Project no.:	**637268**
Project full title:	**Robust Internal Thermal Insulation of Historic Buildings**
Project Acronym:	**RIBuild**
Deliverable no.:	**D4.2 (not yet approved by the European Union)**
Title of the deliverable:	**Metamodelling approaches for hygrothermal performance of interior insulation**

Contractual Date of Delivery to the CEC:	**M54**
Actual Date of Delivery to the CEC:	**M54**
Organisation name of lead contractor for this deliverable:	**KU Leuven**
Author(s):	**Hans Janssen, Astrid Tijskens, Tianfeng Hou**
Participants(s):	**KUL**
Work package contributing to the deliverable:	**WP4**
Internal reviewers:	**Carsten Rode (DTU), John Dalsgaard Sørensen (AAU), Ernst Jan de Place Hansen (AAU)**
Nature:	**R**
Version:	**2.0**
Total number of pages:	**77**
Start date of project:	**01.01.2015**
Duration of project:	**5 years**

Abstract

This report presents the second of two deliverables related to WP 4 'Probabilistic assessment of interior insulation solutions'. The prime objective of WP4 is the development of an efficient strategy for the probabilistic hygrothermal assessment of interior insulation solutions. This deliverable reports the findings from the specific activities on model-order reduction in hygrothermal simulations in Task 4.2 (Section 2) and on neural-network-based metamodelling in Task 4.4 (Section 3).

Keyword list: hygrothermal simulation, model-order reduction, neural-network metamodelling

Table of Contents

1 Executive Summary

This report presents the second of two deliverables related to RIBuild's WP 4 '*Probabilistic assessment of interior insulation solutions*'. The key objective of WP4 is the development of an efficient strategy for the probabilistic hygrothermal assessment of interior insulation solutions. That strategy involves Monte-Carlo-based repetitions of hygrothermal simulations, and hence requires an efficient deterministic hygrothermal simulator and an efficient probabilistic assessment approach. Prior to the initiation of the RIBuild project, both a deterministic simulator and a probabilistic methodology were already available. Their joint application to interior insulation solutions does however require further developments, primarily in relation to their numerical efficiency.

The various tasks in WP4 hence focus on efficiency improvements of both the deterministic simulator and the probabilistic methodology. Tasks 4.1 and 4.2 target the efficient one-, two- and three-dimensional simulation of heat, air and moisture transfer in building components, Tasks 4.3 and 4.4 aim for the efficient probabilistic assessment of hygrothermal performances based on these simulations. In the final Task 4.5, the developments of Tasks 4.1 to 4.4 are to be brought together in a final application example. The first deliverable reported on the general developments in WP4 and on the specific results from Task 4.1 and Task 4.3. This deliverable reports the findings from the specific activities on model-order reduction in hygrothermal simulations in Task 4.2 (Section 2) and on neural-network-based metamodelling in Task 4.4 (Section 3).

Section 2 details the investigation on the performance of two model order reduction methods, Proper Orthogonal Decomposition (POD) and Proper Generalised Decomposition (PGD), with linear and nonlinear calculation objects. It is demonstrated there that the PGD method does not perform well, and the method did hence not get further consideration. The POD method, contrarily, reveals to perform very well for the linear thermal case, as only a limited number of construction modes and a small amount of snapshots are required to obtain accurate results. This method furthermore does well when extrapolated to different problems and when applied in a probabilistic simulation study. For the nonlinear hygrothermal case, the accuracy and performance of POD appears to strongly depend on the degree of nonlinearity of the calculation object. Relative to the linear cases, (much) more construction modes and snapshots are required in order to obtain an accurate result. Since the inefficiency of POD for nonlinear simulations arises from the high computational cost of repeatedly evaluating the nonlinear terms, the Discrete Empirical Interpolation Method (DEIM) is studied to further decrease the computational cost of the hygrothermal simulation. This approach again brings a significant reduction of the computational complexity.

Section 3 describes the examination of the performance of neural networks for metamodeling of hygrothermal behaviour. To that aim, a convolutional neural network (CNN) is trained to predict the behaviour of a 1D masonry wall, taking into account all probabilistic input parameters. The network is trained to predict hygrothermal time series (temperature, relative humidity and saturation degree), which are subsequently used to predict damage using existing damage prediction models (moist freeze-thaw cycles, mould growth, wood decay). The results show that the CNN is able to predict the hygrothermal outputs accurately. The damage predictions, based on the hygrothermal predictions by the network, are accurate in most cases as well. However, in some cases, the mould index and wood mass loss at the wooden beam ends are underestimated, and the indicators for frost damage score low, due to relatively high errors. But, although the performance indicators score less well, these deviations are unlikely to influence decision making significantly.

2 Introduction

This report presents the second of two deliverables related to RIBuild's WP 4 '*Probabilistic assessment of interior insulation solutions*'. The key objective of WP4 is the development of an efficient strategy for the probabilistic hygrothermal assessment of interior insulation solutions. The latter must proceed via numerical simulations, since a multitude of scenarios need to be judged quickly and cheaply. The overall strategy involves Monte-Carlo-based repetitions of hygrothermal simulations, and an efficient strategy hence requires an efficient deterministic hygrothermal simulator and an efficient probabilistic assessment methodology.

Prior to the initiation of the RIBuild project, both a deterministic simulator and a probabilistic methodology were already available, from previous research activities by respectively the Institute of Building Climatology of TU Dresden and the Building Physics Section of KU Leuven. Their joint application to interior insulation solutions did however require further developments, primarily in relation to their numerical efficiency. At that point in time indeed, the computational costs of such assessment of interior insulation solutions would highly exceed the time constraints of the RIBuild project.

The various tasks in WP4 hence focus on efficiency improvements of both the deterministic simulator and the probabilistic methodology. Tasks 4.1 and 4.2 target the efficient one-, two- and three-dimensional simulation of heat, air and moisture transfer in building components, Tasks 4.3 and 4.4 aim for the efficient probabilistic assessment of hygrothermal performances based on these simulations. In the final Task 4.5, the developments of Tasks 4.1 to 4.4 are to be brought together in a final application example.

The original overall set-up of WP4 can be summarized as follows:

Tasks	***participants***
• T4.1: an **efficient 3D HAM** simulation model	TUD
• T4.2: **decomposition techniques** in HAM simulation	KUL
• T4.3: **sequential sampling** for Monte Carlo analysis	KUL
• T4.4: **surrogate models** for HAM simulation	KUL
• T4.5: **exemplary application** illustration	KUL/TUD/AAU/HES-SO
Deliverables	
• D4.1: **basic probabilistic analysis** (T4.1, T4.3)	KUL/TUD
• D4.2: **metamodelling approaches** (T4.2, T4.4)	KUL

The original objectives and methodology for the different WP4 tasks are formulated as follows:

- **Task 4.1: Numerical efficiency of hygrothermal simulation**

  Hygrothermal simulations with Delphin take on a double role in WP4: they may be applied directly in the probabilistic methodology for simple configurations, while they may be applied indirectly as a reference for the reduced-order and/or surrogate modelling for more complex configurations.

  The currently available Delphin 5.8 allows one- and two-dimensional hygrothermal simulations making use of direct solvers for banded systems, with sufficient functionality and efficiency for simple configurations. The more complex configurations require three-dimensional functionalities and iterative solvers with preconditioning.

- **Task 4.2: Reduced-order models in hygrothermal simulation**

Notwithstanding the potential efficiency gains targeted in Task 4.1, direct numerical simulation of heat, air and moisture transfer in building components will remain (relatively) computationally expensive, and alternatives need to be investigated. Task 4.2 aims for 'model order reduction' methods in that respect.

The literature offers multiple references on the successful application of model order reduction methods for hygrothermal simulation, but most of these applications remain restricted to cases where linear heat and vapour transfer are the dominant mechanisms. Its applicability for the far more strongly non-linear liquid transfer – a dominant moisture transfer mechanism in most hygrothermal assessments of the internal insulation solutions – remains unknown, and that will form the prime research target of this task.

- **Task 4.3: Sequential sampling in probabilistic assessment**

  A sequential sampling strategy, merging replicated optimised Latin hypercubes and bootstrap-based error quantification, is presently the state-of-the-art concerning convergence efficiency and monitoring and forms the backbone of the probabilistic strategy. As currently independent optimal Latin hypercubes are combined, the rate of convergence is proportional to $n^{-0.5}$ with n the number of Monte Carlo samples. Theoretically though, quasi-Monte Carlo sampling schemes allow (far) higher convergence rates.

  To that aim, the replicated Latin hypercubes and bootstrap-based errors are to be abandoned in favour of low-discrepancy-based approaches combined with replication-based standard errors. In this task, different possible candidates for the actual low-discrepancy-based sampling schemes will be evaluated and compared, and their combinability with replication-based error quantification will equally be assessed.

- **Task 4.4: Surrogate modelling of hygrothermal performance**

  Static metamodels are at present a crucial element of the probabilistic methodology, in order to reduce the computational cost of the multi-level Monte Carlo sampling scheme. This surrogate modelling has however been limited to static models: the models consider stationary outcomes only, and relate these directly to the variable input parameters. Their static character does however restrict their applicability.

  Dynamic surrogate models will greatly augment the potential of metamodels: explicit inclusion of the temporal variations of the hygrothermal conditions in the surrogate models makes them far more flexible. To this aim, recurrent neural networks are the first option to be examined. Alternatively approaches based on singular value decomposition or principal component analysis can be considered. Complementarily, the reduced-order models resulting from Task 4.2 may offer additional possibilities.

- **Task 4.5 Exemplary application on internal insulation case**

  WP6 targets a comprehensive comparative assessment of internal insulation solutions based on life cycle cost, combining the probabilistic assessment of hygrothermal performance developed in WP4 with the quantification of life cycle costs of internal insulation's benefits and damages formulated in WP5. To support the application of the WP4 techniques in such holistic evaluation, an illustrative application forms the concluding element of this work package.

  In this illustration, a number of internal insulation solutions for external walls with embedded wooden beams, openings in the building envelope such as windows and doors etc., are evaluated via the developed probabilistic hygrothermal assessment strategy.

This deliverable reports on the outcomes from the research activities within Task 4.2 (Section 2) and Task 4.4 (Section 3). Section 2 reports on the investigation of model-order-reduction methods for applications in hygrothermal performance assessment, particularly the Proper Orthogonal decomposition and the Proper Generalised Decomposition. Section 3 describes the examination of neural networks for the emulation of numerical hygrothermal simulation, with a focus on predicting the temporal evolution of temperatures, moisture contents in (insulated) masonry walls.

3 Model-order-reduction for hygrothermal simulations

3.1 General Information

The research on model-order reduction for hygrothermal simulations was carried out by Tianfeng Hou, who started in October 2015 as a Ph.D student at KU Leuven. This research was developed mainly during the period from July 2017 to June 2019, on which roughly 22 person-months were spent. In this section, a brief overview of the progress of his work is provided. First, a detailed literature review about the two model order reduction methods – proper orthogonal decomposition (POD) and proper generalized decomposition (PGD) – is conducted. Next, to obtain a deeper understanding of the fundamental properties and practical applications of POD and PGD, the two model order reduction techniques and the Finite Element Method (FEM) are implemented for simulating the linear heat transfer in a massive masonry wall, and several deterministic linear case studies about investigating the accuracies of the two reduced models and the robustness of using POD for simulating different problems are performed. At the end of this interval, a paper about different model order reduction methods for the simulation of wall heat transfer was presented in the 7th international building physics conference (IBPC 2018). Next, to study the performance of POD and PGD for Monte Carlo based probabilistic analysis, a probabilistic case study on quantifying the distribution of the heat losses via the massive masonry wall is performed. Similar as for the linear HAM simulation, for investigating the use of POD and FEM for simulating (strongly) nonlinear hygrothermal performance, both methods are implemented in a computer code. In addition, several benchmark case studies (HAMSTAD benchmark 2, 3 and 4) are performed to analyse the accuracy and robustness of using POD for simulating different problems. At the end of this interval, a paper about using POD for the simulation of (strong) nonlinear hygrothermal performance has been submitted and accepted by the Central European Symposium on Building Physics in 2019. At last, to further reduce the computational time, the combination of POD with discrete empirical interpolation method (DEIM) is being investigated.

3.2 Introduction

Today, 30% of the European building stock consists of 'historic' buildings built prior to World War II (Ribuild.eu, 2018). These buildings are typically far less energy-efficient than new buildings, and they hence account for a large share of the total energy consumption of buildings. One important measure to reduce their energy consumption is to install internal insulation. However, internal insulation is often associated with moisture damage, and much care should be taken when applying this solution. This report is part of the EU H2020 RIBuild project, which aims at developing effective and comprehensive guidelines for internal insulation in historic buildings. Given that a multitude of scenarios and factors can be easily evaluated with numerical analyses, the use of numerical simulations for hygrothermal performance assessment tends to be the best option. However, the conventional numerical methods based on the space and time discretization can be very time consuming due to the high non-linearity of the equations, the complex multi-dimensional spatial domains and the long-time intervals required. Therefore, a faster surrogate model is highly desired.

Instead of using the standard numerical models, Van Gelder et al. (2014) employed statistical regression and interpolation based surrogate models (such as polynomial regression, Kriging etc.) to reduce the simulation time. However, these simpler surrogate models can only deliver static results: for heat and moisture transfer through a building component, they may predict the yearly total heat loss or yearly average moisture content, but not the evolution of temperature or moisture content profiles over time. Therefore, to obtain the dynamic behaviour with a simplifying surrogate model, model order reduction techniques as alternatives of the statistical surrogate modelling are investigated.

In this report, two model order reduction methods – proper orthogonal decomposition (POD) and proper generalized decomposition (PGD)) – are investigated and compared. The first method belongs to a family of a posteriori methods, as it is built based on the earlier results of the original time-consuming model. The second method, on the other hand, is an a priori method which can be established by a suitable iterative process. Instead of the standard finite element method (FEM), both POD and PGD are used to simulate the building hygrothermal performance, exemplified through both linear and nonlinear case studies.

Below, the introduction of POD and PGD are put forward, with focus on the potential use for modelling of wall heat and moisture transfer. Next, the results of using POD and PGD for simulating the wall performances are respectively presented by both linear and nonlinear scenarios, and discussions with respect to the interpretation of their accuracies follow. Finally, a conclusion on synthesizing the superiorities and current limitations of model order reduction methods are put forward.

Nomenclature			
ρ	Density of the wall $[kg/m^3]$	h_e	Exterior surface transfer coefficient $[\mathrm{W/m^2K}]$
c	Specific heat capacity $[J/kgK]$		
λ	Thermal conductivity $[W/mK]$	h_i	Interior surface transfer coefficient $[\mathrm{W/m^2K}]$
u	Temperature $[°C]$		
t	Temporal variable $[s]$	L	Wall thickness $[m]$
x	Spatial variable $[m]$	POD	Proper orthogonal decomposition
$u_{exterior}$	Exterior temperature $[°C]$	PGD	Proper generalized decomposition
$u_{interior}$	Interior temperature $[°C]$	FEM	Finite element method

3.3 POD and PGD for modelling wall hygrothermal performance

The hygrothermal performance of a building component can be assessed by analysing the transfer of heat and moisture through building materials, and it requires to get numerical simulation results for the coupled heat and moisture transport equation combined with information on the component geometry, its material properties and its boundary conditions. The conventional hygrothermal simulation models are typically based on numerical simulation methods applying a space and a time discretization, for instance, the FEM. As mentioned before, these standard numerical methods can be very time consuming, due to the high number of unknowns after spatial and temporal discretization and non-linearities also complicate the matter because of the need for small time steps and/or iterations. Therefore, in this report two model order reduction methods (POD and PGD) were investigated which reduce the degrees of freedom of the complex system while still mimic the dynamic behaviour (such as the time evolution of temperatures, moisture content…).

3.3.1 Proper orthogonal decomposition

The POD method was first proposed by Kosambi (1943), and has been successfully applied in a variety of engineering fields, such as image processing, signal analysis, data compression and recently in building physical engineering (Tallet et al., 2015; Tallet et al., 2017; Berger et al., 2018). POD is also known as Karhunen - Loeve decomposition, principal component analysis, or singular value decomposition, and the connections of these three methods are provided by Liang et al. (2002). A brief tutorial of POD can be found in (Chatterjee, 2000), while a detailed introduction of its theory and related application for modelling heat transfer process are respectively presented by Liang et al. (2002) and Fic et al. (2005).

The key idea of POD is approximating a high-dimensional process by its 'most relevant information'. In this report, the 'most relevant information' is extracted from earlier simulation results of the original numerical model with singular value decomposition (SVD). After the SVD, the needed POD modes are constructed by selecting the k most important basis vectors, here $k \ll N$, where N is the number of the spatial mesh elements. As a result, these POD modes can be used to construct a reduced model for simulating similar problems (with, for instance, variations in the boundary conditions, in the material properties or in the component geometry, or with a longer simulation period). The reader is referred to (Agudelo, 2009) for a detailed procedure of using POD to construct the reduced-order-model.

3.3.2 Proper generalized decomposition

Despite being able to provide a reduced basis and save the computational time when simulating similar problems, the POD method has an important drawback: to construct a POD, 'a priori knowledge' – the earlier simulation results of the large original model, commonly denoted as snapshots – is needed. This disadvantage in turn gives an extra computational cost and limits its application to 'different but similar problems'. On the contrary, Ladeveze (1985) proposed a different strategy, called 'radial approximation'. This method is based on the hypothesis that the solution of the considered problem is given by a finite sum representation:

$$u(\boldsymbol{x}, t) = \sum_{i=1}^{N} \boldsymbol{X}_i(x) \cdot T_i(t) \tag{1}$$

Here, u is the solution of the target problem, $\boldsymbol{X}_i$ usually stands for the spatial coordinates, T_i represent the temporal parameter. Next, injecting equation (1) to the weighted residual formulation of the coupled heat and moisture transport equation and starting from an initial point based on the related initial and boundary conditions, the solution $u(\boldsymbol{x}, t)$ can be constructed by successive iterative enrichment methods. The procedure is stopped when the convergence criteria are reached. As a result, this strategy allows to approximate the solution without any 'a priori knowledge'. Inspired by this strategy, Ammar et al. (2006) generalized this method to the multidimensional situation and named it proper generalized decomposition (PGD). A detailed tutorial of PGD is proposed by Chinesta et al. (2013), and an application of PGD for simulating thermal processes is provided by Pruliere et al. (2013). In addition, two reviews of PGD are provided by Chinesta et al. (2010) and Berger et al. (2016), with attention for general and building physical engineering applications respectively.

3.4 Linear scenarios

For investigating the performance of POD and PGD for hygrothermal simulations, we start from the simplest scenarios, where both of the material properties and boundary conditions are linear. As the reference situation prior to retrofit is often a massive masonry wall, and that configuration is adopted here as calculation object. To judge the feasibility of internal insulation in historic buildings, the hygrothermal performances of internally insulated massive walls – heat loss, mould growth, wood rot need to be investigated (Vereecken et al. 2015). To simplify the calculation complexity in this section, the performance assessment is limited to the heat losses through the wall. Since quantifying the heat loss requires solution of the temperature profiles of the wall, both the temperature profiles and heat losses over the entire year need to be quantified. To do so, the thermal behaviour of the wall is simulated with FEM, POD and PGD, wherein the conductive heat transfer equation

$$\rho c \frac{\partial u}{\partial t} - \lambda \frac{\partial^2 u}{\partial x^2} = 0 \tag{2}$$

is solved for simple interior and exterior boundary conditions:

$$\lambda \frac{\partial u}{\partial x} = h_e(u(0) - u_{exterior}) \tag{3}$$

$$-\lambda \frac{\partial u}{\partial x} = h_i(u(L) - u_{interior}) \tag{4}$$

3.4.1 Deterministic case study

In this section, a deterministic case study is performed to evaluate the accuracies of POD and PGD. To do so, the conductive heat transfer equation (2) is simulated respectively by the FEM and different model order reduction methods, under specific boundary conditions, material properties and component geometry, with a simulation period of one year. The simulation result of the FEM is taken as the reference solution: more specifically, this reference solution is calculated by the FEM with 200 mesh elements and a fixed time step of one hour. As mentioned in section 3.3, the reduced model obtained by POD is usually used for simulating problems with different boundary conditions, material properties, component geometry, and/or for a longer simulation period. On the other hand, being an a posteriori method, PGD can directly provide approximate solutions of the calculation object without any ‘a priori knowledge’. In this section, the accuracies of PGD and full POD is judged, the term ‘full POD’ here referring to the POD basis that is trained on snapshots of the whole simulation period. The methods are then used for simulating problems with the same boundary conditions, material properties and component geometry. In the second part, the accuracy of using POD for simulating problems that deviate from the training snapshot simulation is illustrated.

3.4.1.1 Input parameters

For the comparison case study of PGD and full POD, the detailed information of the input parameters is mentioned here. For the material properties, the density ρ, thermal capacity c and conductivity λ of the wall are 2100 kg/m^3, 870 J/kgK and 1 W/mK. The boundary conditions are kept restricted to combined convection and radiation, governed by the air temperature. The outdoor air temperature $u_{exterior}$ is taken from climate data for Gaasbeek (Belgium), the indoor air temperature $u_{interior}$ is kept at 20 °C. The interior and exterior surface heat transfer coefficients are $h_i = 8\ \mathrm{W/m^2K}$ and $h_e = 25\ \mathrm{W/m^2K}$. In relation to the component geometry, the thickness of the wall L is 0.2 m.

With respect to the use of POD for simulating problems with a longer simulation period, several POD models are constructed by using the same input values mentioned above, but with snapshots from shorter time intervals: one month, one day, half day, six hours, three hours. All the training scenarios are performed 12 times: once for every month, where the one day, half day, six hours, three hours are always taken at the start of the month.

On the other hand, in order to investigate the performance of POD for simulating problems different in boundary conditions, material properties and/or component geometry, 6 scenarios are considered. In the first 5 scenarios, one element of the input parameters is modified relative to the previous case study, a detailed summary of which is summarized in Table 1. For the 6th scenario, all changes of the first 5 scenarios are imposed simultaneously. It should be mentioned that, all the POD models in these 6 scenarios are constructed by a matrix of snapshots of one single day (the first day of each month).

Table 1: Training properties and testing properties of the 6 scenarios

Scenario	Element modified	Training properties	Testing properties
1	Material properties	ρ: 2100 kg/m^3 & λ: 1 W/mK	ρ: 1700 kg/m^3 & λ: 0.6 W/mK
2	Transfer coefficients	h_i: 6 $\mathrm{W/m^2K}$ & h_e: 10$\mathrm{W/m^2K}$	h_i: 10 $\mathrm{W/m^2K}$ & h_e: 40$\mathrm{W/m^2K}$
2	Wall thickness	L: 0.2m	L: 0.5m
4	Interior temperature	$u_{interior}$: 15°C	$u_{interior}$: 25°C
5	Exterior temperature	$u_{exterior}$: $Roma$ ($Italy$)	$u_{exterior}$: $Arjeplog$ ($Sweden$)

3.4.1.2 Result of the comparison of PGD and full POD

In this section, the accuracies of PGD and full POD are judged. To evaluate the performance of POD and PGD as a function of the number of modes, both of the POD and PGD models are calculated with 1 to 15 modes. To compare the accuracy of PGD and full POD, the average temperature difference (equation (5)) between the FEM solution and the solutions of the two reduced models, as a function of the number of modes, is shown in Figure 1. For getting a more direct view of the performance of full POD and PGD methods, different profiles of temperature simulated by FEM, POD and PGD are compared at different moments, and the result is presented in Figure 2. In addition, since in practice the cumulated heat loss of entire year is usually considered as an indicator of the thermal performance of the wall, the relative deviation of heat losses between the reference solution and different reduced models (equation (6)) are shown in Figure 3.

$$\frac{\sum_{\substack{1\leq i\leq N_x \\ 1<j<N_t}} |u_{FEM}(i,j) - u_{reduced}(i,j)|}{N_x \times N_t} \tag{5}$$

$$\frac{|HL_{ref} - HL_{reduced}|}{HL_{ref}} \tag{6}$$

Here, u_{FEM} , $u_{reduced}$ are respectively the temperature values of the FEM and the different reduced models. N_x and N_t are respectively the number of spatial and temporal discretization elements of different simulation methods. HL_{ref} and $HL_{reduced}$ respectively refer to the heat loss of the reference solution and the different reduced models.

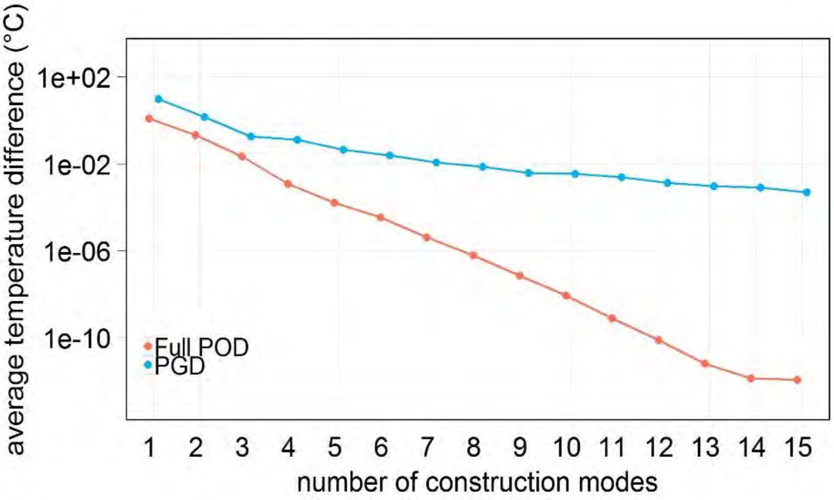


Figure 1: Average temperature difference between reference solution and full POD and PGD approximations.

