer, Learning earth system models from observations: machine learning or data assimilation, Phil. Trans. R. Soc. A 379 (2021) 20200089, http://dx.doi.org/10.1098/rsta.2020.0089.

[59] M. Sonnewald, R. Lguensat, D.C. Jones, P.D. Dueben, J. Brajard, V. Balaji, Bridging observations, theory and numerical simulation of the ocean using machine learning, Environ. Res. Lett. 17 (7) (2021) http://dx.doi.org/10.1088/1748-9326/ac0eb0.

[60] C. Buizza, C. Quilodrán Casas, P. Nadler, J. Mack, S. Marrone, Z. Titus, C. Le Cornec, E. Heylen, T. Dur, L. Baca Ruiz, C.E. Heaney, J.A. Díaz Lopez, K.S.S. Kumar, R. Arcucci, Data learning: Integrating data assimilation and machine learning, J. Comput. Sci. 58 (2022) 101525, http://dx.doi.org/10.1016/j.jocs.2021.101525.

[61] M. Amendola, R. Arcucci, L. Mottet, C.Q. Casas, S. Fan, C. Pain, P. Linden, Y.-K. Guo, Data assimilation in the latent space of a convolutional autoencoder, in: M. Paszynski, D. Kranzlmüller, V.V. Krzhizhanovskaya, J.J. Dongarra, P.M. Sloot (Eds.), Computational Science – ICCS 2021, 2021, pp. 373–386.

[62] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: a survey, J. Mach. Learn. Res. 18 (2018) 1–43, http://dx.doi.org/10.5555/3122009.3242010.

[63] S. Linnainmaa, Taylor expansion of the accumulated rounding error, BIT Numer. Math. 16 (2) (1976) 146–160, http://dx.doi.org/10.1007/BF01931367.

[64] R.E. Wengert, A simple automatic derivative evaluation program, Commun. ACM 7 (8) (1964) 463–464, http://dx.doi.org/10.1145/355586.364791.

[65] Dyson, Pure cool TP04 air purifier, 2018, https://www.dyson.co.uk/refurbished/air-treatment/pure-cool-tower-white-silver, (Accessed 01 July 2024).

[66] Applied Modelling and Computation Group, Imperial College London, Fluidity repository, 2016, https://fluidityproject.github.io/, (Accessed 2 October 2022).

[67] Fluidity manual, 2016, https://fluidityproject.github.io/, (Accessed 2 October 2022).

[68] E. Aristodemou, T. Bentham, C. Pain, R. Colvile, A. Robins, H. ApSimon, A comparison of mesh-adaptive LES with wind tunnel data for flow past buildings: Mean flows and velocity fluctuations, Atmos. Environ. 43 (39) (2009) 6238–6253, http://dx.doi.org/10.1016/j.atmosenv.2009.07.014.

[69] D. Pavlidis, G.J. Gorman, J.L.M.A. Gomes, C.C. Pain, H. ApSimon, Synthetic-eddy method for urban atmospheric flow modelling, Bound.-Layer Meteorol. 136 (2010) 285–299, http://dx.doi.org/10.1007/s10546-010-9508-x.

[70] E. Aristodemou, L.M. Boganegra, L. Mottet, D. Pavlidis, A. Constantinou, C.C. Pain, A. Robins, H. ApSimon, How tall buildings affect turbulent air flows and dispersion of pollution within a neighbourhood, Environ. Pollut. 233 (2018) 782–796, http://dx.doi.org/10.1016/j.envpol.2017.10.041.

[71] J. Song, S. Fan, W. Lin, L. Mottet, H. Woodward, M.D. Wykes, R. Arcucci, D. Xiao, J.-E. Debay, H. ApSimon, E. Aristodemou, D. Birch, M. Carpentieri, F. Fang, M. Herzog, G.R. Hunt, R.L. Jones, C.C. Pain, D. Pavlidis, A.G. Robins, C.A. Short, P.F. Linden, Natural ventilation in cities: the implications of fluid mechanics, Build. Res. Inf. 46 (8) (2018) 809–828, http://dx.doi.org/10.1080/09613218.2018.1468158.

[72] D. Xiao, C. Heaney, L. Mottet, F. Fang, W. Lin, I. Navon, Y. Guo, O. Matar, A. Robins, C. Pain, A reduced order model for turbulent flows in the urban environment using machine learning, Build. Environ. 148 (2019) 323–337, http://dx.doi.org/10.1016/j.buildenv.2018.10.035.

[73] H. Woodward, A.K. Schroeder, C.M.A. Le Cornec, M.E.J. Stettler, H. ApSimon, A. Robins, C.C. Pain, P.F. Linden, High resolution modelling of traffic emissions using the large eddy simulation code fluidity, Atmosphere 13 (8) (2022) 1203, http://dx.doi.org/10.3390/atmos13081203.

[74] C.C. Pain, A.P. Umpleby, C.R.E. de Oliveira, A.J.H. Goddard, Tetrahedral mesh optimisation and adaptivity for steady-state and transient finite element calculations, Comput. Methods Appl. Mech. Engrg. 190 (29) (2001) 3771–3796, http://dx.doi.org/10.1016/S0045-7825(00)00294-2.

[75] L. Mottet, Indoor geometry generator manual, 2021, https://www.researchgate.net/publication/349350000_Indoor_Geometry_Generator_IGG_Manual.

[76] W. Bierman, The temperature of the skin surface, JAMA 106 (14) (1936) 1158–1162, http://dx.doi.org/10.1001/jama.1936.02770140020007.

[77] S. Mandelli, F. Borra, V. Lipari, P. Bestagini, A. Sarti, S. Tubaro, Seismic data interpolation through convolutional autoencoder, in: SEG Technical Program Expanded Abstracts 2018, 2018, pp. 4101–4105, http://dx.doi.org/10.1190/segam2018-2995428.1.

[78] H.M. Tran, F.-J. Tsai, Y.-L. Lee, J.-H. Chang, L.-T. Chang, T.-Y. Chang, K.F. Chung, H.-P. Kuo, K.-Y. Lee, K.-J. Chuang, H.-C. Chuang, The impact of air pollution on respiratory diseases in an era of climate change: A review of the current evidence, Sci. Total Environ. 898 (2023) 166340, http://dx.doi.org/10.1016/j.scitotenv.2023.166340.