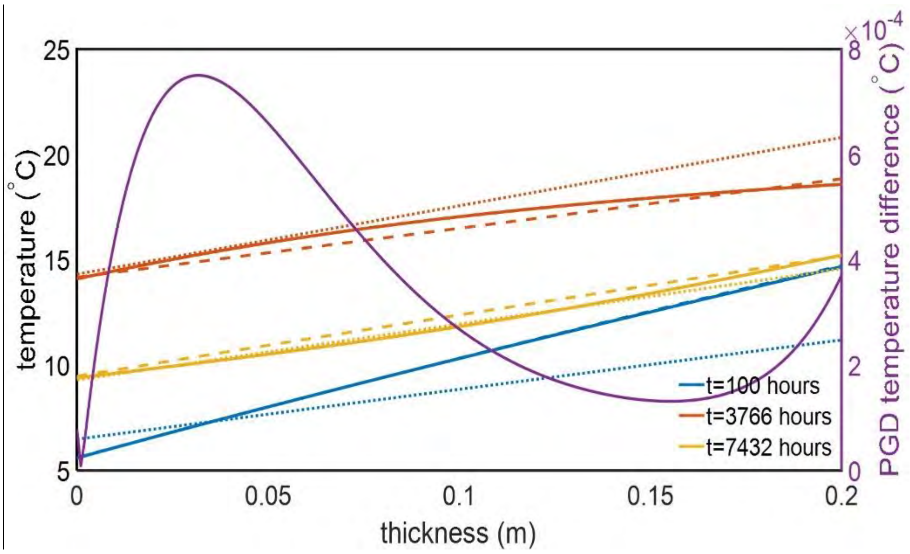


Figure 2: Temperature profiles of the reference solution, PGD with 15 modes and full POD with 15 modes (solid lines), full POD with 2 modes (dashed lines) and PGD with 2 modes (dotted lines). The purple line indicate the absolute temperature difference between the reference solution and the solution of PGD with 15 modes, at the moment of t=100 hours. Thickness = 0 refers to outside.

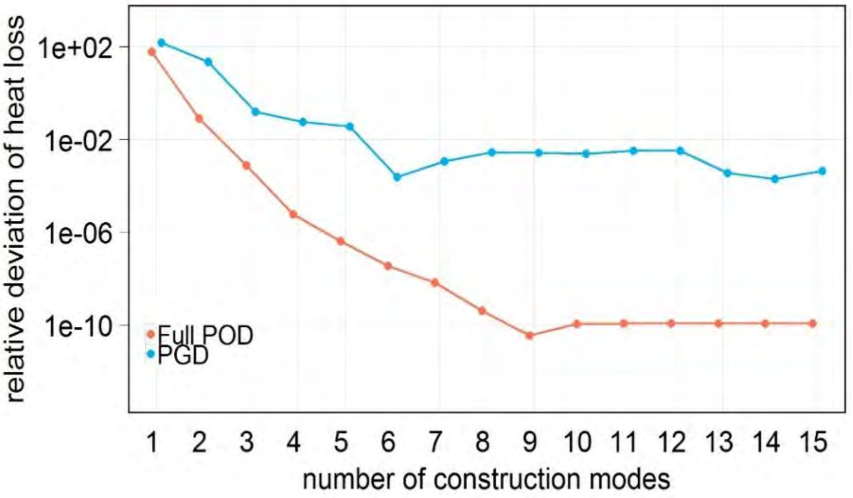


Figure 3: Relative deviation of heat loss between reference solution and full POD and PGD approximations.

Figure 1 illustrates that, the accuracies of full POD and PGD increase as the number of their construction modes raises. It is also shown that, the accuracy of the full POD model increases faster than the PGD model, therefore with the same number of construction modes, the accuracy of the full POD model tends to be higher than the PGD model. In addition, one can see that for both models, a relatively accurate result can be reached with a sufficient number of construction modes: the relative errors for the two reduced-order models go below 1% even with a limited number of modes (less than 8 modes).

Confirming the result of Figure 1, in relation to the temperature profiles, Figure 2 shows visually there is no difference between the reference solution, the PGD approximation with 15 modes and the full POD approximation with 15 modes. In addition, the right y-axis of Figure 2 indicates that, the absolute temperature difference between the reference solution and the solution of PGD with 15 modes is already very small (less than 8×10^{-4}). Moreover, as the absolute temperature difference between the reference solution and full POD with 15 modes is smaller than 1×10^{-10}, this result is not presented in Figure 2. These outcomes confirm that with a sufficient number of construction modes both full POD and PGD can provide an accurate result. On the other hand, larger differences can be respectively found between the reference solution and the solution calculated by the full POD and PGD with only 3 modes.

Comparing with Figure 1, a very similar result can be found in Figure 3 – in relation to the relative deviations of heat loss from the reference solution, the results of the two reduced models decrease as the number of their construction modes increase and these relative deviations can be reached below 1% with a very limited number of modes. However, for both of the two reduced models, after 9 modes this improvement becomes negligible. These findings indicate that the performance of all the model-order-reduction methods do not vary much for quantifying the heat loss instead of calculating the temperature profile.

3.4.1.3 Results of using POD for different problems

In section 3.4.1.2, it was shown that the POD model constructed based on the full set of snapshots can provide a very accurate result when simulating the same system. However, in practice this recalculation of the same problem does not add any value, as all the information is already included in the matrix of snapshots. Instead, the use of POD to construct a reduced model for simulating different problems is investigated in relation to:

1. longer simulation period.
2. variations in the boundary conditions, material properties and component geometry.

Longer simulation period

To compare the accuracy of the POD models constructed by different simulation periods, the relative deviations of heat losses between the reference solution and POD models with the snapshots taken from different shorter simulation periods are shown in Figure 4.

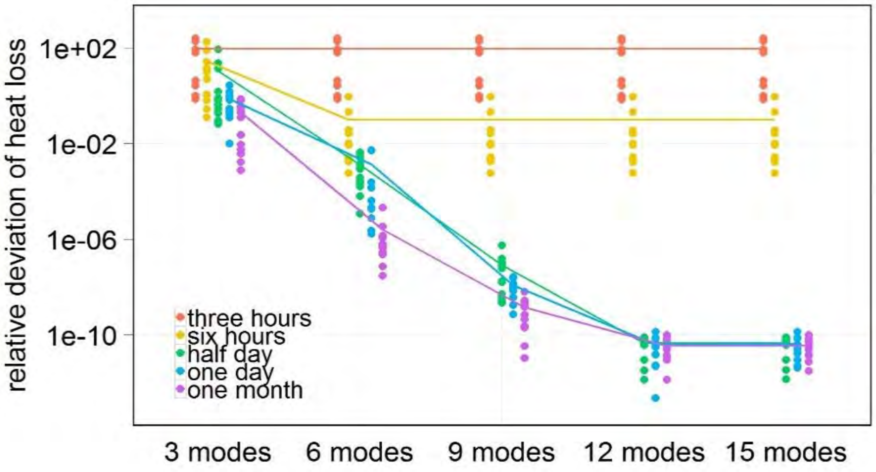


Figure 4: Relative deviations of heat losses between reference solution and POD models constructed by the snapshots of different simulation periods.

Figure 4 illustrates that, except for the POD constructed from the 3 and 6 hours' snapshots, the accuracies of the other POD models increase as the number of their construction modes raises. However, after 9 modes this improvement becomes negligible. It is also shown that the performance of using POD for simulating problems with longer simulation period can provide an accurate result. Only when the number of snapshots is really insufficient (from simulation intervals three or six hours), an inaccurate result may be obtained.

Boundary conditions, material properties, component geometry

Until now, the POD model is constructed for solving a problem by making use of the boundary conditions, material properties and component geometry which are described in section 3.4.1.1. To further investigate the performance of using POD for simulating problems varying in these factors, the relative deviations of heat losses between the reference solution and the POD approximations in relation to the 6 scenarios mentioned in section 3.4.1.1 are shown in Figure 5.

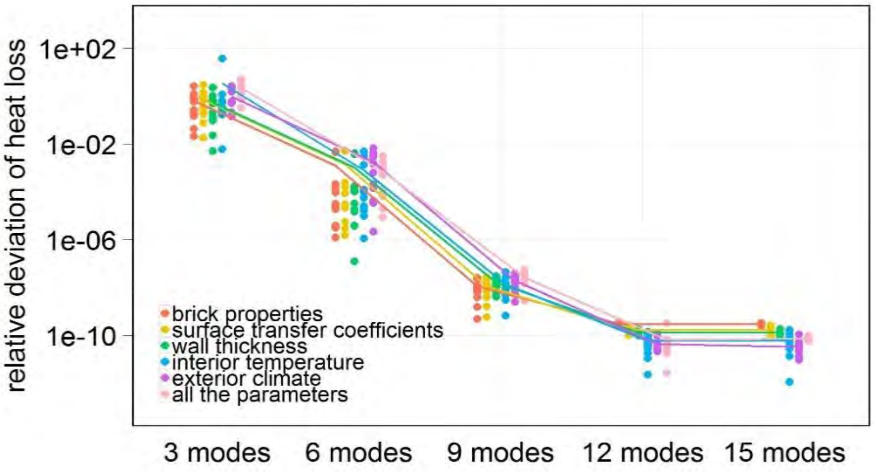


Figure 5: Relative deviations of heat losses between reference solution and the POD models constructed with different boundary conditions, material properties and component geometry.

Figure 5 shows that, the same as in Figure 4, the accuracy of using POD for simulating different problems increases as the number of their construction modes raises and after 9 modes this improvement becomes very small. As a result, it is illustrated that in all the 6 scenarios, with a sufficient number of construction modes, the POD method can provide an accurate result. In addition, comparing with the results of all the 6 scenarios, only fairly small differences among them can be observed.

3.4.1.4 Discussion

Combining the results of Figures 1 to 5, we can conclude that, with enough number of modes the PGD method can provide a relatively accurate result. In relation to the POD, only when the size of the snapshots is six hours or lower, an inaccurate result may be obtained. Further, it is shown that the use of POD to construct a reduced model for simulating different problems can provide an accurate result (Figures 4 and 5), and hence the robustness of the POD method is confirmed.

3.4.2 Probabilistic case study

In section 2.4.1 we have shown that both PGD and POD methods can provide a relatively accurate result for the deterministic simulation of the heat transfer problem. However, since there are numerous uncertainties in the properties of building materials and components, in construction geometry, in climatic loads etc., a probabilistic method is of crucial importance for obtaining not only the related values but also the distributions of the target outcomes. A variety of probabilistic modelling approaches for uncertainty propagation has been developed over the last decades (Oberkampf et al., 2002; Helton, 1994; Draper, 1995; Helton and Davis, 2003; Lee and Chen, 2009). Among them, sampling-based uncertainty propagation through Monte Carlo approaches is the most versatile and widely used, because of its general applicability and typical robustness (Janssen, 2013). A detailed introduction of (quasi-) Monte-Carlo based uncertainty analysis for the hygrothermal simulations are respectively presented by (Hou et al. 2019) and (Janssen, 2013).

3.4.2.1 General information

In this probabilistic case study, the mean and standard deviation of the distribution of cumulated heat loss over one year are simulated within the framework of Monte Carlo. In addition, a Sobol' sequence is used for minimizing the number of the needed simulations (Hou et al. 2018). In order to evaluate the performances of using POD and PGD for the probabilistic case study, the absolute errors between a reference solution and the related POD and PGD approximations are calculated. Here, the reference solution is calculated by 10 repetitions of a 2^{14}-runs scrambled Sobol' sequence (Hou et al. 2018), and each run of the simulation is simulated by the FEM with 200 mesh elements with the simulation period of one year and a fixed time step of one hour. The reference solution for the mean of the resulting heat loss distribution is 74.0071 kWh.

3.4.2.2 Input parameters

In this section, instead of using the fixed values of the parameters in relation to the boundary conditions, material properties and component geometry as described in section 3.5, these parameters are considered probabilistic. An overview of the distributions of these probabilistic parameters is given in Table 2.

Table 2: Probabilistic input parameters and distributions

Input parameter	**Input distribution**
Exterior climate	$D(Italy, Belgium, Sweden)$
Interior temperature [°C]	$U(15, 25)$
Exterior heat transfer coefficient h_e [W/m^{2}K]	$U(10, 40)$
Interior heat transfer coefficient h_i [W/m^{2}K]	$U(6, 10)$
Wall thickness $[m]$	$U(0.2, 0.5)$
Brick density $[kg/m^3]$.	$U(1700, 2100)$
Thermal conductivity $[W/mK]$	$U(0.6, 1.0)$

Explanation of symbols used:
U(a, b): uniform distribution between a and b
D(a, b): discrete uniform distribution with options a and b

In addition, in relation to the POD models, the POD basis is trained based on the central values of the above input distributions. More specifically, for materials properties, the density of the wall ρ is 1900 kg/m^3, the thermal capacity c and conductivity λ of the wall are respectively 870 J/kgK and $0.8 W/mK$. The temperature at the exterior surface $u_{exterior}$ is governed by climate data of Gaasbeek (Belgium), and the indoor air temperature $u_{interior}$ is 20°C at the interior surface. The related interior and exterior surface transfer coefficient are $h_i = 8$ W/m^{2}K and $h_e = 25$ W/m^{2}K, respectively. In relation to the component geometry, the thickness of the wall L is 0.35 m.

3.4.2.3 Results

Since the results of the mean and the standard deviation of the heat loss are very similar, only the result of the mean is shown in this report, given that this is a good illustration of overall findings.

First, in order to evaluate the performance of POD with respect to the size of the snapshots, the absolute errors between the reference solution and POD models constructed by snapshots with different time periods are shown in Figure 6. Here, all the POD models are constructed by 15 modes. Moreover, for assessing the accuracy of POD with respect to the number of construction modes, the absolute errors between the reference solution and POD models constructed by different number of modes are shown in Figure 7. Here, in order to stay in line with the case study in section 3.5.4.2, the simulation period of the snapshots is one single day. At last, for obtaining an overview of the performance of different model order reduction methods, the simulation results of POD, PGD and the FEM in relation to the mean of the heat loss distribution are shown in Figure 8. Here, the simulation period of the snapshots of the POD models is one single day and the POD and PGD models respectively, are constructed by 6 and 15 modes.

In addition, for all the models, the results of the mean of the heat loss distribution are derived by 10 replications of a $[2^3, \dots, 2^9]$-runs of scrambled Sobol sequence. Moreover, 10 independent repetitions of the above 10 replications of the $[2^3, \dots, 2^9]$ evaluations are made at each of these 7 sample sizes to obtain a robust conclusion of the accuracies of the different models.

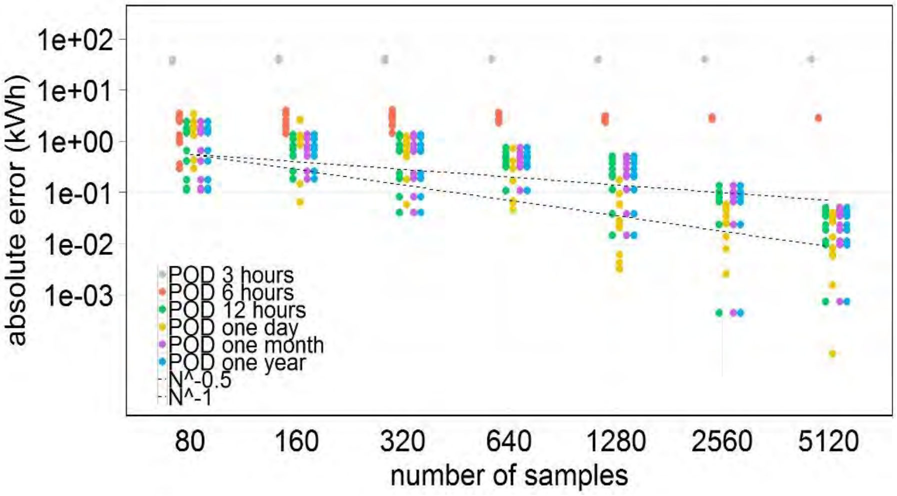


Figure 6: Absolute errors of POD models with the snapshots of different time periods.

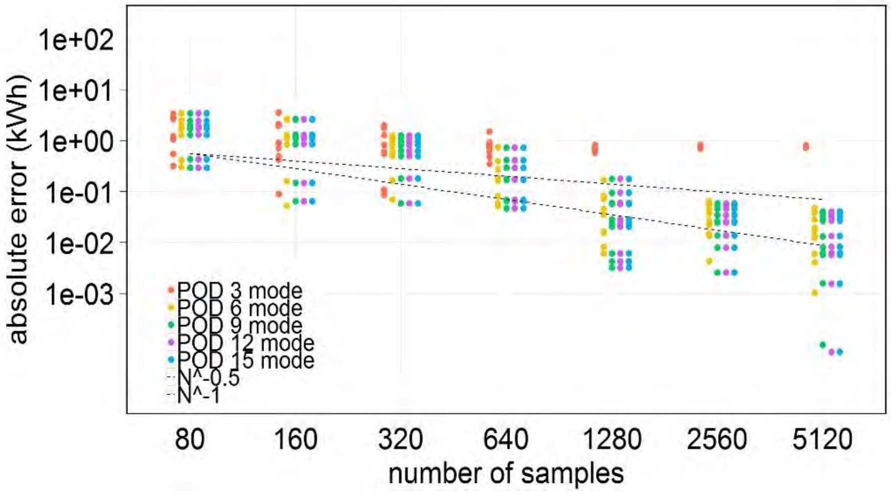


Figure 7: Absolute errors of POD models with different number of construction modes.

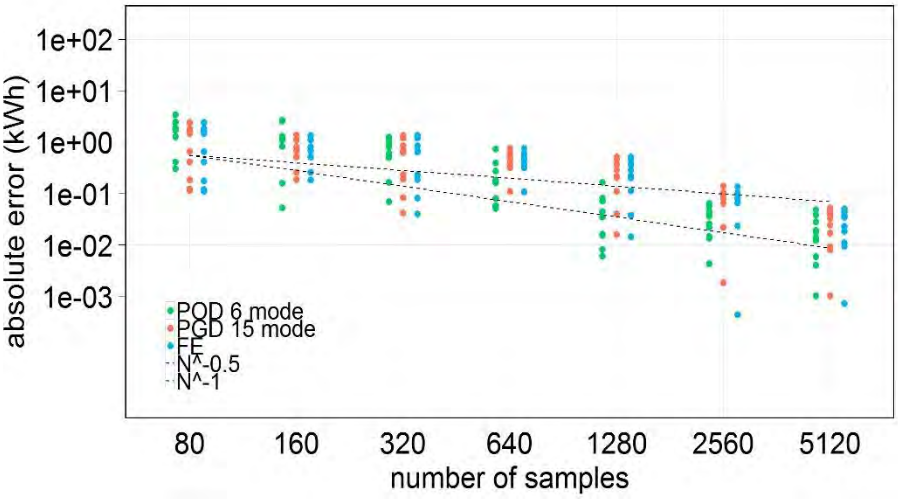


Figure 8: Absolute errors of POD models, PGD models and FE models.

3.4.2.4 Discussion

In Figure 6, except for the POD constructed by the snapshots of 3 and 6 hours, the absolute errors of all the other POD models are similar and decrease as the number of the simulations increase. The convergence rate here is approximately $\frac{1}{n}$, here n is the number of the samples. However, contradictory to Figure 4, the differences of the absolute errors between the POD models in Figure 6 are smaller than the differences of the relative deviations between the POD models in Figure 4. Especially for a small number of simulation runs, these differences tend to be even smaller.

Figure 7 shows that, except for the POD constructed with 3 modes, the absolute errors of the other POD models are very identical. Even for the POD models constructed by 6 and 15 modes, the absolute errors between these two models are very much the same. This result is different from the results shown in Figure 5. However, with 6 construction modes the relative deviations of all the POD models are around 10^{-4}, hence the errors of these 6 modes POD models are 0.001 kWh to 0.01 kWh. Comparing with the absolute errors (0.01 kWh to 1 kWh) for the probabilistic analysis in Figure 7, the errors of these simulation models tend to be rather small, and hence have a very limited effect to the overall convergence behaviour of these POD models.

It is shown in Figure 8 that, the absolute errors of the POD models constructed with 6 modes, the PGD models constructed with 15 modes and the FE models are very similar. More specially, for all of the three models, the absolute errors decrease as the number of simulations increase, with a convergence rate of approximately $\frac{1}{n}$. In addition, combined with all the findings in Figure 6 to 8, in relation to the POD methods, it is shown that the accuracies of POD with 6 and 15 construction modes are very similar. Meanwhile, as the use of POD with less construction modes can significantly save the computational expense, especially for the probabilistic analysis when a large number of repetitions need to be performed, this computational saving could be tremendous. Therefore, the use of POD with a small number of modes (6 modes) is suggested.

3.5 Nonlinear Scenarios

3.5.1 Applications for nonlinear scenarios

In section 2.4, the accuracies of POD and PGD for simulating linear thermal performance have been investigated. However, in the real world, many of the hygrothermal simulations are inherently nonlinear, and hence this section studies the potential use of POD for (strong) nonlinear hygrothermal simulations. It should be mentioned that, as it was shown in section 3.4.1.2 that the accuracy of PGD is (mostly) lower than POD, the investigations in this section focus on the POD method only.

For investigating the performances of POD for hygrothermal simulations, a reference benchmark which can assess the performance of different simulation models is required. In this report in order to demonstrate the study based on different levels of nonlinearity and complexity, three applications – the HAMSTAD benchmarks 2, 3 and 4 (Hagentoft, 2002) – are selected as the illustrative case studies. All of them have a one-dimensional geometrical structure, but are based on different transfer mechanisms with different combinations of boundary conditions and material properties.

More specifically, HAMSTAD benchmark 2 analyses the isothermal moisture redistribution in one material layer caused by sudden drops of the relative humidity of the surroundings. HAMSTAD benchmark 3 centers on air transfer through a light weight wall, wherein the moisture and heat transfer are mainly driven by the air flow. HAMSTAD benchmark 4 analyses a wall with a hygroscopic finishing material at the inside, under the boundary conditions with relative humidity, heat and moisture loads at both inner and outer surfaces. It should be mentioned that, HAMSTAD benchmark 4 represents a very strongly nonlinear and complicated case study since its climatic load is severe and its material properties are rather harsh (Hagentoft et al. 2004). Its climatic load contains various heat and moisture transfer phenomena such as moisture condensation induced by cooling, rain load, alternating drying and wetting, moisture redistribution across the contact surface between two capillary active materials, etc. On the other hand, the first layer of its material having an extremely fast liquid transfer.

The complexity of HAMSTAD benchmark 4 makes it rather difficult for numerical simulation solvers to provide an accurate and stable solution. For instance, Hamopy (Rouchier, 2015) – the simulation tool used for HAMSTAD benchmark 3 and 4 – is not capable to simulate the whole process of HAMSTAD benchmark 4. Therefore, for benchmark 4, in order to investigate the performance of POD for the whole simulation period without losing the strong nonlinearity and complexity of this case study, the climatic load is kept in line with the reference climatic conditions, but with a reduction to 10 percent of the rain load. Detailed information about HAMSTAD benchmarks 2, 3 and 4 and their related construction geometries, material properties and boundary conditions can be found in (Hagentoft, 2002).

In order to evaluate the performance of POD as a function of the number of construction modes, for all the three case studies the results derived by the standard FEM and POD models constructed by different number of modes are compared. The simulation results of the FEM are taken here as the reference solution. More specifically, for HAMSTAD benchmark 2, 3 and 4 their reference solutions are respectively calculated by the FEM with 200, 121 and 1501 spatial mesh nodes. In addition, as mentioned in section 2, the POD is constructed for simulating new problems, hence for HAMSTAD benchmark 3 and 4, the performances of POD as a function of the size of their snapshots are evaluated. More specially, for HAMSTAD benchmark 3 several POD models are constructed by using snapshots from different time intervals (50 days, 30 days, 20 days, 10 days, 5 days and 1 day), and these reduced models are used to predict the results of a new problem with a time period of 100 days. On the other hand, for HAMSTAD benchmark 4, different POD models are constructed by using snapshots of respectively 12 hours and 50 hours, and these reduced models are used to predict the results of a new problem with a time period of 120 hours.

3.5.2 HAMSTAD benchmark 2

For getting a direct view of the performance of POD in relation to HAMSTAD benchmark 2, the moisture content profiles at 100 hours of the reduced models constructed by different number of modes together with the results derived by the reference solution are shown in Figure 9.

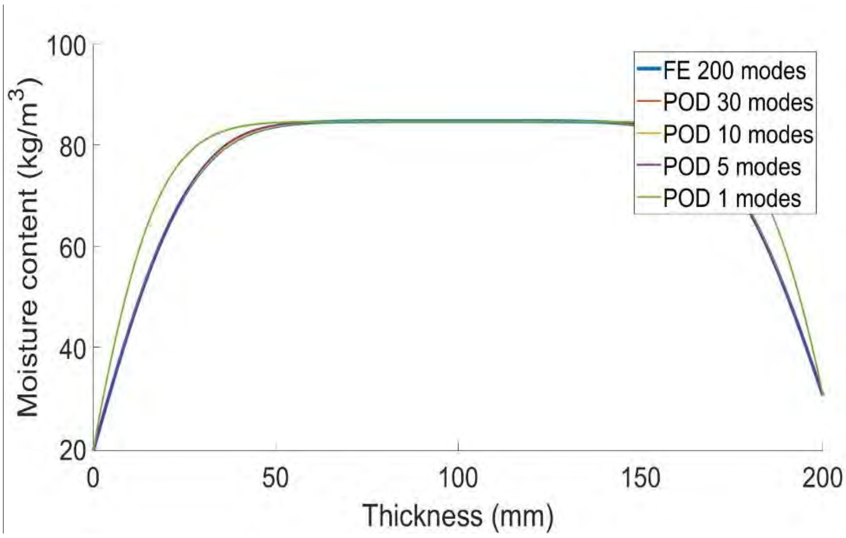


Figure 9: Moisture content profiles at 100 hours of the FE solution and solutions of the reduced models constructed by different number of modes.

Figure 9 shows that the differences between the reference solution and the results of the other POD models are very small, except for the POD model constructed by 1 mode. With more than 10 construction modes, there is no observable differences between the reference solution and the POD approximations. Hence, for HAMSTAD benchmark 2, with a relatively small number of construction modes, the POD method can provide an accurate result of the moisture distribution in the material layer.

3.5.3 HAMSTAD benchmark 3

In the case study of HAMSTAD benchmark 3, three research questions are studied: 1) the accuracy of POD as a function of the amount of construction modes when simulating the same problem, 2) the accuracy of POD as a function of the size of snapshot and the amount of construction modes when simulating new problems with longer simulation periods, and 3) compare the accuracies of POD and FEM as a function of construction modes or nodes.

3.5.3.1 The performance of POD when simulating the same problem

In relation to HAMSTAD benchmark 3, both temperature and moisture distributions at 0.19 m (measured from the outside, the total thickness is 0.2m) in the building component simulated by different reduced models together with the result of the reference solution are shown in Figure 10.

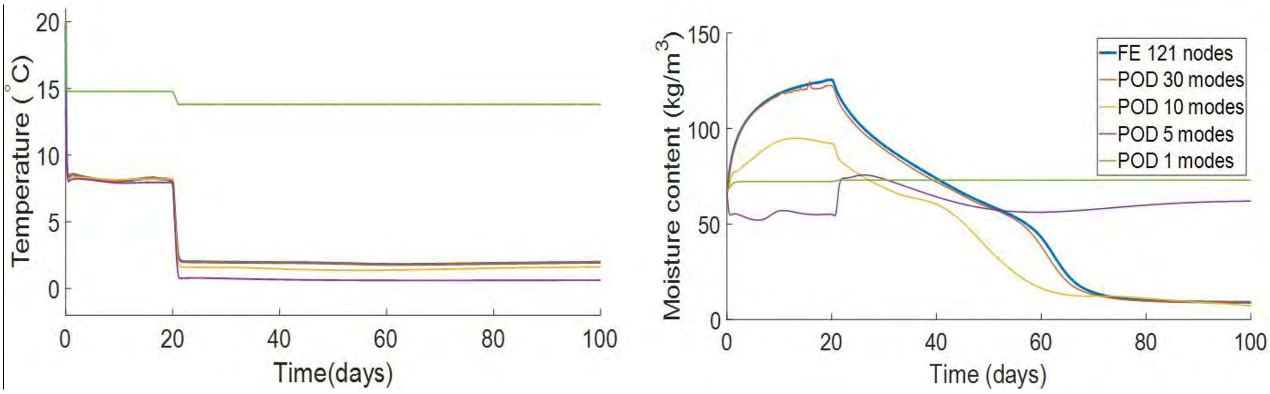


Figure 10: Temperature and moisture content evolutions at 0.19 m of the building component simulated by the FEM and the reduced models constructed by different number of modes.

Figure 10 illustrates the temperature and moisture evolutions at 0.19 m of the building component simulated by different models. It is demonstrated that, the accuracy of POD method increases as the number of its construction modes raises. With respect to the temperature evolutions, with 10 construction modes, the distribution of the temperature can be quantified rather accurately. On the other hand, it is shown that for obtaining a relatively accurate evolutions of the moisture content, at least 30 modes are needed. Therefore, comparing with quantifying the temperature evolutions, due to the stronger nonlinear behaviour of the moisture transfer mechanism, obtaining the moisture content profiles requires more construction modes.

3.5.3.2 POD for simulating new problems with longer periods

In section 2.5.2 and 2.5.3.1, it is shown that the POD model constructed based on the full set of snapshots can provide an accurate result when simulating the same system. However, in practice this recalculation of the same problem does not add any value, as all the information is already included in the matrix of snapshots. Instead, in this section we are investigating the use of POD to construct a reduced model for simulating different problems in relation to longer simulation period. In other words, the accuracies of the POD models constructed by different size of snapshots are evaluated. More specifically, for getting a more direct view of the performance of POD method as a function of the size of its snapshot, similar as before both temperature and moisture evolutions at 0.19 m of the building component simulated by FEM and the reduced models constructed by 25 modes and different size of snapshots are shown in Figure 11. In addition, in order to evaluate the accuracy of POD method as a function of the amount of the construction modes, the average temperature and moisture content differences between the reference solution and POD models built by different size of snapshots, as a function of the amount of their construction modes are shown in Figure 12.

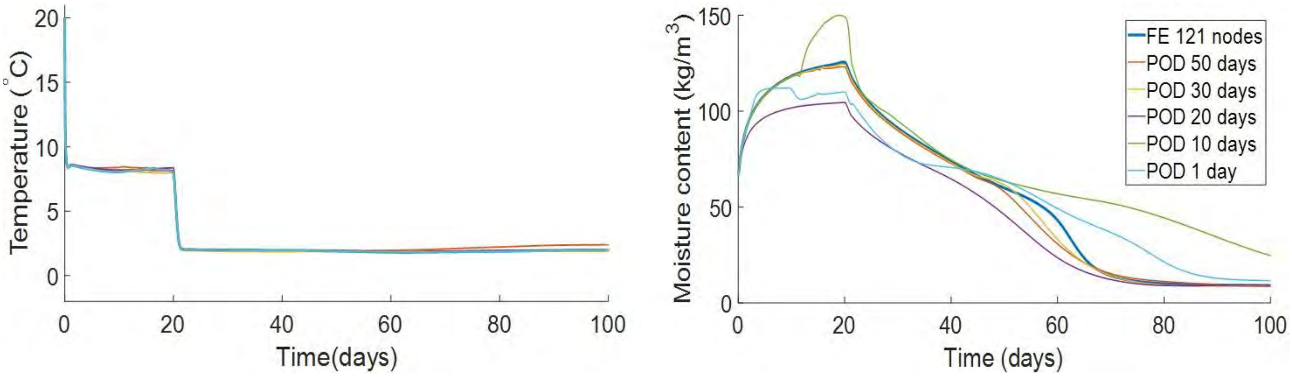


Figure 11: Temperature and moisture content evolutions at 0.19 m of the building component simulated by the FEM and reduced models constructed by 25 modes and different size of snapshots.

Figure 11 is similar to Figure 10, but now for different size of snapshots. It is demonstrated that, with only one day of snapshot and 25 modes, the temperature evolutions can be obtained rather accurately. On the other hand, relative to the temperature evolutions, due to the stronger nonlinear behaviour of the moisture transfer mechanism, obtaining the moisture content profiles requires a much larger size of snapshot.

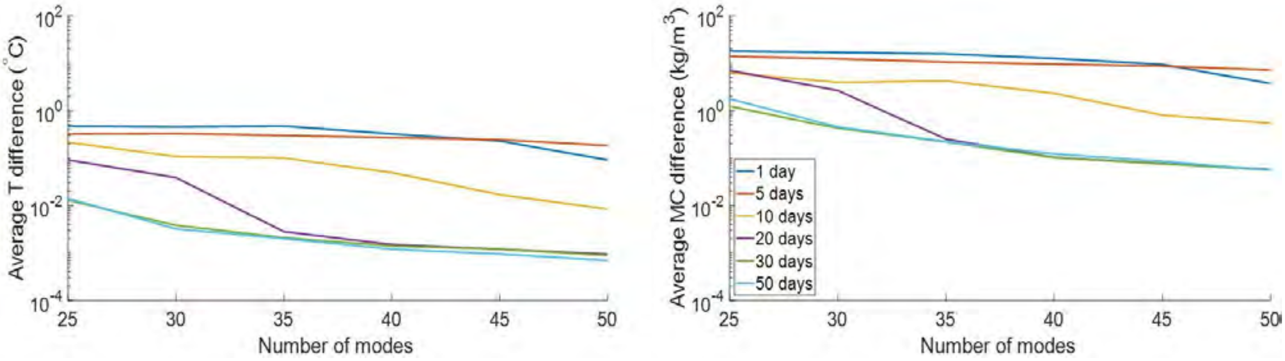


Figure 12: Absolute average temperature (left) and moisture content (right) differences between the FE solution and solutions of the reduced models constructed by different size of snapshots and different number of modes.

Figure 12 confirms the results of Figure 10 and Figure 11, the accuracy of POD method increases as the amount of their construction modes raises and as the size of their snapshot grows. Comparing with the average temperature differences between the reference solution and the results of different POD models, the average moisture content differences are a lot larger (larger than one order of magnitude). Hence, with the same size of snapshot, quantifying the distribution of the moisture content requires more construction modes than capturing the distribution of the temperature. In addition, Figure 12 demonstrates that, with less than 30 modes, relatively large accuracy differences are found between the results of the reduced models constructed by the snapshots of 20 days and 30 days. One possible reason is that for HAMSTAD benchmark 3, since there is a large change of the air pressure gradient from the 20 days to the 21 days (Hagentoft, C. E., 2002), the snapshots that include this period contain the crucial information of the boundary condition and hence has the potential to be the base of a more accurate reduced model.

3.5.3.3 Compare POD with FEM

In section 2.4, 2.5.2, 2.5.3.1 and 2.5.3.2, it was shown that the POD method can deliver accurate results when performing linear and nonlinear hygrothermal simulation and is capable to simulate new problems with longer simulation periods. However, to further illustrate the superiority of POD over the conventional method for instance FEM models, the accuracies of POD models and FEM constructed by the same number of modes and nodes are compared. The reference solution is calculated by FEM with 301 nodes. For the FEM, in order to obtain the same temperature and moisture content profiles as the reference solution, the intermediate values are calculated by interpolation. The results are shown in Figure 13.

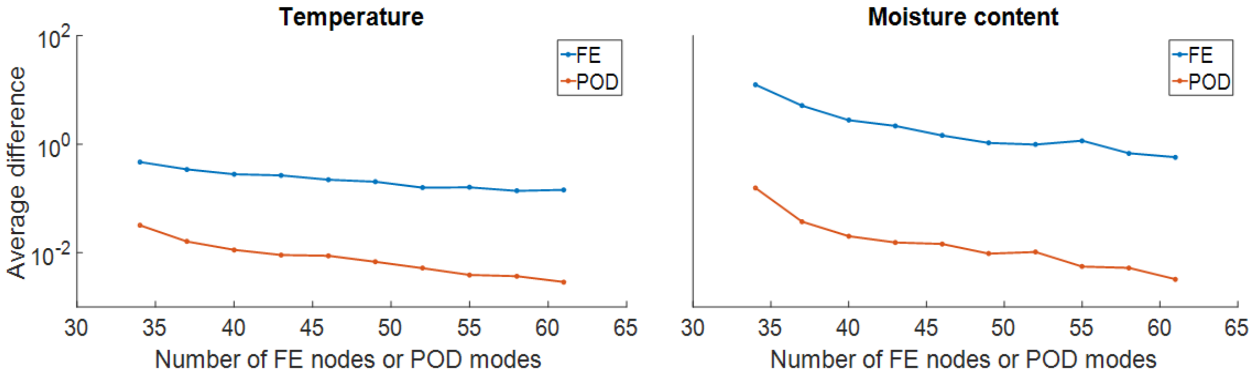


Figure 13: Average temperature and moisture content differences between the reference solution and solutions of the POD and FE models constructed by different number of modes and nodes.

Figure 13 illustrates the average temperature and moisture differences of different models. It is demonstrated that, with the same number of construction modes and nodes, the accuracy of POD methods are (much) higher than FEM and the differences are almost larger than two orders of magnitudes. Therefore, for obtaining the same target accuracy, the POD method requires much less modes than the FEM. However, since POD method requires additional matrix multiplication operations than FEM when constructing the related stiffness and mass matrices for solving the system of equations, comparing the number of modes and nodes of POD and FEM can only be regarded as an indicator of their efficiencies.

3.5.4 HAMSTAD benchmark 4

As mentioned in section 2.5.1.1, the fourth benchmark is a very strongly nonlinear and complicated case study, and hence its computational cost is rather high. In this section, instead of providing an exhausted investigation of the performances of POD as a function of the size of its snapshot and the number of its construction modes, only a limited selection of the scenarios are studied. More specifically, to illustrate the performance of POD as a function of the size of snapshots, two scenarios (POD models with snapshots of respectively 12 and 50 hours) are analysed. Moreover, in relation to the number of construction modes, in each of the two scenarios the performances of POD models with different number of modes are compared.

The results are presented in the below two subsections. To illustrate moisture content and temperature distributions at the external and internal surfaces during the whole simulation period, the values of temperature and moisture content at the outer and inner surfaces of the structure are presented in Figures 14 to 17. On the other hand, in order to show moisture content and temperature profiles across the wall at time steps during or after some rapid changes in climatic load, temperature and moisture profiles inside the structure are shown in Figures 18 to 27.

3.5.4.1 Moisture and temperature distributions at the outer and inner surface

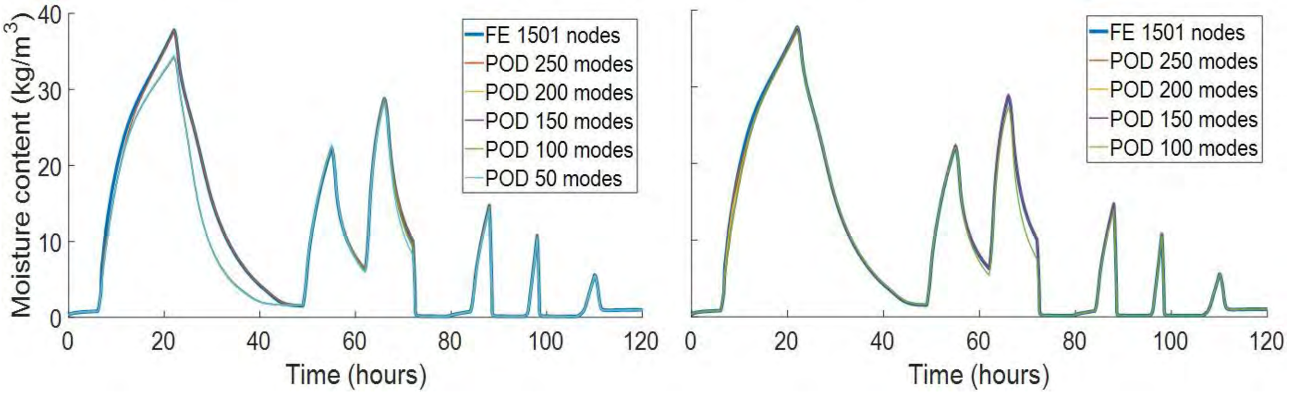


Figure 14: Moisture content evolutions on outer surface simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

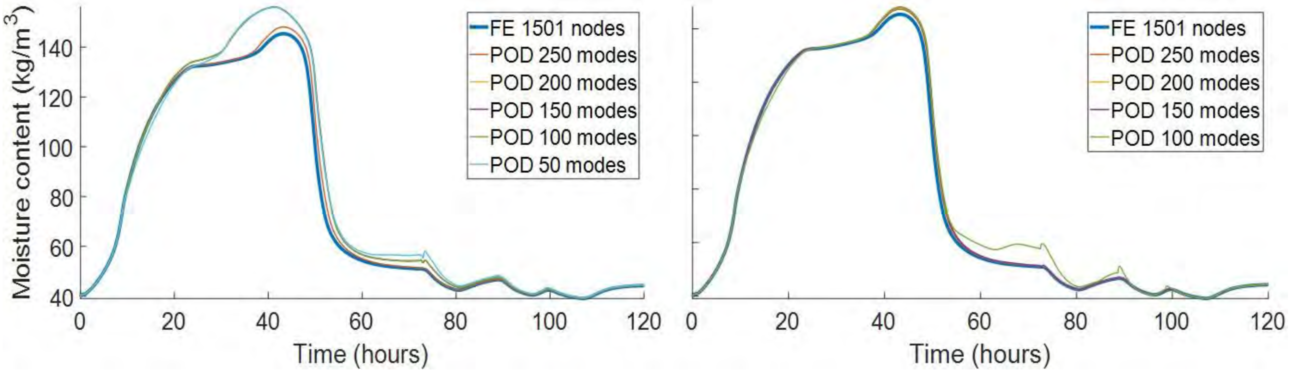


Figure 15: Moisture content evolutions on inner surface simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

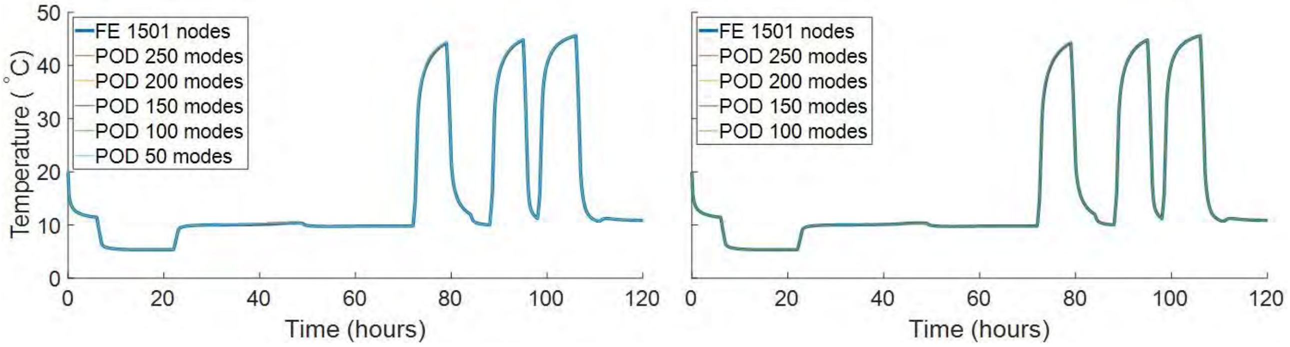


Figure 16: Temperature evolutions on outer surface simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

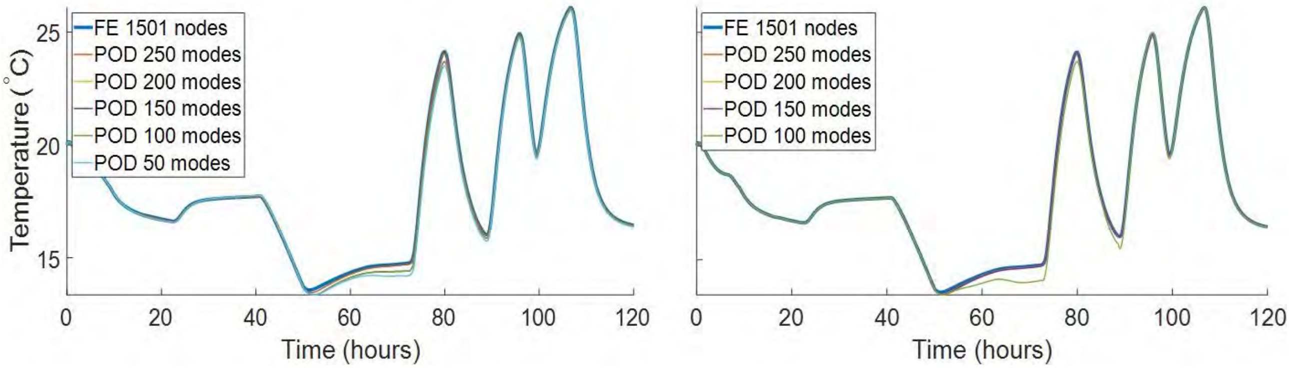


Figure 2-17: Temperature evolutions on inner surface simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

Figures 14 to 17 represent the moisture content and temperature evolutions at outer and inner surfaces of the structure simulated by different models. For POD models based on the snapshot of 12 hours (left figures), the differences between the reference solution and the POD model is relatively large, except for the 250 modes POD models. On the other hand, for the POD models constructed by the snapshot of 50 hours (right figures), differences between the models are visually very small, except for the POD model constructed by 100 modes. Therefore, with the same number of construction modes, POD models constructed by the snapshot of 50 hours are more accurate than those POD models with 12 hours' snapshot for quantifying the moisture content and temperature distributions. And hence, the performances of POD models are improved by increasing the size of snapshot.

Figures 14 to 17 also show that the accuracy of POD increases with the number of construction modes. However, for POD models with 12 hours' snapshot, rather large errors can be found in the POD approximations with less than 250 modes. In addition, for POD models with 12 hours' snapshot and built by 50 to 200 modes, the improvements of their accuracies are relatively small. On the other hand, a fairly good approximation can be obtained by POD models with 50 hours' snapshot constructed by only 150 modes. Further, the differences between the reference solution and the results of POD models for quantifying the temperature distributions are visually (much) smaller than for quantifying the moisture content distributions. Therefore, using POD models to predict the temperature distributions requires a smaller size of snapshot and/or less construction modes than for quantifying the moisture content distributions.

3.5.4.2 Moisture and temperature profiles inside the structure

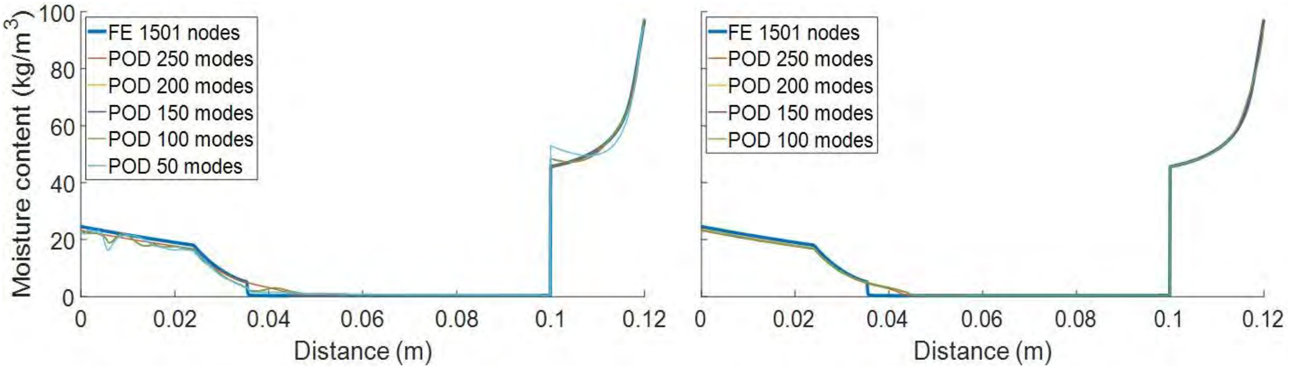


Figure 18: Moisture profiles at 12 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

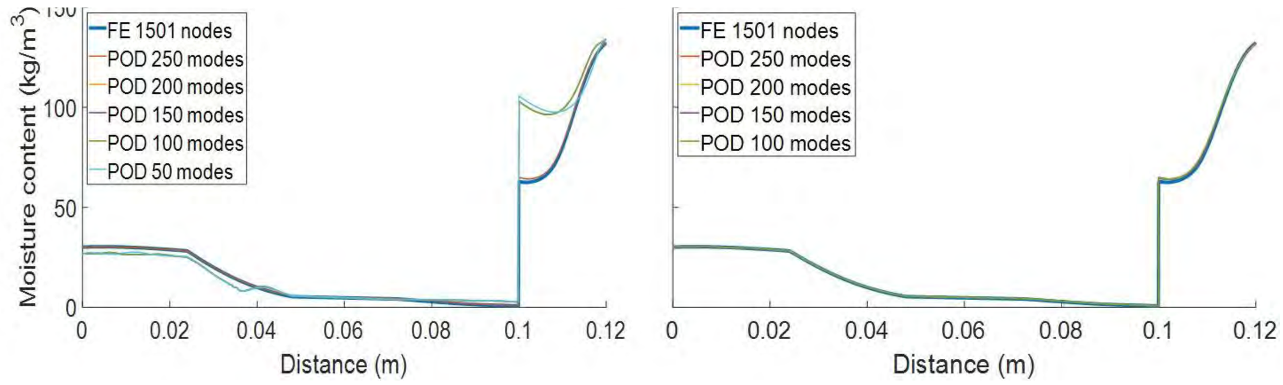


Figure 19: Moisture profiles at 24 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

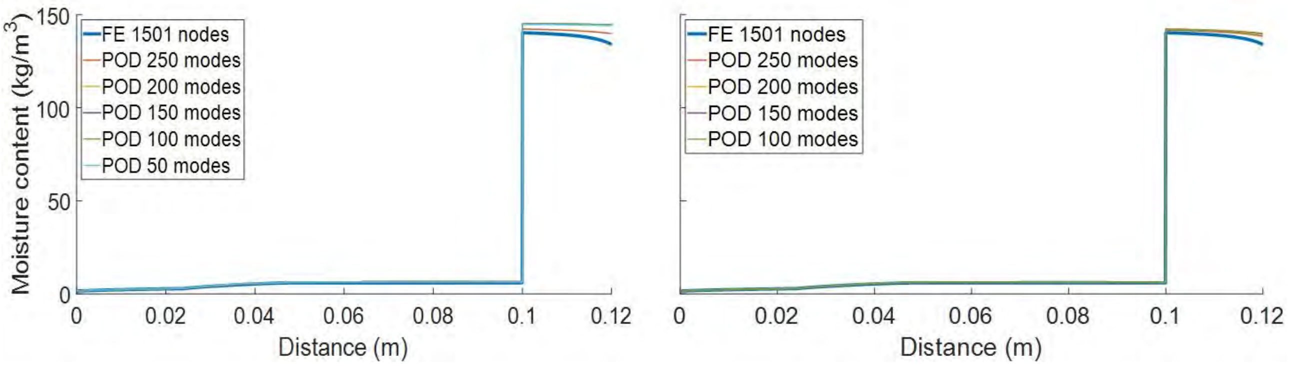


Figure 20: Moisture profiles at 48 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

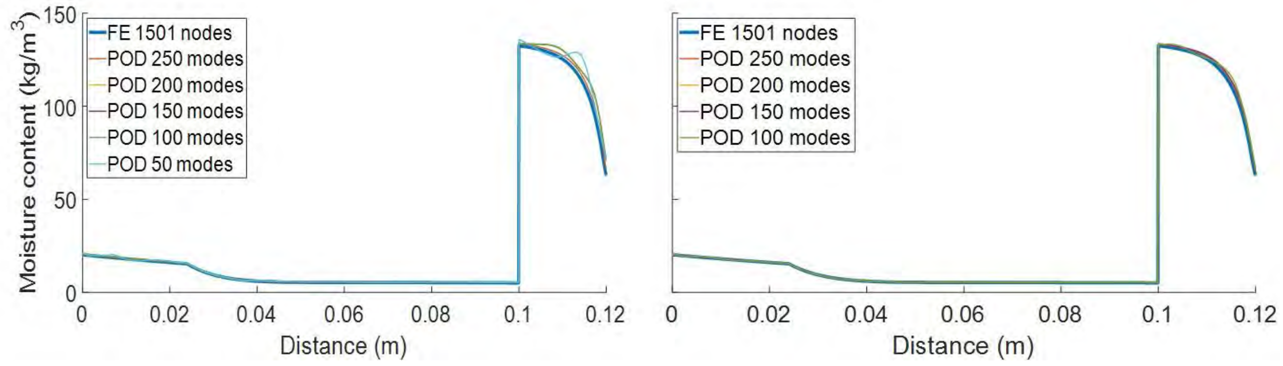


Figure 21: Moisture profiles at 54 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

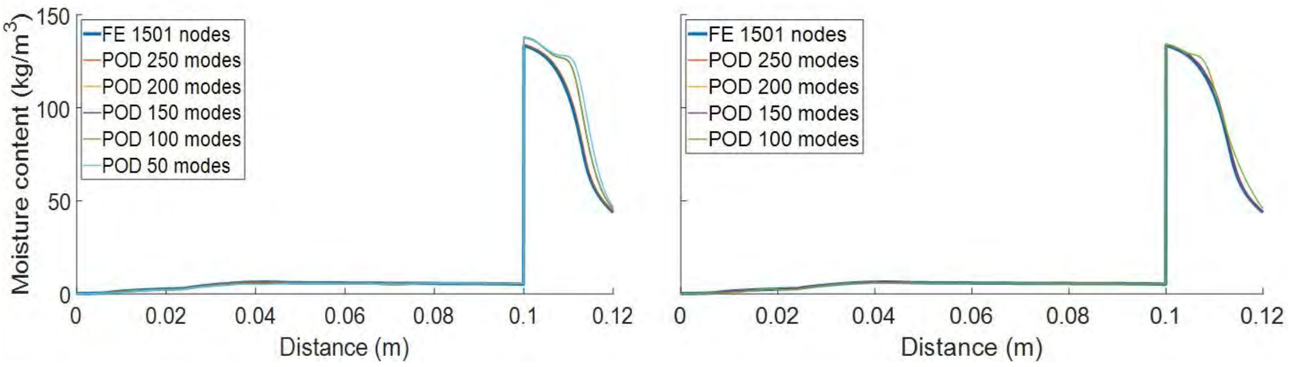


Figure 22: Moisture profiles at 78 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

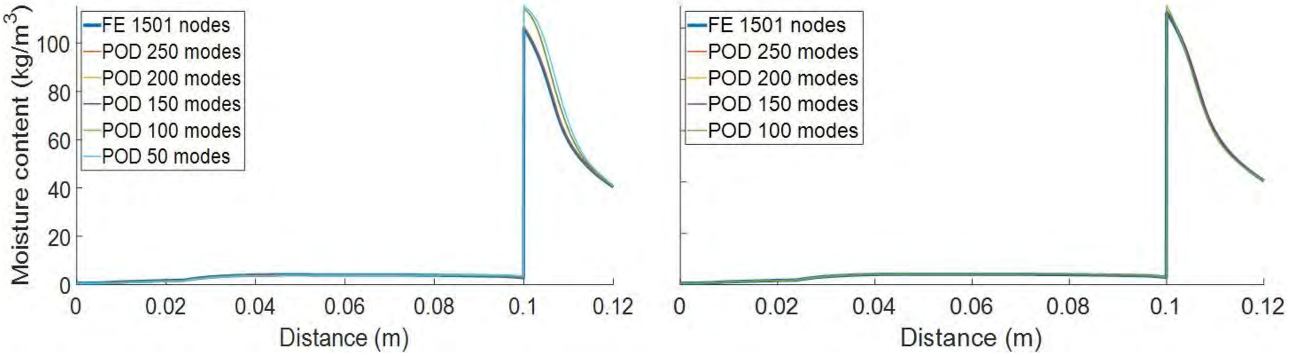


Figure 23: Moisture profiles at 96 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

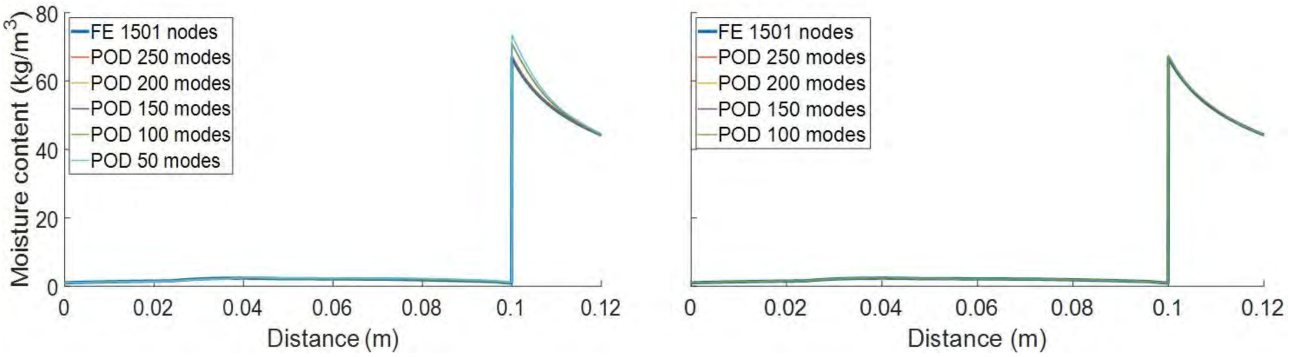


Figure 24: Moisture profiles at 120 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

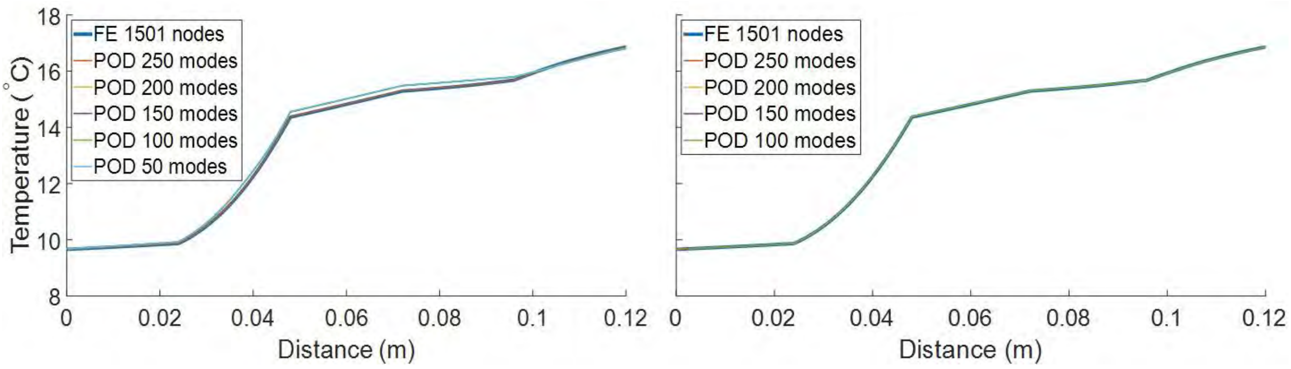


Figure 25: Temperature profiles at 24 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

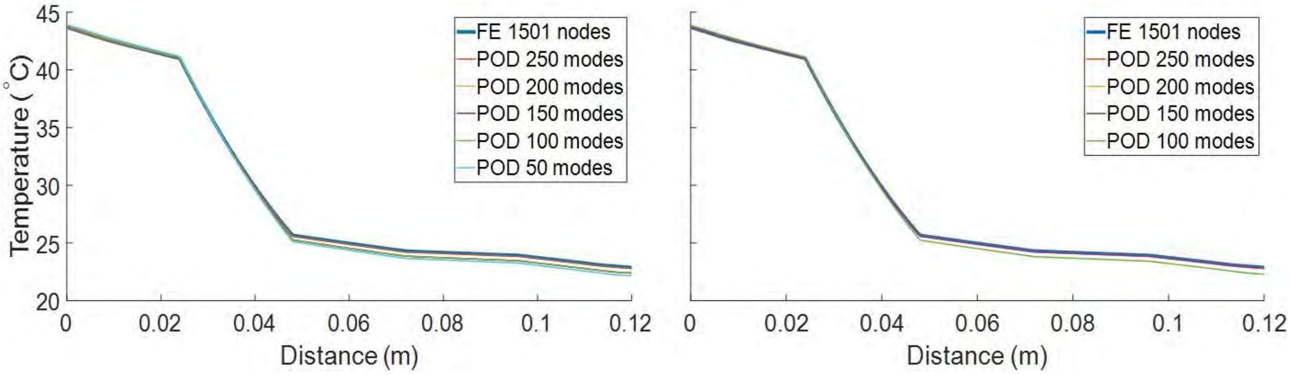


Figure 26: Temperature profiles at 78 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

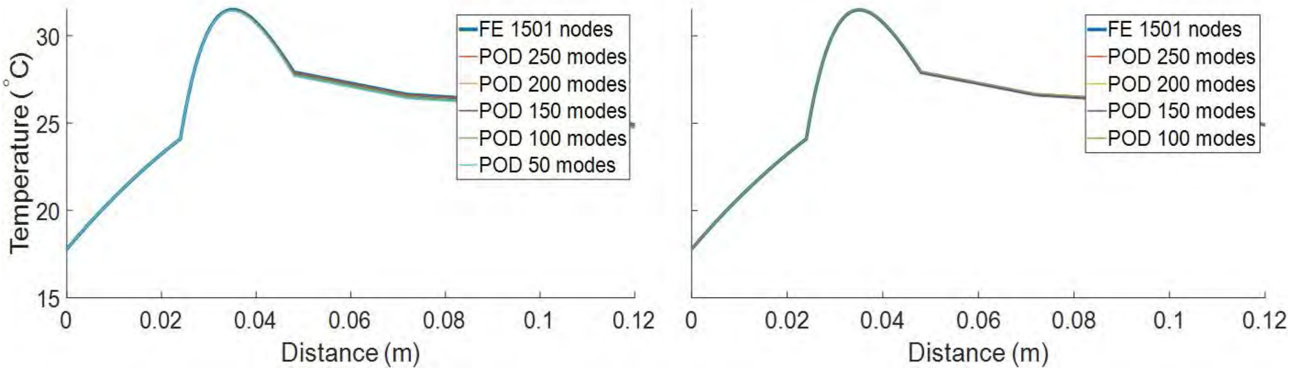


Figure 27: Temperature profiles at 96 hours, simulated by FE and POD constructed by 12 (left) and 50 (right) hours snapshot, with different number of construction modes.

Figures 18 to 27 confirm the results of Figures 14 to 17. Comparing the left and right parts of Figures 18 to 27, it is demonstrated that with the same number of construction modes, the accuracy of POD models constructed by a snapshot of 50 hours is relatively higher than those POD models with 12 hours' snapshot for quantifying the moisture and temperature profiles across the structure. On the other hand, from each sub-figures of Figures 18 to 27, it is illustrated that the accuracies of POD increase as the number of the construction modes raise. For the POD models with 12 hours' snapshot, a roughly 250 modes are required in order to get a relatively accurate result. However, a rather good approximation can be obtained by the POD models with 50 hours' snapshot with only 100-150 construction modes.

3.5.5 Discrete empirical interpolation method

POD reduces the computational complexity and computational time for simulating a large complexed system by approximating the high dimensional process by its "most relevant information". However, for nonlinear problems, similar as the conventional simulation methods (for instance the finite element method and finite volume method), the inefficiency arises from the high computational cost in repeatedly evaluating the nonlinear terms still in the scope of the dimension of the original problem (Chaturantabut et al. 2010).

To overcome this bottleneck, Chaturantabut et al (2010) proposed the discrete empirical interpolation method (DEIM). The main idea is to approximate the nonlinear term by combining projection with interpolation. DEIM projects the nonlinear terms onto a reduced basis and evaluated at only several specific locations which are selected by a greedy algorithm. The component values at other locations can be interpolated through the reduced basis by an interpolation matrix that can be pre-computed offline (Tiso et al. 2013). Wherein, the reduced basis related to each nonlinear term are obtained by POD method.

For investigating the potential use of DEIM for nonlinear hygrothermal simulations, HAMSTAD benchmarks 2 is selected as the illustrative case study. In order to evaluate the performance of DEIM as a function of the number of construction modes and interpolation points, the results derived by Finite Volume method (FV) with 200 volumes and the reduced models constructed by different number of modes (25 modes, 20 modes, 15 modes, 10 modes, 5 modes) and interpolation points (65 points, 55 points, 45 points, 35 points, 25 points) are compared. The simulation results of the FV are taken here as the reference solution.

In this report for getting a direct view of the performance of DEIM, the moisture content profiles at 100 hours of the reduced models constructed by different number of modes and interpolation points together with the result of the reference solution are shown in Figure 28. In addition, to obtain a comprehensive view of the performance of DEIM as a function of the number of construction modes and interpolation points, the accuracies of different models with different number of modes and/or interpolation points are shown in Figure 29.

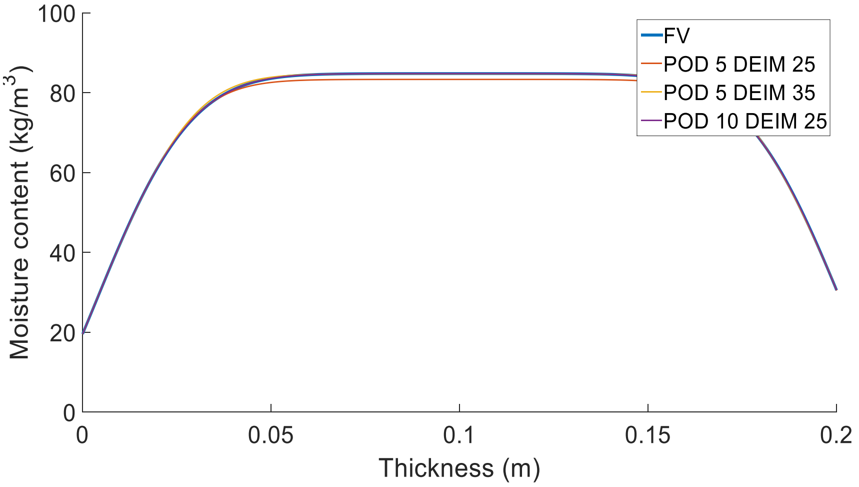


Figure 28: Moisture content profiles at 100 hours of the reference solution and solutions of the reduced models constructed by different number of modes and DEIM points.

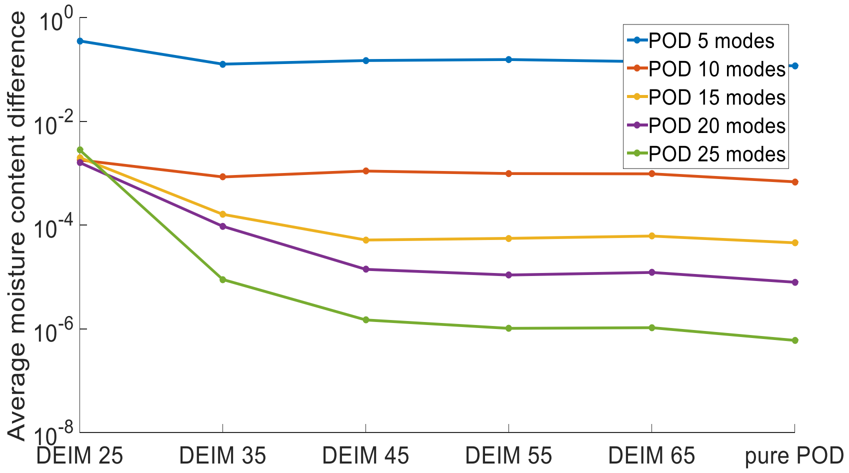


Figure 29: Average moisture content difference between the reference solution and solutions of the reduced models constructed by different number of modes and DEIM points.

Figure 28 shows that for the DEIM model with 5 POD modes and 35 interpolation points, and the DEIM model with 10 POD modes and 25 interpolation points there is no observable differences between the reference solution and the DEIM approximations. Hence, for HAMSTAD benchmark 2, with a relatively small number of construction modes and interpolation points, DEIM can provide an accurate result of the moisture distribution in the material layer.

Figure 29 presents an overview of the accuracies of different models constructed by different number of POD modes and DEIM points. It confirms the results of Figure 28, the accuracy of the DEIM method increases as the amount of their construction modes and interpolation points grows. On the other hand, Figure 29 also shows that with more than 45 DEIM points the accuracies of the reduced models do not increase significantly.

3.5.6 Discussion

For HAMSTAD benchmarks 2 and 3, it is shown that the use of POD for simulating nonlinear hygrothermal performance has the potential to provide an accurate result and is able to predict the results of a new problem with longer simulation period. Moreover, for obtaining the same target accuracy, the POD method requires much less modes than the FEM. In addition, it is shown that the DEIM has the potential to further reduce the computational cost for simulating nonlinear hygrothermal performance.

However, caution should be taken when choosing the number of simulation snapshots and construction modes to build a reduced model. In relation to the snapshots, it is strongly recommended to include the crucial information and the periodic patterns of the boundary conditions as much as possible. On the other hand, with respect to the modes, choosing an appropriate number of modes depends on the extent of the nonlinearity of the target problem. Normally the stronger the nonlinearity, the more modes is needed to obtain an accurate result. Therefore, in practice an appropriate size of snapshot could be selected at first and considered as the reference solution. Next, the reduced model was constructed by keeping adding modes and comparing the result with the reference solution until no further improvement can be obtained. Finally, the constructed reduced model was used for simulating new problems with a longer simulation time period.

For HAMSTAD benchmark 4 similar conclusions are drawn, the accuracy of POD increases as the number of construction modes grows and the size of its snapshot raises, and the size of its snapshot tends to be the most effective factor. However, due to the very strong nonlinear behaviour of this case study, an accurate and stable solution is rather difficult to be derived by many numerical solvers. For the POD method, some instabilities are found during the simulation period especially for those POD models built by a relatively small size of snapshot and with a rather small amount of construction modes. Hence, the efficiency of POD may suffer from the strong nonlinear behaviour of the target simulation object, and a robust solver which can provide stable solutions for very strong nonlinear problems is highly demanded.

3.6 Conclusion

In Section 2 of this report, the performance of two model order reduction methods (POD and PGD) was investigated, based on both linear and nonlinear scenarios. In relation to the linear scenario, it is shown that the POD method is able to provide an accurate result with a fairly small size of snapshot (more than 6 hours) and relatively small number of construction modes (more than 6). It is also illustrated that, the use of POD to construct a reduced model for simulating different problems can provide an accurate result, and hence the robustness of the POD method is confirmed. In relation to the linear probabilistic case study, the accuracies of POD with 6 and 15 construction modes are very similar, and hence the use of POD with a smaller number of modes (6 modes) is suggested in order to reduce the overall computational expense as much as possible, but that issue still need to be quantified.

On the other hand, for the nonlinear scenario the accuracy of POD is strongly depended on the extent of the nonlinearity of the target problem. Comparing with the linear scenario, much more construction modes and much larger snapshot are required in order to obtain an accurate result. In relation to the snapshots, we strongly suggest to include the crucial information and the periodic patterns of the boundary conditions as much as possible. Moreover, since the inefficiency of POD arises from the high computational cost in repeatedly evaluating the nonlinear terms, to further reduce the computational cost of the hygrothermal simulation DEIM is studied. It shows that a significant reduction of the computational complexity can be achieved by implementing DEIM.

In the RIBuild project, since a huge amount of time-consuming simulations need to be performed in order to provide an effective and comprehensive guidelines for internal insulation in historic buildings, the issue of the computational expense tends to be crucial. In this report, the potential of using model order reduction methods to reduce the computational expensive for different hygrothermal (thermal) analysis have been studied. Based on the results it is concluded that for linear and moderately nonlinear simulations, POD can significantly reduce the computational complexity of the original time-consuming simulations. On the other hand, for strong nonlinear simulations, some instability issues may occur and hence a more robust solver which can provide accurate and stable solutions is highly demanded.

3.7 References

Agudelo M. 2009. The application of proper orthogonal decomposition to the control of tubular reactors. PhD thesis, Katholieke Universiteit Leuven, Belgium.

Ammar A., Mokdad B., Chinesta F., Keunings R. 2006. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. Journal of non-Newtonian fluid Mechanics 139:153-176.

Berger J., Mazuroski W., Oliveira R.C., Mendes N. 2018. Intelligent co-simulation: neural network vs. proper orthogonal decomposition applied to a 2D diffusive problem. Journal of Building Performance Simulation 11:1-20.

Berger J., Mendes N., Guernouti S., Woloszyn M., Chinesta F. 2017. Review of reduced order models for heat and moisture transfer in building physics with emphasis in PGD approaches. Archives of Computational Methods in Engineering 24:655-667.

Chatterjee A. 2000. An introduction to the proper orthogonal decomposition. Current science 78:808-817.

Chaturantabut S., Sorensen D.C. 2010. Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing 32(5), 2737-2764.

Chinesta F., Ammar A., Cueto E. 2010. Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. Archives of Computational methods in Engineering 17:327-350.

Chinesta F., Keunings R., Leygue A. 2013. The proper generalized decomposition for advanced numerical simulations: a primer. Springer Science & Business Media.

Draper D. 1995. Assessment and propagation of model uncertainty. Journal of the Royal Statistical Society. Series B (Methodological) 57:45–97.

EN B., 2007. 15026: 2007: Hygrothermal performance of building components and building elements-Assessment of moisture transfer by numerical simulation.

Fahl M., Sachs E.W. 2003. Reduced order modelling approaches to PDE-constrained optimization based on proper orthogonal decomposition. In: Biegler L T., Heinkenschloss M., Ghattas O., van Bloemen Waanders B. (eds) Large-Scale PDE-Constrained Optimization. Lecture Notes in Computational Science and Engineering 30. Springer, Berlin, Heidelberg

Fic A., Białecki R.A., Kassab A.J. 2005. Solving transient nonlinear heat conduction problems by proper orthogonal decomposition and the finite-element method. Numerical Heat Transfer, Part B: Fundamentals 48:103-124.

Hagentoft C.E. 2002. Hamstad wp2-benchmark package. Technical report, Dept. of Building Physics-Chalmers University of Technology.

Hagentoft C.E., Kalagasidis A.S., Adl-Zarrabi B., Roels S., Carmeliet J., Hens H., ... & Adan O. 2004. Assessment method of numerical prediction models for combined heat, air and moisture transfer in building components: benchmarks for one-dimensional cases. Journal of thermal envelope and building science 27(4), 327-352.

Helton J.C. 1994. Treatment of uncertainty in performance assessments for complex systems. Risk analysis 14:483–511.

Helton J.C., Davis F.J. 2003. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. Reliability Engineering & System Safety 81:23–69.

Hou T., Nuyens D., Roels S., Janssen H. 2019. Quasi-Monte Carlo based uncertainty analysis: Sampling efficiency and error estimation. Reliability Engineering & System Safety (accepted).

Janssen H. 2013. Monte Carlo based uncertainty analysis: Sampling efficiency and sampling convergence. Reliability Engineering & System Safety 109:123–132.

Kosambi D.D. 2016. Statistics in function space. In: Ramaswamy R. (eds) D.D. Kosambi. Springer, New Delhi, 115-123.

Ladeveze P. 1985. On a family of algorithms for structural mechanics. Comptes rendus de l'Académie des Sciences 300(2), 41-44.

Lee S.H., Chen W. 2009. A comparative study of uncertainty propagation methods for black-box-type problems. Structural and Multidisciplinary Optimization 37:239–253.

Liang Y.C., Lee H.P., Lim S.P., Lin W.Z., Lee K.H., Wu C.G. 2002. Proper orthogonal decomposition and its applications—Part I: Theory. Journal of Sound and vibration 252:527-544.

Oberkampf W.L., DeLand S.M., Rutherford B.M., Diegert K.V., Alvin K.F. 2002. Error and uncertainty in modeling and simulation. Reliability Engineering & System Safety 75:333-357.

Pruliere E., Chinesta F., Ammar A., Leygue A., Poitou A. 2013. On the solution of the heat equation in very thin tapes. International Journal of Thermal Sciences 65:148-157.

Ribuild.eu. 2018. Available at: http://ribuild.eu/about.

Rouchier S. 2015. hamopy: Heat, Air and Moisture transfer in Python.

Tallet A., Allery C., Allard F. 2015. POD approach to determine in real-time the temperature distribution in a cavity. Building and Environment 93:34-49.

Tallet A., Liberge E., Inard C. 2017, February. Fast POD method to evaluate infiltration heat recovery in building walls. Building Simulation 10(1), 111-121.

Tiso P., Rixen D.J. 2013. Discrete empirical interpolation method for finite element structural dynamics. In Topics in Nonlinear Dynamics, Volume 1, 203-212. Springer, New York, NY.

Van Gelder L., Das P., Janssen H., Roels S. 2014. Comparative study of metamodelling techniques in building energy simulation: Guidelines for practitioners. Simulation Modelling Practice and Theory 49:245-257.

Vereecken E., Van Gelder L., Janssen H., Roels S. 2015. Interior insulation for wall retrofitting–A probabilistic analysis of energy savings and hygrothermal risks. Energy and Buildings 89:231-244.

4 Dynamic metamodels for hygrothermal simulations

4.1 General information

The research on metamodelling was conducted by Astrid Tijskens, who started in October 2015 as a PhD researcher at KU Leuven. This section gives a short overview of the progress of her work on this subject. To start, a literature study on metamodelling in general and neural networks in specific was performed, in order to get an overall understanding. The knowledge gained by this literature study was used to experiment with several types of metamodels. Different types of neural networks were applied to predict the hygrothermal response of two simplified case studies. Finally, the performance of the most promising neural network type was tested on a fully probabilistic case study. In addition, two conference papers on the use of neural networks for predicting the hygrothermal performance (International Building Physics conference, September 23-26 2018, Central European Symposium on Building Physics, September 2-5 2019) was written, as well as a journal paper (under revision).

To start, an introduction on why metamodelling is useful is given in section 4.2. Next, section 4.3 gives a theoretical summary of artificial neural networks. The application of different neural network types on a two simplified case studies is described in sections 4.4 and 4.5. Finally, in section 4.6, the performance of the most promising neural network type is evaluated on a fully probabilistic case study.

4.2 Introduction

In recent years, traditional deterministic assessments in building physics have evolved towards a probabilistic framework (Annex 55, 2015; Vereecken et al. 2015). When evaluating the hygrothermal behaviour of a building component, there are many inherently uncertain parameters, such as the exterior climate, geometry, material… A probabilistic simulation enables taking into account these uncertainties, which allows evaluating the hygrothermal behaviour and the related damage risks more reliably. However, this often involves thousands of simulations, which easily becomes computationally inhibitive, especially when analysing more-dimensional component connection details. To overcome this time-efficiency issue, the use of metamodels, which aim at imitating the original hygrothermal model with a strongly reduced calculation time, is studied. Many different metamodelling strategies exist, of which multiple linear regression (MLR) or polynomial linear regression might be the most frequently used. MLR attempts to model the relationship between multiple input variables and an output variable by fitting a linear equation. This often performs well when predicting aggregated values such as the total heat loss (Van Gelder et al. 2014). On the other hand, many damage criteria, such as wood decay or mould growth, require evaluation over time, as such damage often has a long incubation time, where after it accumulates. Hence, more advanced metamodelling strategies that can handle time series prediction are needed. Furthermore, the model must be able to capture the highly non-linear and non-stationary pattern of the hygrothermal response of building components. Currently, recurrent neural networks are the state-of-the-art when it comes to time series prediction or sequence modelling. Hence, this research focusses on recurrent neural networks, as they have proven successful in other non-linear and non-stationary research applications.

4.3 Artificial neural networks

Artificial neural networks (ANNs) are information-processing systems, inspired by the biological brain (McCulloch and Pitts, 1988; Rosenblatt, 1963; Rumelhart et al., 1986). The basic structure of an ANN is a network of small processing units or nodes, which represent the neurons, joined to each other by weighted connections representing the synapses between the neurons. Before an ANN can be used to make predictions, it needs to be 'trained'. By presenting the ANN input-output pairs of a training dataset, the network tries to fit its parameters to this data, and learns to predict the correct output.

Many varieties of ANNs were developed over the years, with widely varying properties. Overall, two categories of ANNs exist: ANNs with acyclic connections, and those with cyclic connections. The first type is called feedforward neural networks (FNNs). Feedforward networks are frequently used for supervised learning tasks such as classification and regression. The most widely used form of FNN, and the one we focus on in section 4.3.1, is the multilayer perceptron. ANN with cyclic connections are referred to as feedback, recursive, or recurrent neural networks. These type of networks are particularly interesting when dealing with time-series or sequences and are discussed in Section 4.3.2.

4.3.1 Multilayer perceptron

Since a multilayer perceptron (MLP) does not have cyclic connection between its nodes, all nodes can be arranged into layers with connections feeding forward from one layer to the next. This is illustrated in Figure 30. The first layer is the input layer, here the input data is presented to the network. The last layer is the output layer, where the network produces its prediction data. In between the input and the output layer, there is at least one hidden layer. The outputs in each layer can be calculated given the outputs from the previous layers.

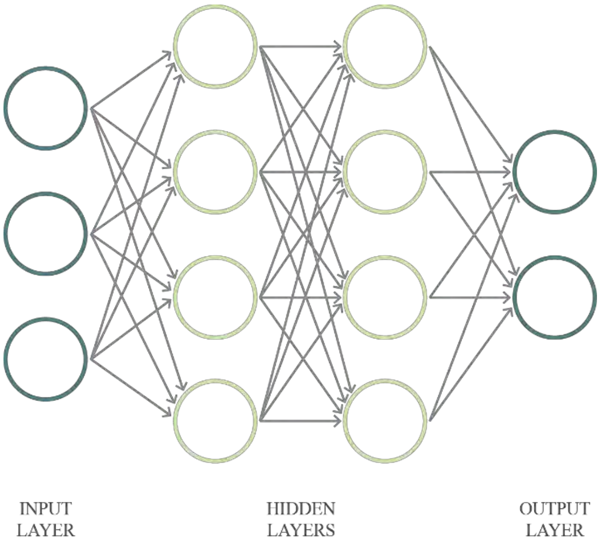


Figure 30: A multilayer perceptron: all nodes can be arranged into layers with connections feeding forward from one layer to the next.

A training dataset $(\boldsymbol{X}, \boldsymbol{Y})$ contains multiple samples, where each sample is an (input data point $\boldsymbol{x}$, output data point $\boldsymbol{y}$) pair. The real-valued vector $\boldsymbol{X}$ contains S input data points $\left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots, \boldsymbol{x}^{(S)}\right)$ where S denotes the total number of samples. Each input data point $\boldsymbol{x}^{(s)}$ is a real-valued vector containing the different features $(x_1^{(s)}, x_2^{(s)}, \dots, x_f^{(s)})$. Thus, the input vector $\boldsymbol{X}$ has two dimensions: the number of samples S and the number of features f. A feature is an input parameter, e.g. the exterior temperature. Hence, the input layer contains as many nodes as there are input features. The real-valued vector $\boldsymbol{Y}$ contains S output data points $\left(\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(S)}\right)$. Each output data point $\boldsymbol{y}^{(s)}$ is a real-valued vector containing the different targets $(y_1^{(s)}, y_2^{(s)}, \ldots, y_K^{(s)})$. Thus, the output vector $\boldsymbol{Y}$ has two dimensions: the number of samples S and the number of targets K. A target is a parameter the network tries to predict, e.g. the temperature at a certain position in the building component. Hence, the output layer contains as many nodes as there are output targets. The data points predicted by a model are labelled $\hat{\boldsymbol{y}}^{(s)}$. For convenience, the superscript (s) is sometimes left out.

4.3.1.1 Forward pass

The network architecture consists of nodes j and biases b. Each node j is associated with an activation function $l_j(\cdot)$. Each connection from node i to j is associated with a weight w_{ji}. This notation follows the convention adopted in several foundational papers (Hochreiter and Schmidhuber, 1997; Gers et al., 2000; Gers, 2001, Sutskever et al., 2011) and denotes the 'to-from' weight corresponding to the directed connection to node j from node i. The value v_j of each node j is calculated by applying its activation function to a weighted sum of the values of its input nodes (Equation 4-1).

$$v_j = l_j\left(\sum_i w_{ji} \cdot v_i\right) = l_j(a_j)$$ **Equation 4-1**

The weighted sum inside the parentheses of Equation 4-1 is called the incoming activation and denoted as a_j. This computation is represented in diagrams by depicting nodes as circles and connections as arrows joining them (Figure 31). When appropriate, the exact activation function is indicated with a symbol, e.g. σ for sigmoid.

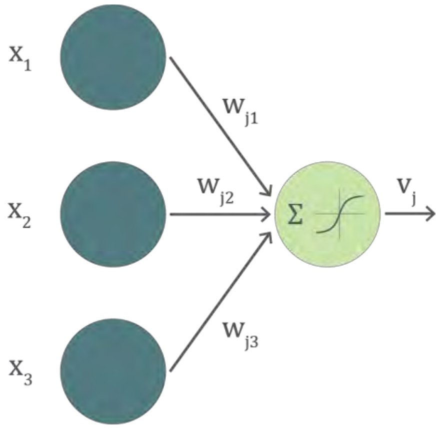


Figure 31: Each node in an artificial neural network computes a (nonlinear) function of a weighted sum of its inputs.

Several neural network activation functions are plotted in Figure 32. The most common choices are the hyperbolic tangent

$$\tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$ **Equation 4-2**

and the logistic sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$ **Equation 4-3**

The two functions are related by the following linear transformation:

$$\tanh(x) = 2\sigma(2x) - 1$$ **Equation 4-4**

Hence, they are largely equivalent as activation functions. However, their output ranges are different; if an output between 0 and 1 is required (e.g. the output represents a probability), the logistic sigmoid should be used. An crucial feature of both the hyperbolic tangent and the logistic sigmoid is their nonlinearity. Nonlinear neural networks are more powerful than linear ones since they can model nonlinear equations. Nonlinear networks can gain considerable power by using successive hidden layers (Hinton et al., 2006; Bengio and LeCun, 2007), in contrast to linear networks, where multiple linear layers have the same processing capacity as a single hidden layer. Another key property is that both functions are differentiable, which allows the network to be trained with gradient descent (see further).

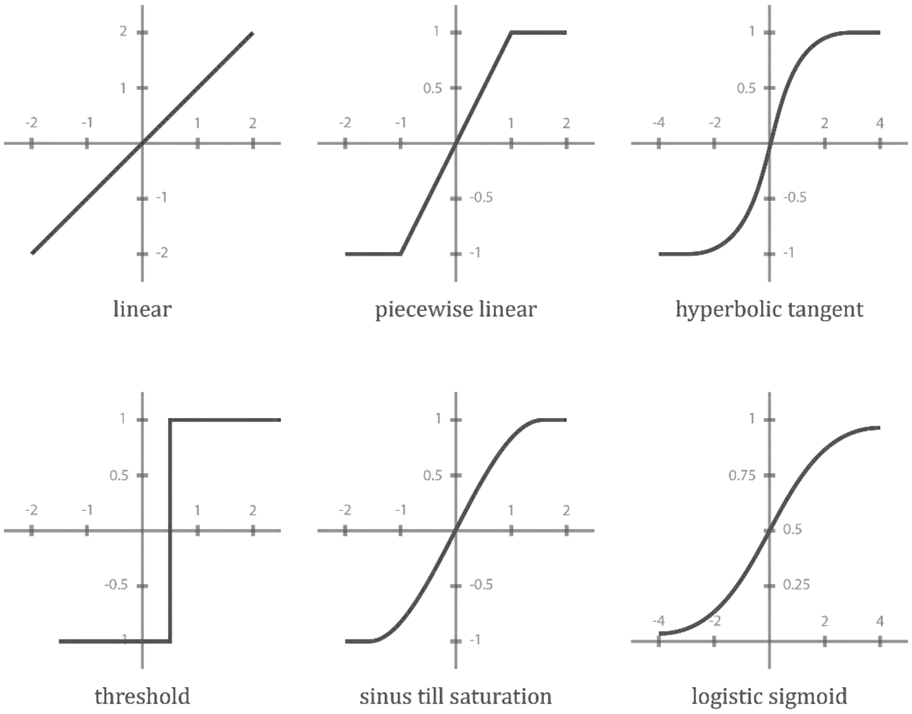


Figure 32: Neural network activation functions.

The activation function at the output nodes depends on the application. For regression tasks, typically a linear output is used. For multiclass classification, though outside the scope of this research, usually a softmax nonlinearity is applied.

4.3.1.2 Loss function

When an MLP is trained, it is presented with input-output pairs. An input data point $\boldsymbol{x}$ is presented to the input layer, each hidden layer is successively computed, until an output $\hat{\boldsymbol{y}}$ is generated. This process is the forward pass of the network. Learning is accomplished by iteratively updating each of the weights to minimize a loss function $\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y})$, which penalizes the distance between the predicted output $\hat{\boldsymbol{y}}$ and the true output $\boldsymbol{y}$. The mean squared error (MSE) is by far the most popular loss function. This loss function ensures that a large error receives much greater attention than a small error. The MSE criterion is optimal and results in a maximum likelihood estimation of the weights if the distributions of the feature vectors are Gaussian (Chauvin and Rumelhart, 1995). This is desirable for most applications. In some situations, other loss functions such as the mean absolute error, maximum absolute error, and median squared error, may be preferred.

4.3.1.3 Backpropagation

The most successful learning algorithm is backpropagation (Rumelhart et al., 1986). The backpropagation algorithm tries to minimise the loss function, using an optimal set of network parameters (weights and biases). To do this, backpropagation calculates the partial derivative of the loss function $\mathcal{L}$ with respect to each parameter in the network, by iteratively applying the chain rule. Because the loss is a function of the outputs from the network, the errors are calculated backwards, starting from the output layer. Once the input layer is reached, the network parameters are updated by gradient descent.

To calculate the gradient, backpropagation proceeds as follows. First, one input data point $\boldsymbol{x}$ is propagated forward through the network: at each node a value v_j is calculated and in the output layer the outputs $\hat{\boldsymbol{y}}$ are produced. At each output node k, a loss function value $\mathcal{L}(\hat{y}_k, y_k)$ is computed. Subsequently, for each output node k, the error δ_k is calculated as follows:

$$\delta_k = \frac{\partial \mathcal{L}(\hat{y}_k, y_k)}{\partial \hat{y}_k} \cdot l'_k(a_k)$$ **Equation 4-5**

with

$$\frac{\partial \mathcal{L}(\hat{y}_k, y_k)}{\partial \hat{y}_k} = \hat{y}_k - y_k$$ **Equation 4-6**

and where l'_k measures how fast the activation function l_k changes with its incoming activation a_k – it is the derivative of l_k with respect to a_k. Given these errors δ_k, the error δ_j for each node j in the previous layer is calculated as follows:

$$\delta_j = l'(a_j) \sum_k \delta_k \cdot w_{kj}$$ **Equation 4-7**

This step is repeated for each previous layer to yield the error δ_j for every node j given the error for each node connected to j by an outgoing connection. Thus, the error is ‘backpropagated’, hence the name of the algorithm.

This process of calculating the error δ_j for every node j is repeated for each input data point $\boldsymbol{x}^{(s)}$ of the training set. Each value $\delta_j^{(s)}$ represents the derivative $\partial \mathcal{L} / \partial v_j^{(s)}$ of the loss function with respect to that node’s incoming activation, for one input data point. Given the values $v_j^{(s)}$ calculated during the forward pass, and the values $\delta_j^{(s)}$ calculated during the backward pass, the overall error terms Δ_j for every node j are obtained as follows:

$$\Delta_j = \sum_s \delta_j^{(s)} \cdot v_j^{(s)} \qquad \textbf{Equation 4-8}$$

Finally, the partial derivative of the loss function $\mathcal{L}$ with respect to a parameter w_{ji} is:

$$\frac{\partial \mathcal{L}}{\partial w_{ji}} = \frac{1}{S}\Delta_j \qquad \textbf{Equation 4-9}$$

4.3.1.4 Gradient descent algorithm

Once the derivative of the loss $\mathcal{L}$ with respect, to the network parameters is calculated, the weights are updated by a gradient descent optimization algorithm. Because the loss surface is non-convex, there is no assurance that backpropagation will reach a global minimum. However, several optimization techniques exist that allow escaping local minima. The most common optimization algorithm is stochastic gradient descent (SGD) using mini-batches. With batch size (i.e. the number of samples presented to the network before updating its weights) equal to one, the stochastic gradient update equation can be written as follows:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} F_i \qquad \textbf{Equation 4-10}$$

where η is the learning rate and $\nabla_{\boldsymbol{w}} F_i$ is the gradient of the objective function with respect to the parameters $\boldsymbol{w}$ as calculated on a single example (x_i, y_i). Here, $\nabla_{\boldsymbol{w}} F_i$ is equal to $\frac{\partial \mathcal{L}}{\partial w_{ji}}$. Many variants of SGD are used to accelerate learning. AdaGrad (Duchi et al., 2011), AdaDelta (Zeiler, 2012), and RMSprop (Hinton et al., 2012) tune the learning rate adaptively for each feature. The popular algorithm AdaGrad adapts the learning rate by caching the sum of squared gradients with respect to each parameter at each time step. The step size for each feature is multiplied by the inverse of the square root of this cached value. AdaGrad leads to fast convergence on convex error surfaces, but because the cached sum is monotonically increasing, the method has a monotonically decreasing learning rate, which may be undesirable on highly non-convex loss surfaces. RMSprop modifies AdaGrad by introducing a decay factor in the cache, changing the monotonically growing value into a moving average. Other common SGD variants are momentum methods, which add a decaying sum of the previous updates to each update. When the momentum parameter is tuned well and the network is initialized well, momentum methods can train recurrent networks to levels of performance that were previously only achievable with computationally expensive Hessian-free optimization (Sutskever et al., 2013).

4.3.1.5 Time-series modelling

In time-series prediction, the MLP is sometimes used for predicting the next time step based on the current step (Soleimani-Mohseni et al. 2006). However, since the MLP does not have cyclic connections, the output depends solely on the current input, and not on any past inputs. Thus, each step of the sequence is processed independently. Hence, the MLP cannot model input-output relations that span multiple time steps.

When using MLP's for time-series modelling, the different samples S are no longer independent samples but each represent a time steps of a sequence. However, the MLP has no notion about this time correlation, hence we keep the notation superscript s instead of t. Some MLP applications implicitly capture time by concatenating each input data point $\boldsymbol{x}^s$ with some number of its immediate predecessors $\boldsymbol{x}^{s-1}$, $\boldsymbol{x}^{s-2}$, …, presenting the MLP with a sliding window of context about each data point (Cottrell et al., 1995; Kemajou et al. 2012). We refer to this type of network as a multilayer perceptron with a time window (MLP TW). The advantage of this approach is that this is an easy way to incorporate the time-aspect of the dataset, without needing to reach to the more computational expensive recurrent networks. On the other hand, the MLP TW fails to capture patterns outside of this window and as the window size needs to be determined beforehand, a considerable number of experiments is required to identify the optimum time window. Furthermore, this method only allows small sliding windows, as the dataset increases drastically with the window step size.

In section 4.3.3 we will apply both the MLP and the MLP TW as metamodel for predicting the hygrothermal behaviour of a masonry wall.

4.3.2 Recurrent neural network

Recurrent neural networks are feedforward neural networks augmented by the inclusion of connections over adjacent time steps, introducing a notion of time into the model. These cyclic connections, called recurrent connections, connect the nodes within a hidden layer across time, but cannot connect nodes from different layers.

As we are dealing with time-series, both the input vector $\boldsymbol{X}$ and the output vector $\boldsymbol{Y}$ are three-dimensional. A training set contains multiple samples, where each sample is an (input sequence, output sequence) pair, denoted $(\boldsymbol{x}^{(s)}, \boldsymbol{y}^{(s)})$. An input sequence is denoted $(\boldsymbol{x}^{(s)(1)}, \boldsymbol{x}^{(s)(2)}, \dots, \boldsymbol{x}^{(s)(T)})$ where each data point $\boldsymbol{x}^{(s)(t)}$ is a real-valued vector containing the different features $(x_1^{(s)(t)}, x_2^{(s)(t)}, \dots, x_f^{(s)(t)})$. An output sequence is denoted $(\boldsymbol{y}^{(s)(1)}, \boldsymbol{y}^{(s)(2)}, \dots, \boldsymbol{y}^{(s)(T)})$, where each data point $\boldsymbol{y}^{(s)(t)}$ is a real-valued vector containing the different targets $(y_1^{(s)(t)}, y_2^{(s)(t)}, \dots, y_K^{(s)(t)})$. The maximum time index of the sequence is called T. The sequences predicted by a model are labelled $\hat{\boldsymbol{y}}^{(s)(t)}$. For convenience, the superscript (s) is sometimes left out.

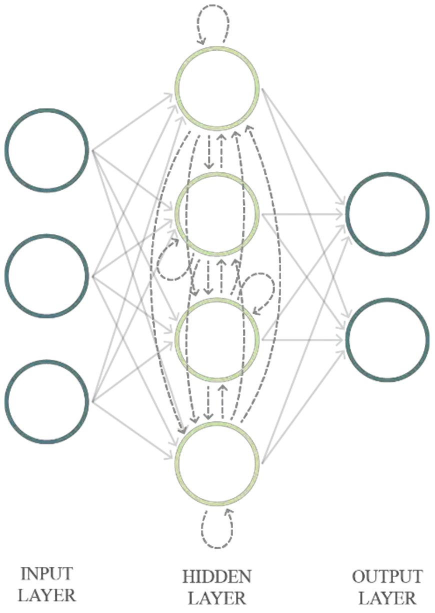


Figure 33: A recurrent neural network: recurrent connections connect nodes within the same layer across time.

4.3.2.1 Forward pass

At time t, nodes with recurrent connections receive input from the current data point $\boldsymbol{x}^{(t)}$ and also from hidden node values $\boldsymbol{h}^{(t-1)}$ in the network's previous state. The output $\widehat{\boldsymbol{y}}^{(t)}$ at each time t is calculated given the hidden node values $\boldsymbol{h}^{(t)}$ at time t. Input $x^{(t-1)}$ at time $t-1$ can influence the output $\widehat{\boldsymbol{y}}^{(t)}$ at time t and later through the recurrent connections in the hidden layers. The following two equations specify all calculations necessary for each time step in a simple recurrent neural network with one hidden layer, as shown in Figure 33:

$$\boldsymbol{h}^{(t)} = \sigma\left(W^{hx}\boldsymbol{x}^{(t)} + W^{hh}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}_h\right) \qquad \textbf{Equation 4-11}$$

$$\widehat{\boldsymbol{y}}^{(t)} = c\left(W^{yh}\boldsymbol{h}^{(t)} + \boldsymbol{b}_y\right) \qquad \textbf{Equation 4-12}$$

Here W^{hx} is the matrix of conventional weights between the input and the hidden layer and W^{hh} is the matrix of recurrent weights between the hidden layer and itself at the previous time step. The vectors $\boldsymbol{b}_h$ and $\boldsymbol{b}_y$ are bias parameters which allow each node to learn an offset. The complete sequence of hidden node values $\boldsymbol{h}$ can be calculated by starting at $t = 1$ and recursively applying Equation 4-11, incrementing t at each step. The principle for multiple past time steps is similar.

4.3.2.2 Unfolding

The dynamics of the network depicted in Figure 33 across time steps can be visualized by unfolding it as in Figure 34. By unfolding, the network can be interpreted not as cyclic, but rather as a deep network with one layer per time step and shared weights across time steps. Hence, it is clear that the unfolded network can be trained across many time steps using backpropagation.

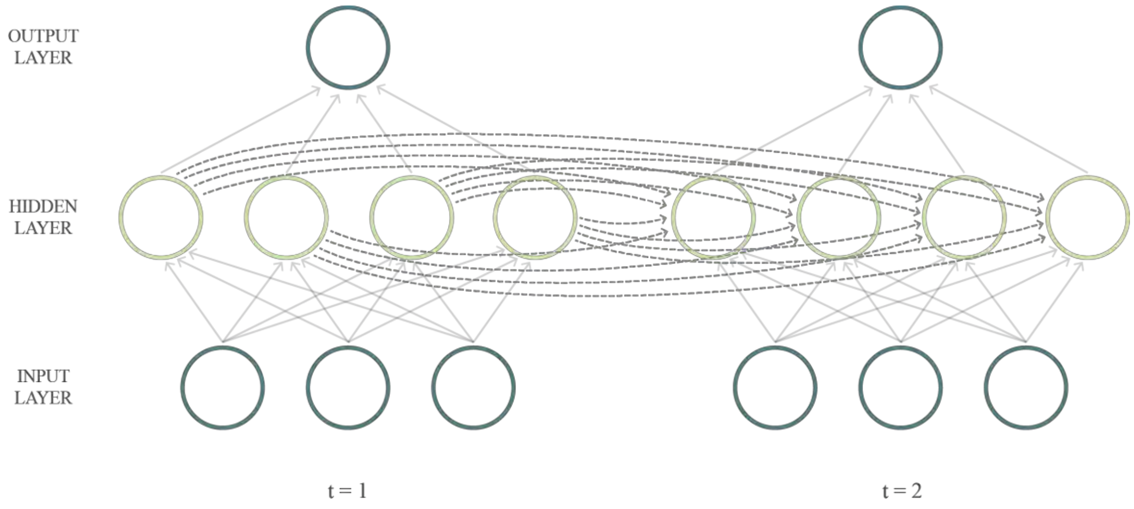


Figure 34: The recurrent network of Figure 33, unfolded across time steps.

4.3.2.3 Backpropagation through time

Backpropagation through time (BPTT) was introduced by Werbos (1990) and all recurrent networks in common current use apply this algorithm for their training. Given the partial derivatives of some differentiable loss function $\mathcal{L}$ with respect to the network outputs, the next step is to determine the derivatives with respect to the weights. Like standard backpropagation, BPTT consists of a repeated application of the chain rule. The subtlety is that, for recurrent networks, the loss function depends on the activation of the hidden layer not only through its influence on the output layer, but also through its influence on the hidden layer at the next time step. Therefore

$$\delta_j^{(t)} = l'\left(a_j^{(t)}\right)\left(\sum_k \delta_k^{(t)} \cdot w_{kj} + \sum_i \delta_i^{(t+1)} \cdot w_{ji}\right) \qquad \textbf{Equation 4-13}$$

Each value δ_j represents the derivative $\partial\mathcal{L}/\partial a_j$ of the total loss function with respect to that node's incoming activation. The complete sequence of errors δ can be calculated by starting at $t = T$ and recursively applying Equation 4-13, decrementing t at each step. Note that $\delta_j^{(T+1)} = 0 \ \forall j$, since no error is received from beyond the end of the sequence. Finally, keeping in mind that weights are shared across time steps, we sum over the whole sequence to get the derivatives with respect to the network weights:

$$\frac{\partial\mathcal{L}}{\partial w_{ji}} = \sum_t \frac{\partial\mathcal{L}}{\partial a_j^{(t)}} \frac{\partial a_j^{(t)}}{\partial w_{ji}} = \sum_t \delta_j^{(t)} v_i^{(t)} \qquad \textbf{Equation 4-14}$$

4.3.2.4 Vanishing or exploding gradient

Training recurrent networks can be challenging due to the difficulty of learning long-range dependencies. When errors are backpropagated across many time steps, the problem of vanishing and exploding gradients may occur (Bengio et al. 1994). As described in section 4.3.2.3 and 4.3.1.3, each of the network's weights receives an update proportional to the partial derivative of the loss function with respect to the current weight in each iteration of training. If the weight along the recurrent connection is less than one, the contribution of the input at the first time step to the output at the final time step (i.e. the gradient) will decrease exponentially fast as a function of the length of the time interval in between. This is called the vanishing gradient problem, which can cause the network to stop learning. Vice versa, if the weight along the recurrent connection is more than one, the gradient will increase exponentially fast. This is called the exploding gradient problem, which can cause the network to become unstable. The latter can be solved relatively easily, because it can be truncated or squashed. Truncated backpropagation through time (TBPTT) (Williams and Zipser, 1989) imposes some maximum number of time steps along which error can be propagated. While TBPTT with a small cut-off can be used to alleviate the exploding gradient problem, it sacrifices the ability to learn long-range dependencies. Another solution is to clip the gradient at a pre-defined threshold (Pascanu et al., 2013). Vanishing gradients are more problematic because it is not obvious when they occur or how to deal with them. In the next section, a recurrent neural network architecture is described that uses carefully designed nodes as a solution to the vanishing gradient problem.

4.3.2.5 Network architectures

As with feedforward networks, many varieties of RNN have been proposed, such as Elman networks (Elman, 1990), Jordan networks (Jordan, 1990), time delay neural networks (Lang et al., 1990) and echo state networks (Jaeger, 2001). The most successful RNN architecture, however, is the Long Short-Term Memory network (LSTM), developed by Hochreiter and Schmidhuber (1997). More recently, the Gated Recurrent Unit (GRU) was proposed by Cho et al. (2014) and has gained growing interest ever since. The remainder of this section will hence focus on these two types of recurrent networks.

LSTM

Hochreiter and Schmidhuber (1997) introduced the LSTM primarily in order to overcome the vanishing gradient problem. In this model, each ordinary node in a hidden layer is replaced by a memory cell (Figure 35), which could be interpreted as a subnet. Each memory cell contains a recurrent self-connected node, the cell state, ensuring that the gradient can pass across many time steps without vanishing or exploding, and originally two multiplicative units, the ‘gates’, that control the flow of information to the cell state. Later, Gers et al. (2000) improved the LSTM by introducing the forget gate (Figure 36), which enables the cell to flush its internal state. To distinguish references to a memory cell and not an ordinary node, we use the subscript c. The multiplicative gates allow LSTM cells to store and access information over long periods of time, thereby mitigating the vanishing gradient problem. The term ‘gate’ was chosen because its value is used to multiply the value of another node. It the gate’s value is zero, then flow from the other node is cut off. If the gate’s value is one, all flow is passed through.

The input node g_c receives weighted input from the input layer $\boldsymbol{x}^{(t)}$ at the current time step and (along recurrent connections) from the hidden layer at the previous time step $\boldsymbol{h}^{(t-1)}$. The inputs are summed and run through a tanh activation function, although in the original LSTM paper, the activation function is a sigmoid. The input gate i_c is a sigmoidal node that receives the same input as the input node. The value of the input gate multiplies the value of the input node and thus controls the flow of information from new input to the cell state. The cell state s_c is the heart of each memory cell. It has a linear activation and a recurrent self-connection with fixed unit weight. Because this connection spans adjacent time steps with constant weight, error can flow across time steps without vanishing or exploding. This connection is often called the ‘constant error carousel’. The forget gate controls what information from the previous cell state is flushed. With the forget gate, the equation to calculate the cell state on the forward pass is as follows:

$$s_c^{(t)} = g_c^{(t)} \odot i_c^{(t)} + f_c^{(t)} \odot s_c^{(t-1)} \qquad \textbf{Equation 4-15}$$

The output node h_c receives information from the cell state s_c and usually applies a tanh activation function, as this gives the output of each cell the same dynamic range as an ordinary tanh hidden unit. The output gate o_c is a sigmoidal node that receives weighted input from the input layer at the current time step and from the hidden layer at the previous time step. The value of the output gate multiplies the value of the output node and thus controls what output is produced by the memory cell.

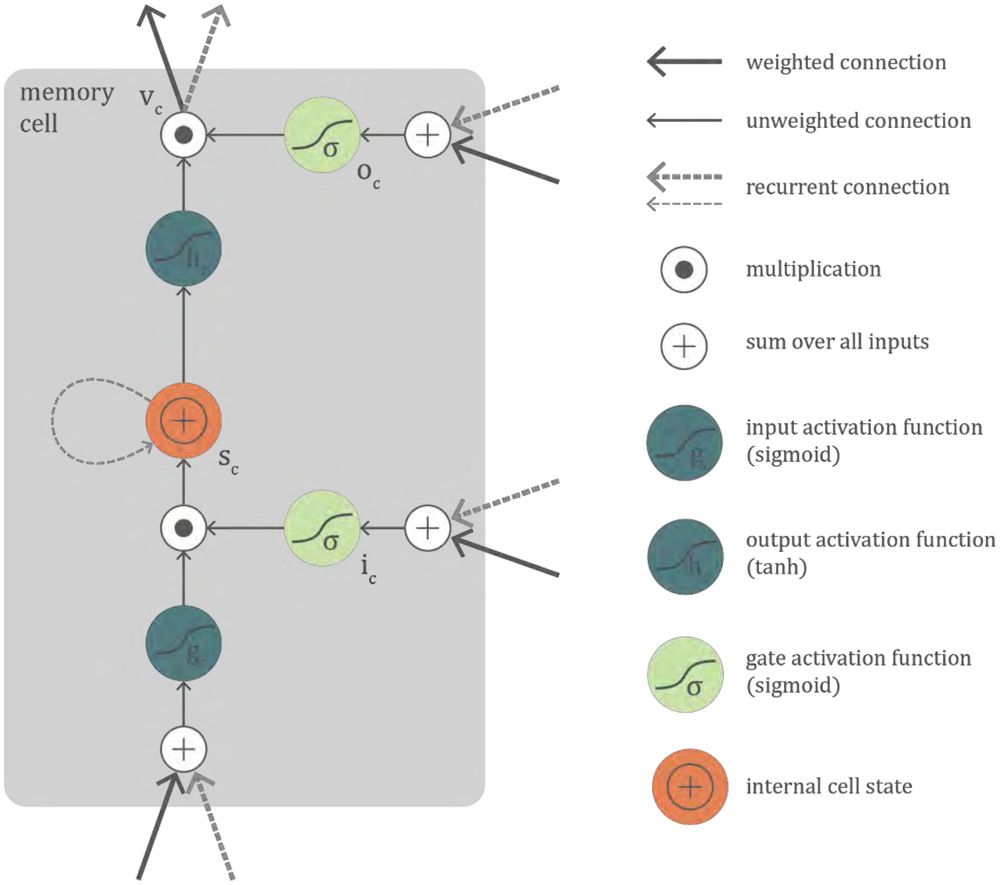


Figure 35: The LSTM cell as originally proposed by Hochreiter and Schmidhuber (1997).

Over the past decades, LSTM has proved successful at a range of real-world problems requiring long range memory, such as music generation (Eck and Schmidhuber, 2002), reinforcement learning (Bakker, 2002), speech recognition (Graves and Schmidhuber, 2005; Graves et al., 2006), natural langauge modelling (Sundermeyer et al., 2012), prediction of air pollution concentration (Xiang et al., 2017), financial marketing predictions (Fischer and Krauss, 2018), and many more.

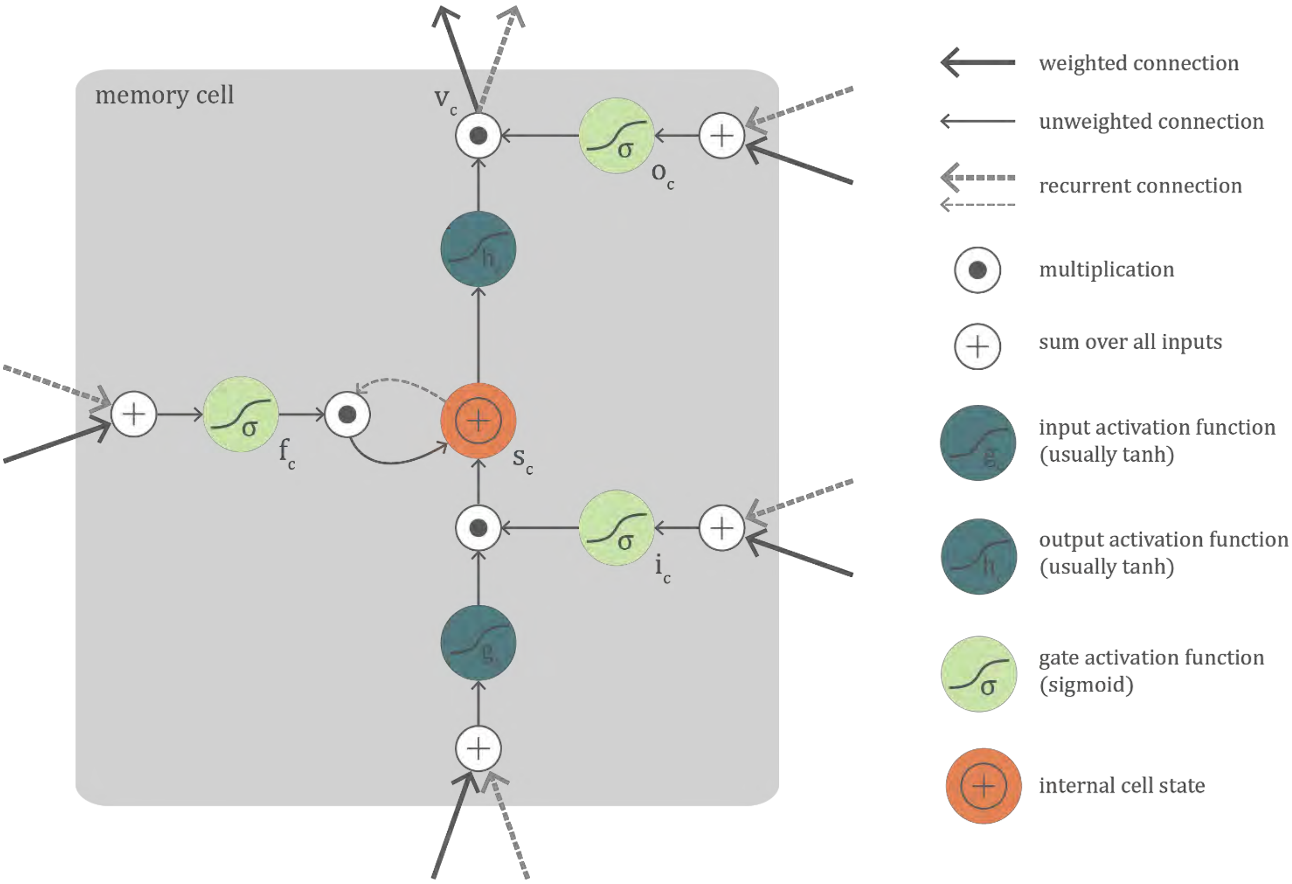


Figure 36: LSTM memory cell with a forget gate as described by Gers et al. (2000).

GRU

Based on the LSTM architecture, Cho et al. (2014a) proposed the gated recurrent unit (GRU), which is much simpler to compute and implement. The GRU (Figure 37) has only two gates, the reset gate r_c and the update gate u_c, which both receive input from the previous hidden state$h_c^{(t-1)}$ and the current input $x^{(t)}$. In contrast to the LSTM, the GRU does not haven an internal state but only a hidden state. The output of the GRU is calculated as follows:

$$v_c^{(t)} = u_c h_c^{(t-1)} + (1 - u_c) s_c^{(t)}$$ **Equation 4-16**

Where,

$$s_c^{(t)} = \tanh\left(W x^{(t)} + U\left(r_c^{(t)} \odot h_c^{(t-1)}\right)\right)$$ **Equation 4-17**

Taking a linear sum between the previous state and the newly computed state is similar to the functioning of the forget gate and the input gate of the LSTM. The GRU, however, does not have a mechanism to control the degree to which its hidden state is exposed, but exposes the whole state each time. When the reset gate is close to 0, the hidden state is forced to ignore the previous hidden state and reset with the current input only. This effectively allows the hidden state to drop any information that is found to be irrelevant later in the future, thus allowing a more compact representation. The update gate, on the other hand, controls how much information from the previous hidden state will flow to the current hidden state. As each hidden unit has separate reset and update gates, each hidden unit will learn to capture dependencies over different time scales. The units capturing short-term dependencies will tend to have frequently active reset gates, while the units capturing long-term dependencies will have mostly active update gates.

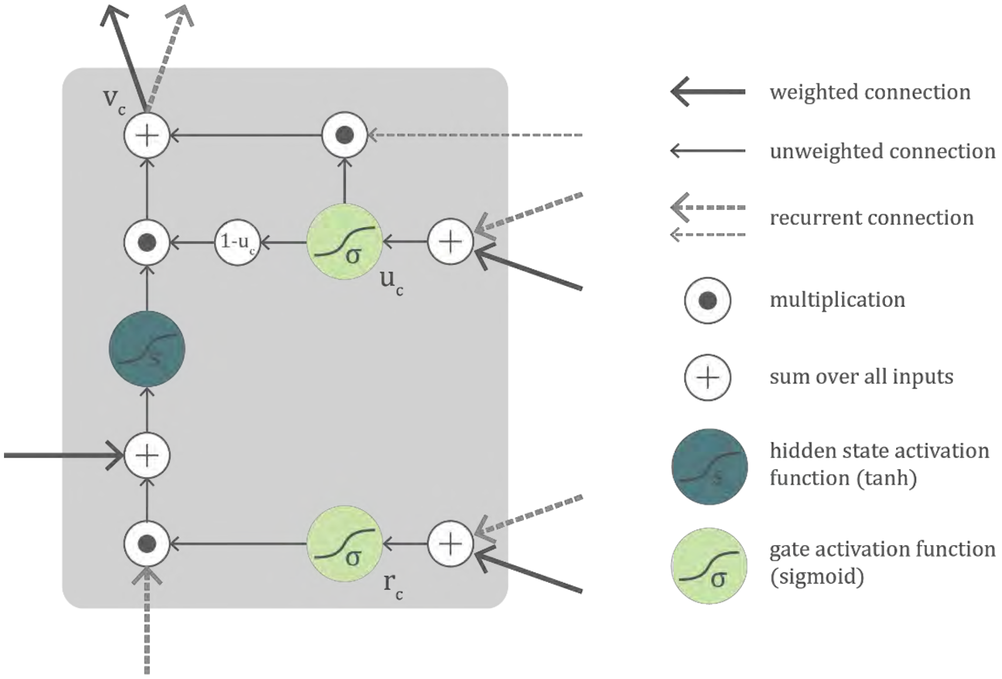
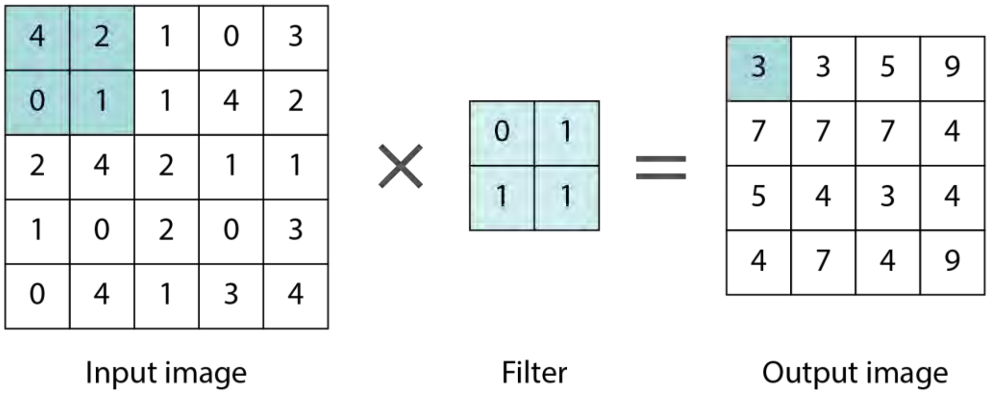


Figure 37: The GRU as proposed by Cho et al. (2014a)

The GRU network has proved successful at a range of real-world problems requiring long range memory, such as contextual video recognition (Jung et al., 2018), financial marketing predictions (Shen et al., 2018), speech recognition (Ravanelli et al., 2017) and many more.

4.3.3 Convolutional neural network

Convolutional neural networks are a class of deep feedforward neural networks most commonly applied to analysing images. The architecture of a CNN is analogous to that of the connectivity pattern of neurons in the human brain, where individual neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. Collections of such fields overlap to cover the entire visual area. The convolution in a CNN copies the response of an individual neuron to visual stimuli. A convolution is a mathematical combination of two functions to produce a third function; it merges two sets of information. In the case of a CNN, the convolution is performed on the input data with the use of a filter to then produce a feature map. The filter slides over the input and at every position, a matrix multiplication is performed. This is repeated for each feature and the result is summed into a new feature map. In case of image processing, the input has four dimensions: samples, width (number of pixels), height (number of pixels) and channels (e.g. three RGB channels). The filter slides over the width and height, for each channel, as shown in Figure 38. The size of the filter determines how many input pixels transfer information to the next layer, i.e. the size of the filter determines the size of the receptive field. By using multiple filters, comparable to using multiple neurons in a hidden layer, different feature maps can be learned.



4	2	1	0	3
0	1	1	4	2
2	4	2	1	1
1	0	2	0	3
0	4	1	3	4

Input image

×

0	1
1	1

Filter

=

3	3	5	9
7	7	7	4
5	4	3	4
4	7	4	9

Output image

Figure 38: A convolutional operation on one channel of an image.

4.3.3.1 Time-series modelling

More recently, CNNs have been applied to sequence learning (van den Oord et al., 2016; Borovykh et al., 2017; Bai et al., 2018). In this case, the input has three dimensions: samples s, time steps t and features f. Hence, the filter is one-dimensional and slides along the time steps. When working with time series or sequences, usually dilated causal convolutions are used. Causal convolutions ensure that the output of the filter does not depend on future inputs. In a dilated convolution, the filter is applied over an area larger than its length by skipping input values with a certain step. Stacked dilated convolutions enable networks to have very large receptive fields with just a few layers, while preserving the input resolution throughout the network as well as computational efficiency. Often, the dilation factor is exponentially increased for every layer, allowing the receptive field to grow exponentially with network depth. This principle is shown in Figure 39.

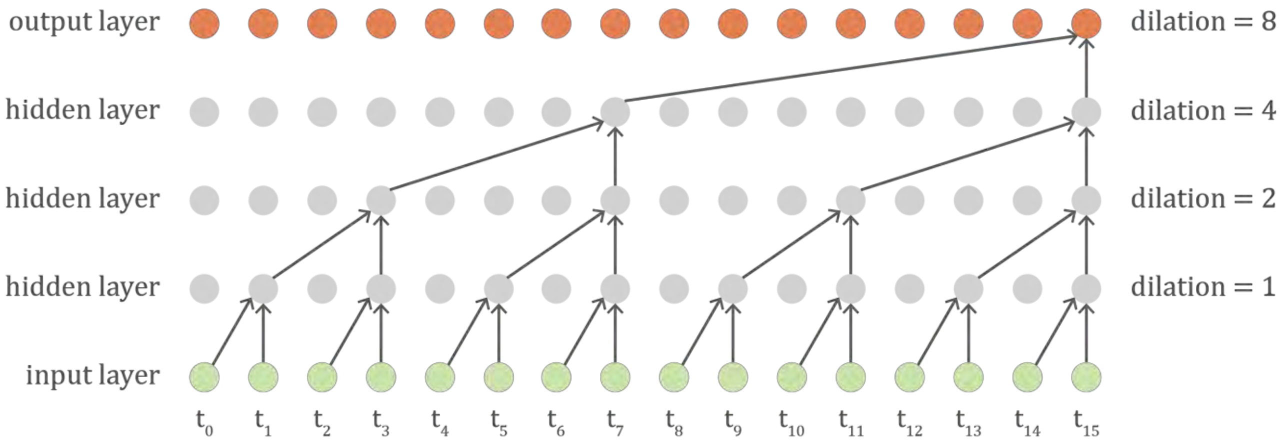

output layer
hidden layer
hidden layer
hidden layer
input layer
dilation = 8
dilation = 4
dilation = 2
dilation = 1
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15}

Figure 39: Due to the causal dilated convolutions, an output time step receives information from more input time steps, i.e. the receptive field, with increasing number of hidden layers.

4.3.3.2 Network architecture

The architecture of the CNN network, shown in Figure 40, was based on the Wavenet architecture (van den Oord et al., 2016). Residual blocks with dilated causal convolutions and gated activation units [40] are stacked to create a larger receptive field. Each block has a residual connection to the next block and a parameterised skip connection to the last two layers of the network, both of which speed up convergence and help training deep networks.

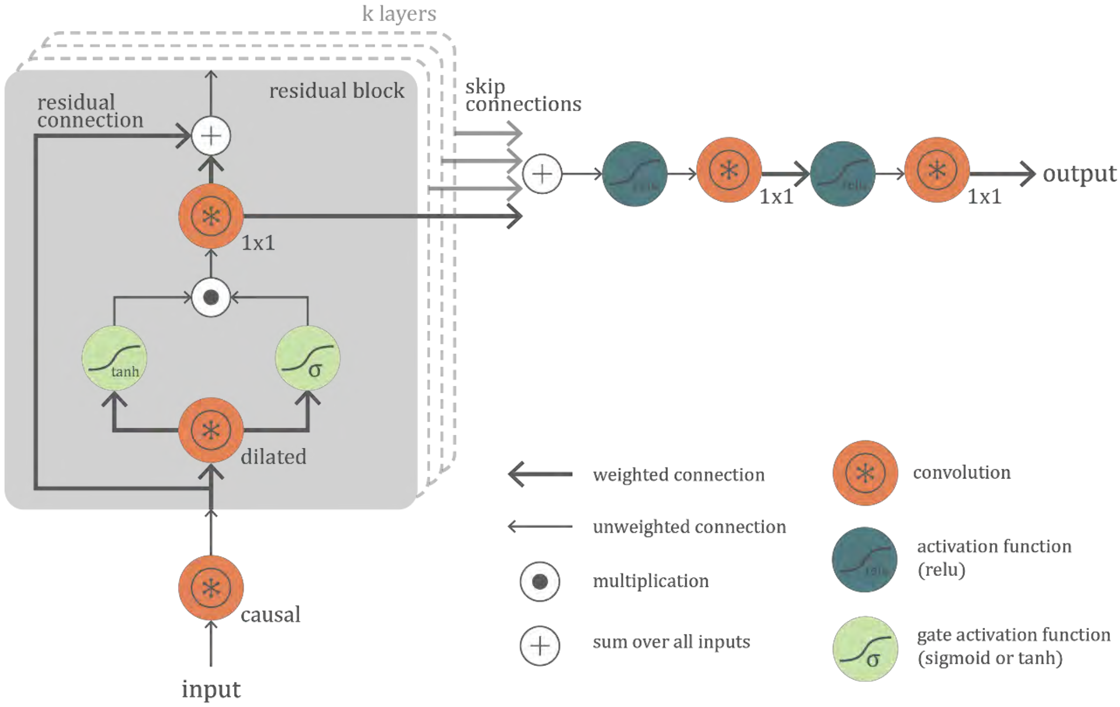


Figure 40: The used CNN architecture with residual blocks and skip connections, based on the Wavenet architecture.

A CNN does not have 'memory' the way the RNN does; the output at a certain time step does not depend on the output on previous time steps, as can be seen in Figure 39. However, thanks to the stacked residual blocks, the CNN has access to a large range of inputs at previous time steps. The advantage of this type of network is that output time steps can be calculated in parallel, which results in a much faster training and predicting compared to the RNN. A disadvantage is that one needs to know how large the receptive field should be when building the network, as information outside the receptive field cannot be used to predict the current time step.

4.4 Explorative application on a very simplified case study

4.4.1 Very simplified case study

To explore the performance of the neural networks described above, they are applied to predict the hygrothermal performance of a 36 cm massive masonry wall without insulation. In this explorative study, the probabilistic aspect of the influencing parameters is not yet taken into account fully, as this allows for a more efficient exploring of several network architectures on a smaller dataset. The training and validation data was obtained via hygrothermal simulations in Delphin 5.8. To estimate whether frost damage, wood decay of embedded wooden beam ends and mould growth will occur, the temperature (T), relative humidity (RH) and moisture content (MC) are monitored at the associated positions for a period of 6 years (see Table 3). The used input parameters are shown in Table 4; the brick material properties are given in Table 5. To account for variability in boundary conditions, different years of the external climate were used, as well as different wall orientations. Since the interior climate is calculated based on the exterior climate, this variability is also included.

In total, 24 samples were simulated, of which 18 samples were used for training and 6 for validation. Given the time series of the external temperature and relative humidity, the wind-driven-rain load, the short-wave radiation and the internal temperature and relative humidity, the neural networks are trained to predict the T, RH and MC time series.

Table 3: The monitored quantities for the damage patterns at different positions in the wall.

Damage pattern	Position	Quantity
Frost damage	0.5 cm from exterior surface	T, RH, MC
Decay of wooden beam ends	5 cm from interior brick surface	T, RH
Mould growth	Interior surface	T, RH

Table 4: Used input parameters for hygrothermal simulations of brick wall.

Input parameter	Value
Brick wall thickness	360 mm
External climate	Gaasbeek, Belgium
Internal climate	cfr. EN 15026 A
Wall orientation	U(0,360)
Rain exposure factor	1
Solar absorption	0.4
Start year	D(1,24)

** U(a,b): uniform distribution between a and b*
D(a,b): discrete distribution between a and b

Table 5: Properties of the used brick type.

Material property	Value
Dry thermal conductivity (W/m²K)	0.87
Dry vapour resistance factor (-)	14
Capillary absorption coefficient (kg/m²s$^{0.5}$)	0.277
Capillary moisture content (m³/m³)	0.25
Saturation moisture content (m³/m³)	0.35

4.4.2 Network architecture

In this study, the MLP, MLP TW, LSTM and GRU are included. Several hyper-parameters should be pre-set before building and training the networks, including the number of hidden layers and the number of nodes in these hidden layers. All constructed networks have a single hidden layer, as comparative experiments showed no benefits using multiple hidden layers. Furthermore, networks with 32, 64, 128 and 256 hidden units were tested. In case of the MLP TW, a time window of 24 hours was explored. A larger time window, which would be required to capture long-term dependencies, resulted in an extensive input dataset, which became too memory-intensive. The networks are trained by minimising the mean squared error (MSE) via backpropagation (MLP and MPL TW) or backpropagation-though-time (LSTM, GRU). Based on the results of comparative experiments, the LSTM and GRU networks were trained using the RMSprop learning algorithm (Hinton et al., 2012) with a learning rate of 0.002. The MLP and MLP TW networks were trained using the Adam learning algorithm (Kingma and Ba, 2015) with a learning rate of 0.001. In general, before presenting data to the network, the data is standardised (zero mean, unit variance) to overcome influences from parameter units. In this study, both the input and output data are standardised, as this was found to improve training speed and accuracy.

4.4.3 Performance evaluation

The networks' accuracy is tested on the validation data set and evaluated by three indicators: the normalised root mean square error (NRMSE), normalised mean absolute error (NMAE), and coefficient of determination (R2), formulated as follows:

$$NRMSE = \frac{\sqrt{\frac{1}{T}\Sigma(y-\hat{y})^2}}{y_{max}-y_{min}} \qquad NMAE = \frac{\frac{1}{T}\Sigma|y-\hat{y}|}{y_{max}-y_{min}} \qquad R2 = 1-\frac{\Sigma(y-\hat{y})^2}{\Sigma(y-\bar{y})^2}$$

Equation 4-18

where y is the true output, $\hat{y}$ is the predicted output, $\bar{y}$ is the mean of the true output and T is the total number of time steps over all samples. The RMSE and the MAE are normalised to remove the scale differences between the different outputs. Although the networks are trained to predict all outputs simultaneously, these performance indicators are calculated for each output separately. This allows assessing which outputs are more difficult to predict and which ones are easy.

4.4.4 Results and discussion

Figure 41 presents the three performance indicators (rows) for all outputs (columns) and each network type. This graph shows that some outputs are easier to predict than others; all networks are capable to predict the temperature accurately, as well as the interior surface relative humidity. Since the wall temperature and surface relative humidity respond almost immediately to a change in boundary conditions, not much memory is needed to accurately predict these profiles; for these outputs, the MLP with a time window of 24 hours is about as accurate as the more complicated LSTM or GRU. The relative humidity and moisture content at 0.5 cm from the exterior surface (frost damage) and the relative humidity at the wooden beam ends (wood decay) appear less evident to model. As moisture is transported inwards only slowly, there is a large delay between a change in boundary conditions (e.g. a heavy rain shower) and the response in relative humidity in the wall. At the wooden beam ends, this response delay can go up to several months. Hence, the MLP and MLP TW, which have no or only limited memory, are incapable to capture these long-term temporal dependencies and perform poorly. The LSTM and GRU, on the other hand, are able to capture these complex long-term patterns because of their connections to information from long-past time-steps.

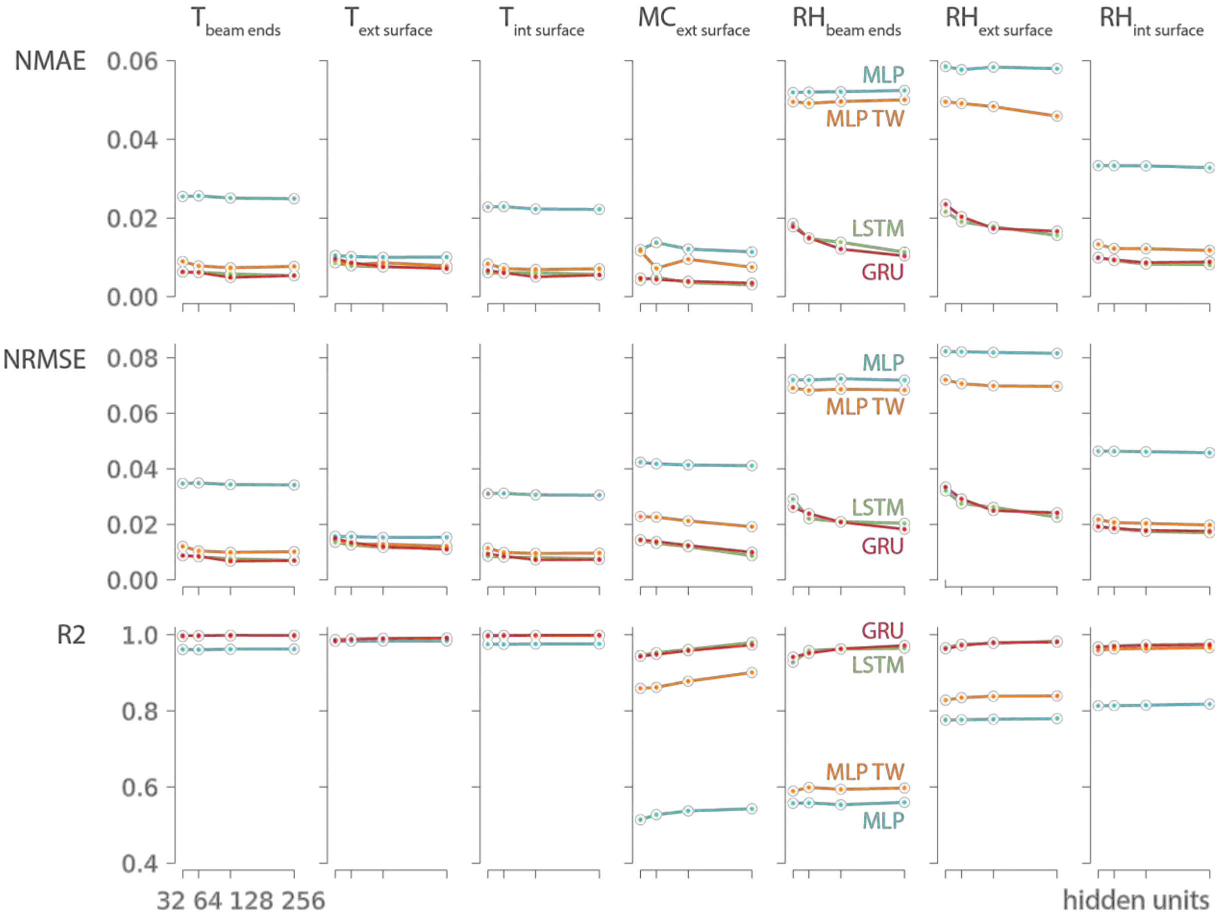


Figure 41: The performance indicators show that the temperature patterns are easy to model, while only the memory networks types (LSTM, GRU) are able to model the moisture content and relative humidity patterns accurately.

Figure 42 to Figure 45 compare the predicted output to the true output time series of one sample not used for training, for each network type with 256 nodes. In case of the LSTM and GRU, the residuals ε_t show that that the error is small on all outputs.

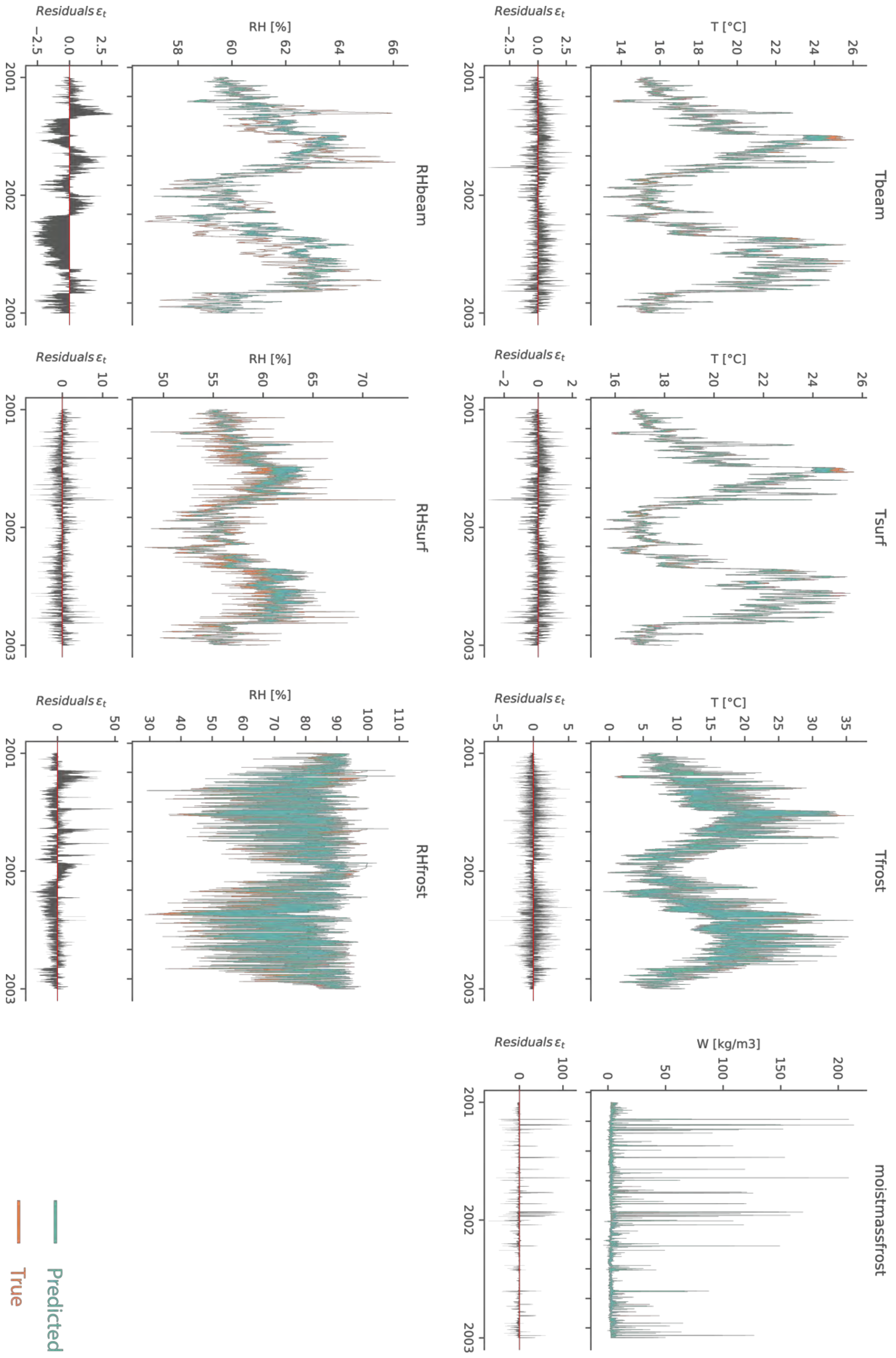


Figure 42: Predicted output (blue), true output (orange) and residuals of one sample of the test set, predicted by an MLP with 256 nodes.

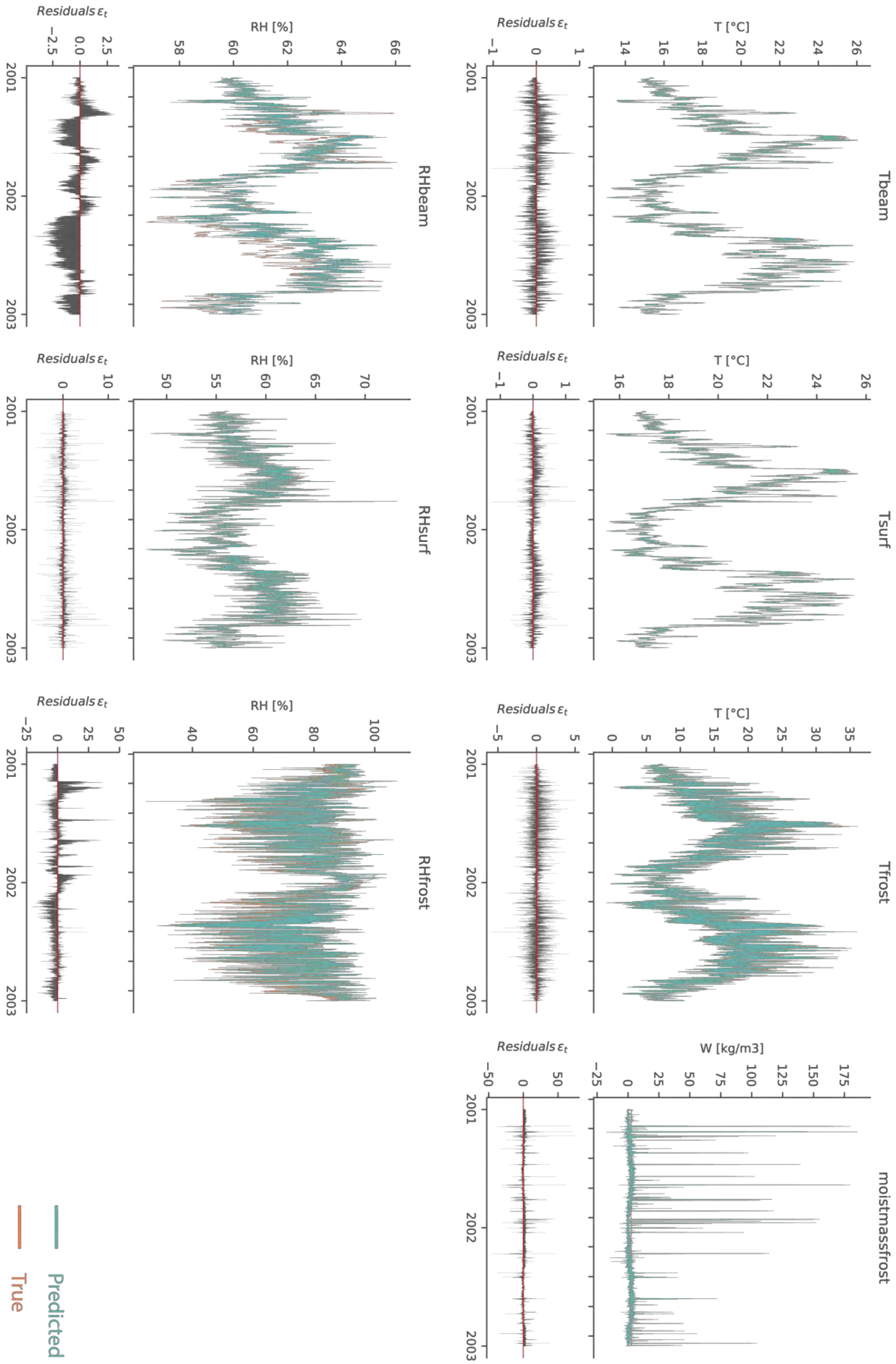


Figure 43: Predicted output (blue), true output (orange) and residuals of one sample of the test set, predicted by an MLP with a time window of 24 hours and 256 nodes.

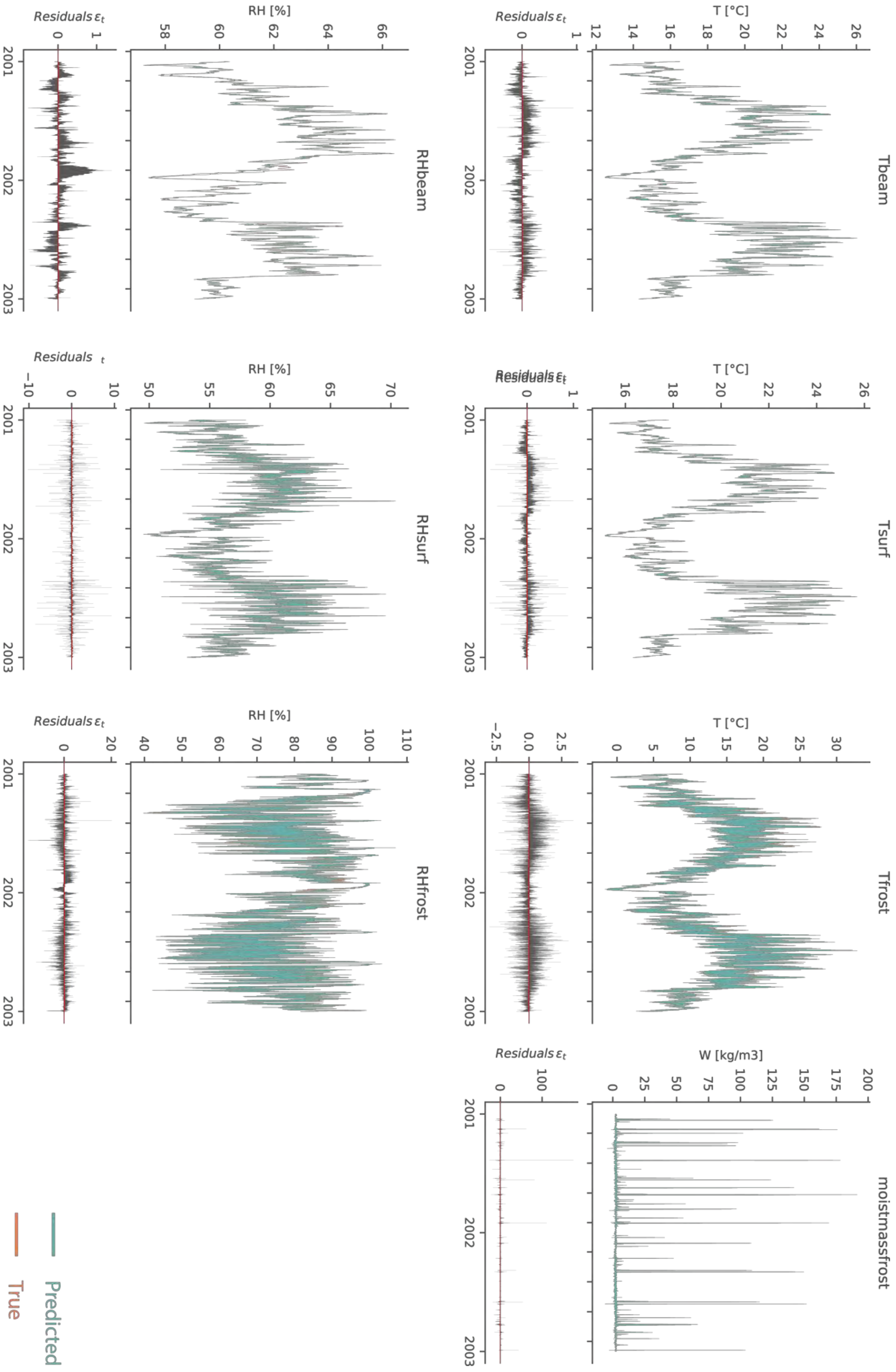


Figure 44: Predicted output (blue), true output (orange) and residuals of one sample not used for training, predicted by an LSTM with 256 nodes.

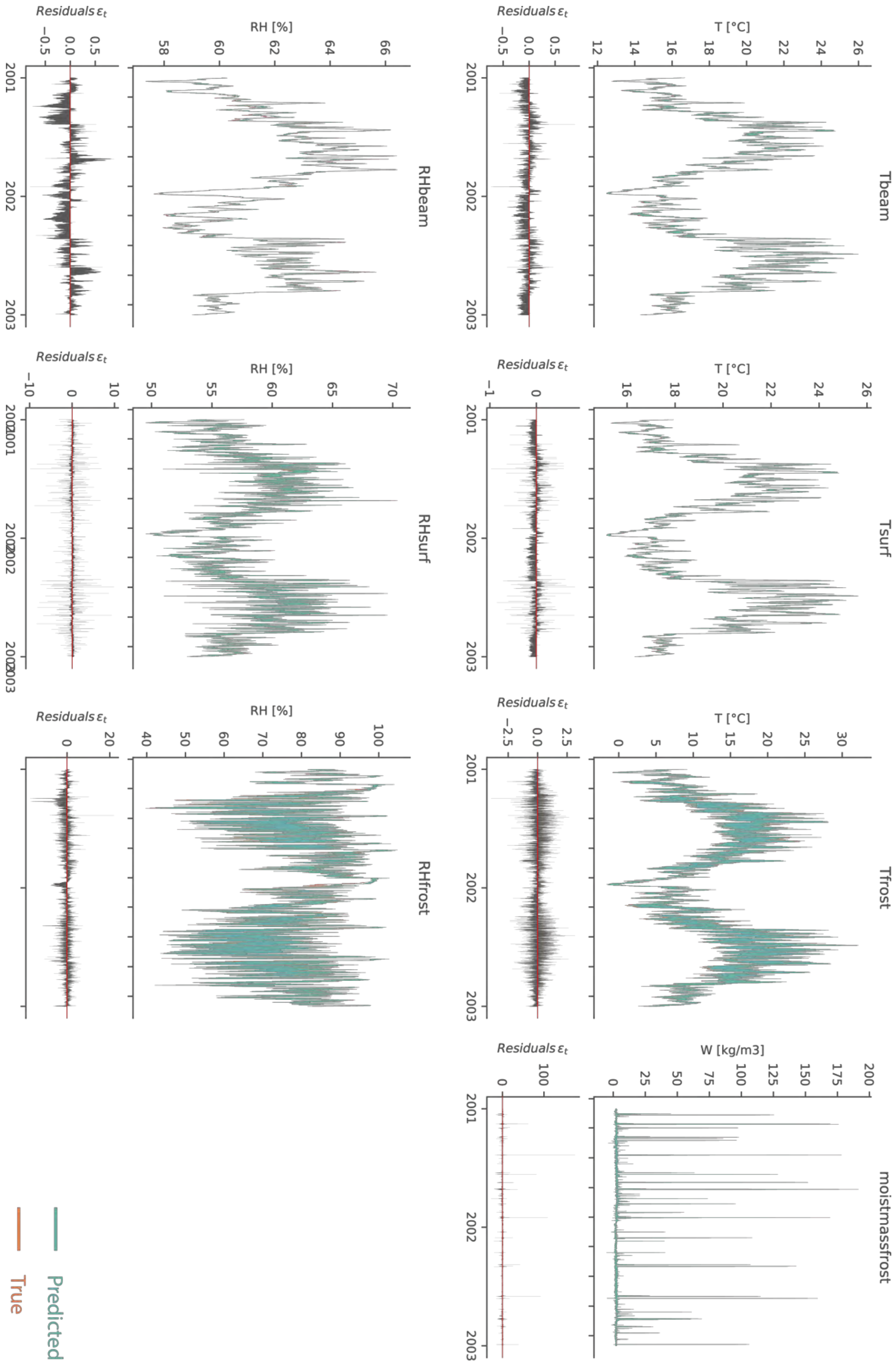


Figure 45: Predicted output (blue), true output (orange) and residuals of one sample not used for training, predicted by a GRU with 256 nodes.

Furthermore, it was found that the number of hidden units appears to have limited effect on the prediction performance in case of the MLP and MLP TW, though it increased the training time (Figure 46). In case of the LSTM and GRU on the other hand, more hidden units resulted in an improved prediction performance and a decrease in training time. It appears that the memory networks converge faster when they have more hidden units. Though prediction accuracy is very similar for both the LSTM and the GRU, the latter required less training time as it has fewer network parameters.

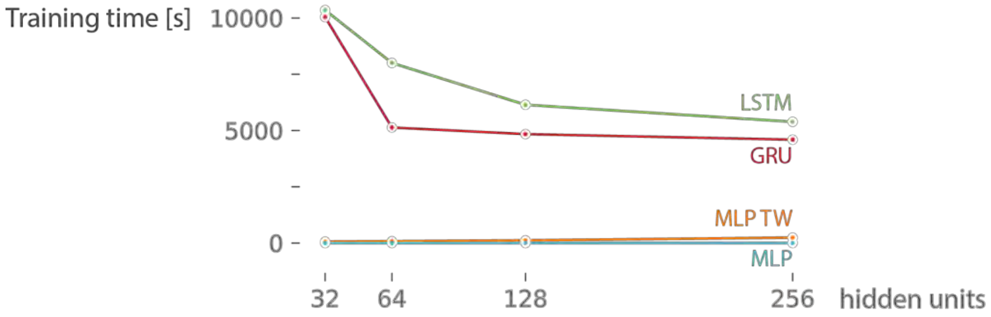


Figure 46: The training time for the memory network types (LSTM, GRU) decreases with increasing number of hidden units. The simulation time for one sample is 1 s for all network types. The reference simulation time for one sample in the original hygrothermal model is about 720 s.

4.4.5 Conclusions

In this study, the hygrothermal simulation model for a massive masonry wall is replaced by a much faster metamodel. Four different types of neural networks were considered as metamodel: a traditional multilayer perceptron (MLP), a multilayer perceptron with a pre-defined time window (MLP TW), a long-short term memory network (LSTM) and a gated recurrent unit network (GRU). Only the last two types have dynamic memory. The MLP TW has only access to the past time steps within the pre-defined time window, and the MLP tries to predict based on the current time step only. It was found that all network types were capable to predict the temperature accurately. Since the temperature responds almost immediately to a change in boundary conditions, not much memory is needed to capture these patterns. By contrast, only the LSTM and the GRU were able to accurately capture the long-term dependencies needed to predict the relative humidity and moisture content, as these respond much slower to a change in boundary conditions. Both types of memory networks were found to have almost equal predicting accuracy, though the GRU converged faster and thus required less training time.

This study shows that it is possible to replace the hygrothermal simulation model with a much faster metamodel. Given the large number of calculations needed for an extensive probabilistic evaluation of the damage risks involved in applying internal insulation, this approach can decrease the total calculation time significantly.

4.5 Explorative application on a simplified case study

4.5.1 Simplified case study

A massive masonry wall is evaluated as a 1D construction and consists of 30 cm solid ceramic brick with 1 cm of plaster as interior finishing. As this is an explorative study, not all influencing probabilistic parameters are taken into account yet, as this allows exploring different network types and architectures more efficiently. The used probabilistic input parameters are shown in Table 7, the ceramic brick material properties are given in Table 7: Input parameter distributions. To account for variability in boundary conditions, future climate data of four Belgian cities (ClimateforCulture, 2014), as well as different years of the data and different wall orientations are used. The wind-driven rain load is calculated based on the wall orientation and the exterior climate data. The interior climate is calculated according to European standard EN 15026, based on the exterior climate, and thus includes variability as well. The inputs are pre-processed to contain only time series, as this simplifies the network architecture. The wall orientation, which is a scalar input parameter, is adopted via the wind-driven rain load and the direct and diffuse short-wave radiation time series. The same output quantities used in the previous case study (Table 3) are used here as well. In total, 80 samples were simulated, of which 64 samples were used for training and 16 for testing.

Table 6: Input parameter distributions

Parameter	Value
Exterior climate	D(Gent; Gaasbeek; Oostende, StHubert)
Exterior climate start year	D(2020; 2047)
Wall orientation degrees from North	U(0; 360)
Interior climate [44]	D(load A; load B)

U(a, b): uniform distribution between a and b
D(a, b): discrete distribution between a and b

Table 7: Input parameter distributions

Material property	value
Dry thermal conductivity (W/m²K)	0.87
Dry vapour resistance factor (-)	140
Capillary absorption coefficient (kg/m²s$^{0.5}$)	0.046
Capillary moisture content (m³/m³)	0.13
Saturation moisture content (m³/m³)	0.24

Before presenting the input and output data to the neural networks, all data are standardised (zero mean, unit variance). This ensures that all input features are on the same scale, which allows weighting all features equally in their representation. To ensure errors are penalised equally for all targets, the output data are standardised as well. Once the networks are trained on the train dataset, their performance is evaluated on the test dataset, as this gives an indication of the networks' performance when used as a replacement of the original model.

4.5.2 Network architecture

In this study, the MLP, LSTM, GRU and CNN network types are included. The architecture of the neural networks is described in section 4.3. Table 8 shows the used hyperparameters of all network types. These parameters are chosen based on a limited number of experiments and thus might not be optimal. To find the most optimal parameters, a hyperparameter optimization is required, which is time intensive. As this is an explorative study only, this has not been covered yet. Additionally, all metamodels were constructed to have a similar number of trainable parameters, for reliable comparison of training time and performance. For each network type, five replications are trained, since different initialisation of the network parameters can lead to different final models.

Table 8: Neural network parameters

NN type	Hidden layers	Hidden neurons	Trainable parameters	Optimizer	Additional parameters
MLP	5	512	$1.1 \cdot 10^6$	Adam, lr = 0.002	
LSTM	3	256	$1.3 \cdot 10^6$	RMSprop, lr = 0.002	
GRU	3	256	$1.0 \cdot 10^6$	RMSprop, lr = 0.002	
CNN	22*	128**	$1.2 \cdot 10^6$	Adam, lr = 0.002	filter width = 4

* Two stacks of 11 layers with dilatation factors {1, 2, 4, 8, … , 512, 1024}

** Number of filters in convolutional layers

4.5.3 Performance evaluation

The prediction accuracy of the different metamodels is evaluated using three performance indicators: the root mean square error (RMSE), the mean absolute error (MAE), and the coefficient of determination (R2), quantified as follows:

$$RMSE = \sqrt{\frac{1}{T}\Sigma(y-\hat{y})^2} \qquad MAE = \frac{1}{T}\Sigma|y-\hat{y}| \qquad R2 = 1 - \frac{\Sigma(y-\hat{y})^2}{\Sigma(y-\bar{y})^2} \qquad \textbf{Equation 4-19}$$

where y is the true output, $\hat{y}$ is the predicted output, $\bar{y}$ is the mean of the true output and T is the total number data points. The performance indicators are calculated both for all outputs together as well as for each output separately. The first allows to easily compare the performance of different models, while the latter allows assessing which outputs are more difficult to predict. Additionally, the models' training time is evaluated.

4.5.4 Results and discussion

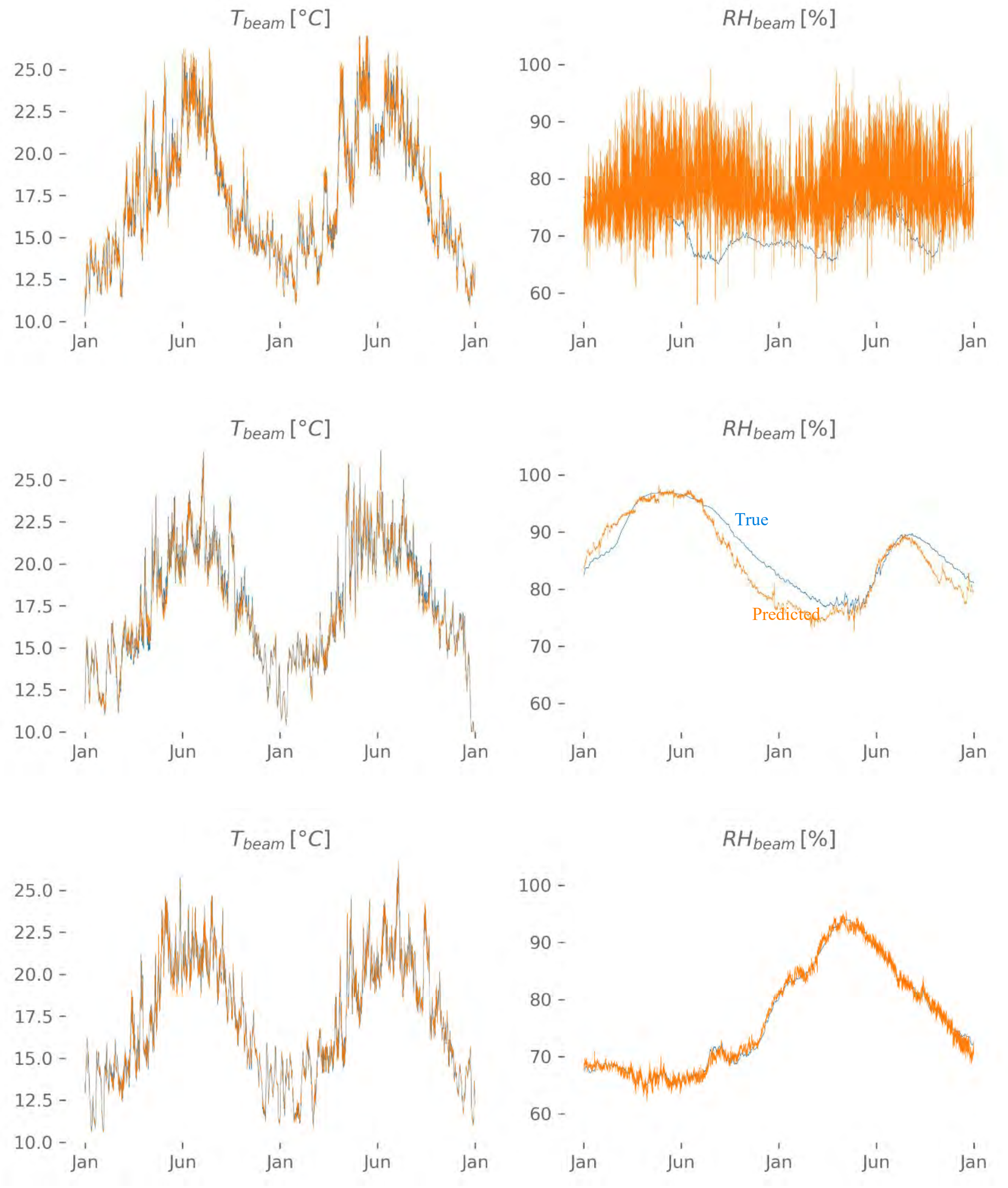


Figure 47: Examples of the predicted temperature and relative humidity course at the position of the wooden beam ends for the MPL (top), LSTM (middle) and CNN (bottom). Note that different samples are simulated. The prediction is shown in orange, the true value in blue.

Figure 47 shows examples of the predictions for the temperature and relative humidity at the wooden beam ends by the different neural network types. For these predictions, samples from the test set were used. This figure clearly show that all networks are able to predict the temperature accurately, but only the networks with knowledge about the past can capture the pattern of the relative humidity.

Figure 48 shows the performance indicators on the test set as defined by Equation 4-19, calculated over all outputs, as well as the training time for each neural network type. These results indicate that the MLP is not suited for time series prediction, due to its lack of information on past time steps. The RNNs and CNN on the other hand, perform almost equally well. However, the CNN is a factor 10 faster to train compared to the RNNs. Whereas the RNN models take some 1,5 to 3,5 hours to train the current problem, the CNN models require only 15 to 20 minutes. The prediction time for these networks also differs: it takes 15 seconds for the RNN models to predict 16 samples, whereas it takes only 5 seconds for the CNN model. As a reference, the time required to simulate one sample with the original hygrothermal software is 4-5 minutes.

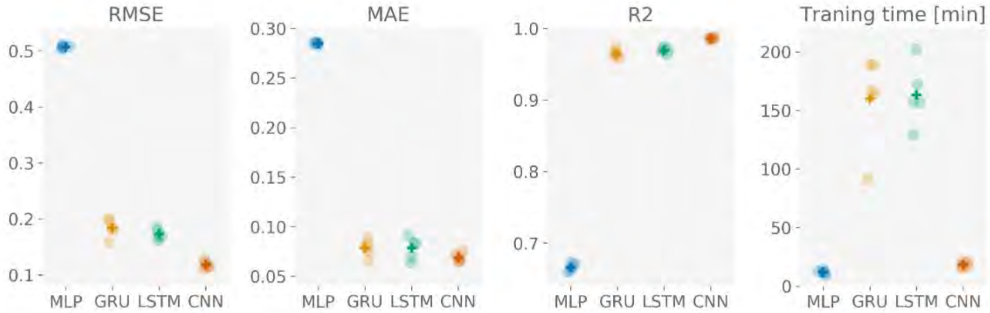


Figure 48: The performance indicators, calculated over all outputs together, and training time for each neural network type. The dots show the results of the five replications, while the crosses represent the average result.

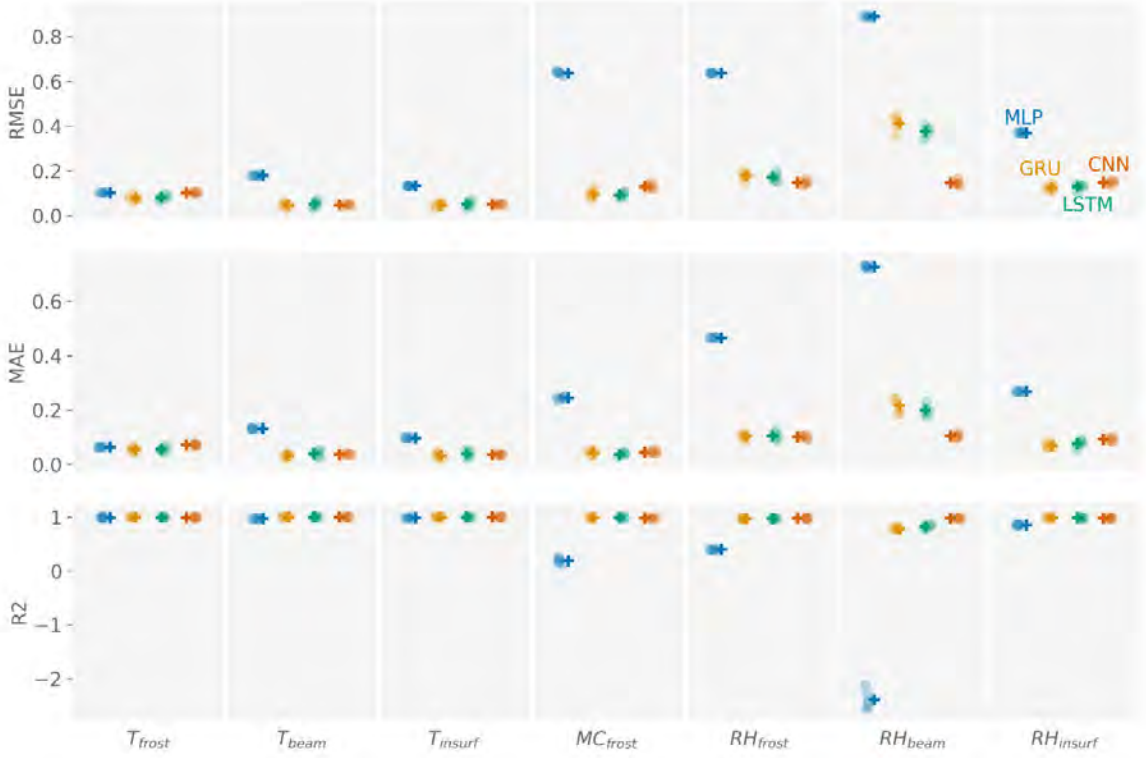


Figure 49: The performance indicators, calculated over each output separately for each neural network type. The dots show the results of the five replications, while the crosses represent the average result.

When looking at the performance indicators for each output separately, shown in Figure 49, it is clear that all models are suited to model the temperature profiles. As was already observed in Figure 47, also the MLP performs adequately with respect to the temperatures. Because the wall temperatures responds almost immediately to a change in boundary conditions, not much memory is needed to accurately predict these time series. The relative humidity and moisture content series appear less evident to model. As moisture is transported inwards only slowly, there is a large delay between a change in boundary conditions (e.g. a heavy rain shower) and the response in relative humidity in the wall. At the wooden beam ends, this response delay can go up to several months. The MLP is clearly incapable to capture these long-term temporal dependencies, due to its lack of memory. The RNN and CNN models, on the other hand, are able to capture these complex long-term patterns because of their connections to information from past time-steps. In case of the relative humidity at the wooden beam ends, where the response delay is the largest, the CNN performs significantly better than the RNN models. RNN models have difficulty learning dependencies from a distant past, because the information needs to persist over a large number of time steps. The CNN on the other hand, has direct access to this long-term information, due to its architecture (see Figure 39). Hence, it is more suited to capture long-term dependencies. This is also visible in Figure 50 and Figure 51, which compare the true output of one test set sample to the output predicted by the LSTM and CNN model, respectively. The residuals (grey) indicate how large the error is between true output (blue) and prediction (orange). Compared to the CNN prediction, the LSTM predicted relative humidity at the beam ends shows larger residuals, and moreover over longer periods. This error is inherent to the LSTM (and GRU) architecture, because each predicted output time step depends on the previous predicted time step(s), which allows errors to accumulate. The CNN has no information on its previous predicted time steps, and thus suffers much less of this problem. However, a different error pattern can be observed in Figure 50; because of this lack of connections to previous predictions, more variability between consecutive output time steps is noted. This causes the predicted time series to be more noisy and explains why the CNN model performs slightly worse when predicting the moisture content or temperature.

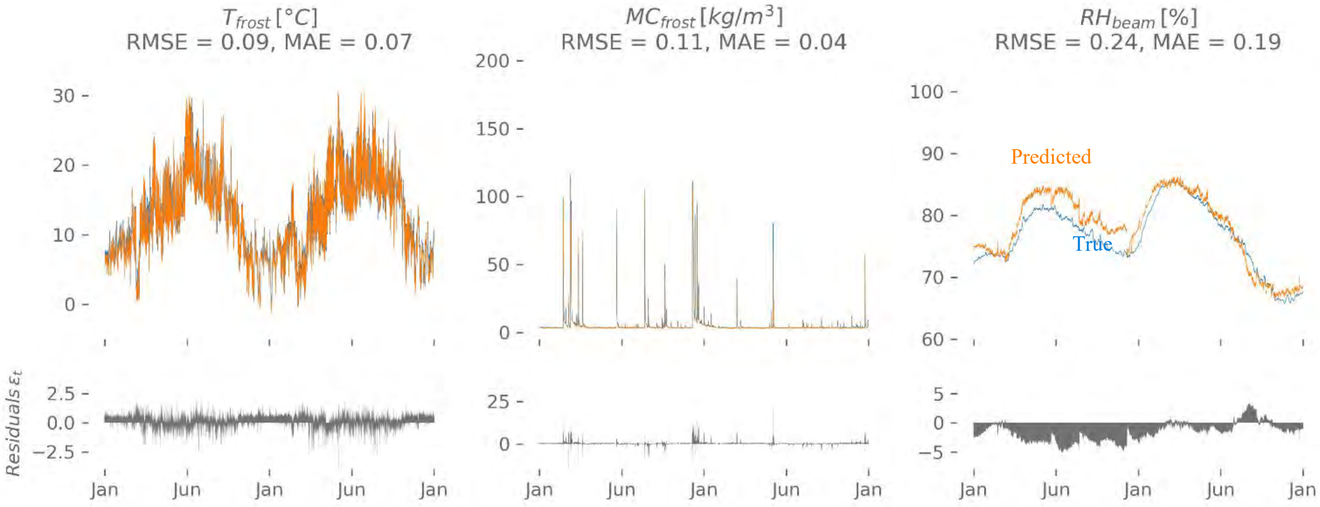


Figure 50: The true output (blue), the output as predicted by the LSTM model (orange) and the residuals ε_t (grey).

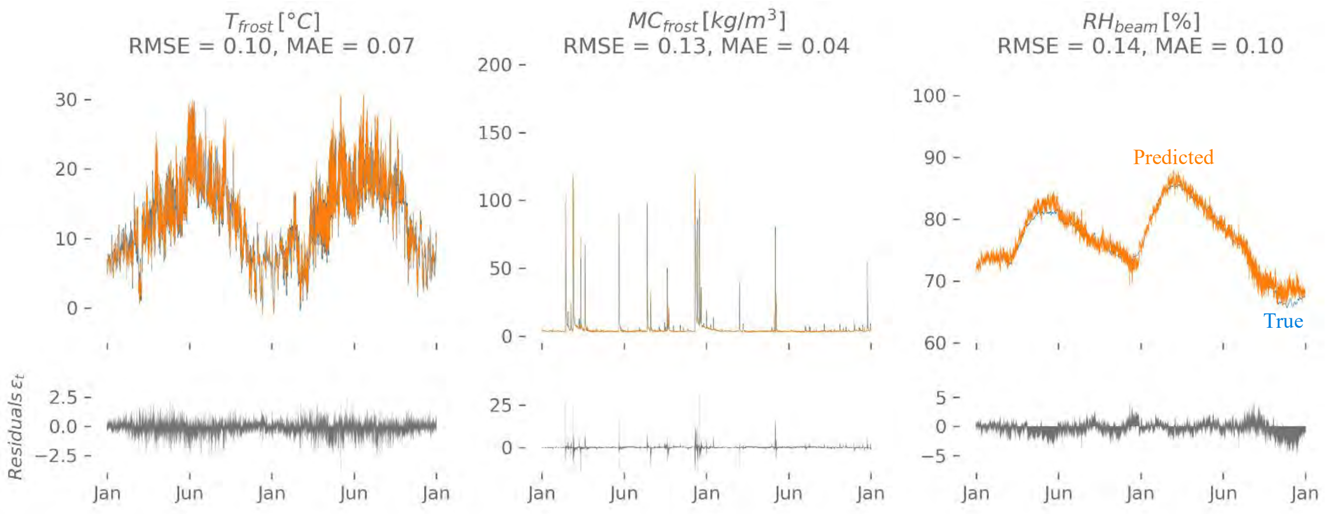


Figure 51: The true output (blue), the output as predicted by the CNN model (orange) and the residuals ε_t (grey).

4.5.5 Conclusions

In this study, fast metamodels are used to replace heavy hygrothermal simulation models in probabilistic assessments of a building component. As the hygrothermal response of a building component is highly non-linear and transient, we opt for neural networks for time series as metamodel, as these have proven successful in many other fields. Three popular types of neural networks are considered: the multilayer perceptron (MLP), recurrent neural networks (RNN), of which the long-short term memory network (LSTM) and gated recurrent unit network (GRU) are included, and the convolutional neural network (CNN). The networks are trained to predict the hygrothermal time series such as the temperature, relative humidity and moisture content at certain positions in a masonry wall, based on the outdoor and indoor climate time series. The MLP has no memory and can only access the current input time step. The RNN's and the CNN both have a memory mechanism, though it works differently for both network types: the RNNs carry information from past input and output time steps through the next one, while the CNN has direct access to multiple past input time steps at once but not to past output time steps. It was found that all network types were capable to predict the temperature accurately. Since the temperature responds almost immediately to a change in boundary conditions, not much memory is needed to capture these patterns. By contrast, only the RNNs and the CNN were able to capture the long-term dependencies needed to predict the relative humidity and moisture content, as these respond much slower to a change in boundary conditions. The CNN performed significantly better at predicting the relative humidity at the wooden beam ends, compared to the RNNs, due to the difference in memory mechanism. Additionally, the CNN is a factor 10 faster to train compared to the RNNs, because its architecture allows processing multiple time steps simultaneously, whereas this happens sequentially in case of the RNNs. To conclude, the CNN appears most suited for the prediction of highly non-linear hygrothermal time series and shows promise to apply as metamodel in more complex probabilistic assessments. Further research is needed to see how the CNN performs when the probabilistic aspect is taken fully into account, i.e. when also scalar and categorical input parameters are included.

4.6 Application on a fully probabilistic case study

4.6.1 Case study

The calculation object in this study is again a 1D cross section of a massive masonry wall. The masonry wall is simplified to a single isotropic brick layer; no mortar joints are modelled. Note also that no construction details such as corners or embedded wooden beam ends are modelled. For a 1D wall, the characteristics and boundary conditions that are expected to influence the hygrothermal performance significantly are considered probabilistic and are shown in Table 7. To deal with variability in climatic conditions, future climate data of four Belgian cities (ClimateforCulture, 2014) and different years of the data are used. To incorporate variability in the wall conditions, uniform distributions of the wall orientation, the solar absorption and the exposure to wind-driven rain are considered. The wind-driven rain load is calculated by using the catch ratio as described in (Blocken, 2002). The catch ratio relates the wind-driven rain (WDR) intensity on a building to the unobstructed horizontal rainfall intensity and is function of the reference wind speed and the horizontal rainfall intensity for a given position on the building façade and wind direction. In this model, variability in wall position and potential shelters, trees or surrounding buildings are reckoned with by the exposure factor. Additionally, the transiency and variation of the wind speed is taken into account via the convective heat transfer coefficient $h_c = h_0 + k_s \cdot v_{wind}^{k_e}$ (EN ISO 06946) where $h_0 = 4$ and $k_e = 1$. The exterior moisture transfer coefficient is related to the exterior heat transfer coefficient through the Lewis relation. The properties of the brick wall itself are subjected to uncertainty as well. Therefore, a uniform distribution of the wall thickness is considered. Furthermore, as the brick material properties vary widely as well, different brick types are included. The characteristics of the currently used brick types can be found in Table 10. Finally, the interior conditions are calculated according to EN 15026 and variability in building use is taken into account by using two different humidity loads.

The remaining boundary conditions are all variables either with small variations or of less importance for the current study of a 1D wall. Therefore, these boundary conditions are treated deterministically. An overview of the deterministic boundary conditions is given in Table 4.3.

Table 9: Probabilistic input parameters and distributions

Parameter	Value
Exterior climate	D(Gent; Gaasbeek; Oostende, StHubert)
Exterior climate start year	D(2020; 2047)
Wall orientation [degrees from North]	U(0; 360)
Solar absorption [-]	U(0.4; 0.8)
Ext. heat transfer coefficient slope k_s [J/m^{3}K]	U(1; 8)
WDR exposure factor [-]	U(0; 2)
Brick wall thickness [m]	U(0.2; 0.5)
Brick material	D(Brick 1; Brick 2; Brick 3)
Interior climate	D(load A; load B) cfr. EN15026

U(a, b): uniform distribution between a and b
D(a, b): discrete distribution between a and b

Table 10: Brick type characteristics

Parameter	Brick 1	Brick 2	Brick 3
Dry thermal conductivity [W/m²K]	0.87	0.52	1.00
Dry vapour resistance factor [-]	139.52	13.25	19.00
Capillary absorption coefficient [kg/m²s$^{0.5}$]	0.046	0.357	0.100
Capillary moisture content [m³/m³]	0.128	0.266	0.150
Saturation moisture content [m³/m³]	0.240	0.367	0.250

Table 11: Discrete input parameters

Parameter	Value
Exterior surface	
Long wave emissivity	0.9
Interior surface	
Total heat transfer coefficient h [W/m²K]	8
Moisture transfer coefficient β [s/m]	3 x 10^{-8}
Initial conditions	
Initial temperature [°C]	20
Initial relative humidity [%]	50

In Deliverable 4.1, the hygrothermal performance of a masonry wall was evaluated by looking at frost damage at the exterior surface, decay of embedded wooden beam ends and mould growth on the interior surface. The latter is especially important in case of thermal bridges, and thus of less importance in 1D simulations and excluded in this study. Table 12 gives an overview of the previously used damage prediction models and the required hygrothermal time series to evaluate them. The risk on frost damage is evaluated based on the number of moist freeze-thaw cycles per year at 0.5 cm from the exterior surface. A 'moist' freeze-thaw cycle is a freeze-thaw cycle that occurs in combination with a moisture content (in this case at 0.5 cm from the exterior surface) that is high enough to induce frost damage. In this study, a moisture content higher than 25% of the saturated moisture content is assumed to be critical and thus to entail a risk on frost damage. Note that this is a rather arbitrary value, as currently no precise prediction criteria is at hand. An indication of the decay risk of wooden beam ends can be made using the VTT wood decay model, which calculates the percentage of mass loss of the wooden beam end based on the temperature and relative humidity (Viitanen et al., 2010). Alternatively, the VTT mould growth model can be used, as mould growth can be a sign that wood root might develop next. The VTT mould growth model calculates the Mould Index based on the fluctuation of the temperature and relative humidity (Ojanen et al., 2010). The Mould Index is a value between 0 and 6, going from no growth to heavy and tight mould growth. In the updated VTT model, the expected material sensitivity to mould growth is implemented as well. In the current studies, the materials are always assumed to belong to the class ‘very sensitive’, hence the obtained results will be worst case. Note, however, that in a 1D wall study solely a rough indication of the wood decay risk is acquired, as two and three dimensional heat and moisture transport as well as potential air rotations around the wooden beam end are neglected.

Table 12: Damage prediction models and required Delphin output

Damage pattern	**Prediction model**	**Required hygrothermal time series**
Frost damage	Moist freeze-thaw cycles	T, RH, saturation degree
Decay of wooden beam ends	VTT wood decay model	T, RH
Mould growth	Updated VTT mould growth model	T, RH

To perform a probabilistic evaluation of the massive masonry wall, the input parameters described above are sampled multiple times, using a Sobol sampling scheme. In this study, 960 samples are used, of which 768 are used for training and 192 for testing. The deterministic model is simulated once for each sampled input parameter combination, using the hygrothermal simulation environment Delphin 5.8.

4.6.2 Network architecture

The architecture of the CNN network used in this paper, shown in Figure 52, was based on the Wavenet architecture (van den Oord, 2016). Its structure is the same as decribed in section 4.3.3.2, but a modification is made to include the scalar input parameters (wall orientation, brick characteristics). In each residual block, the sequence inputs are conditioned on these scalar input parameters. This allows producing output patterns with different characteristics.

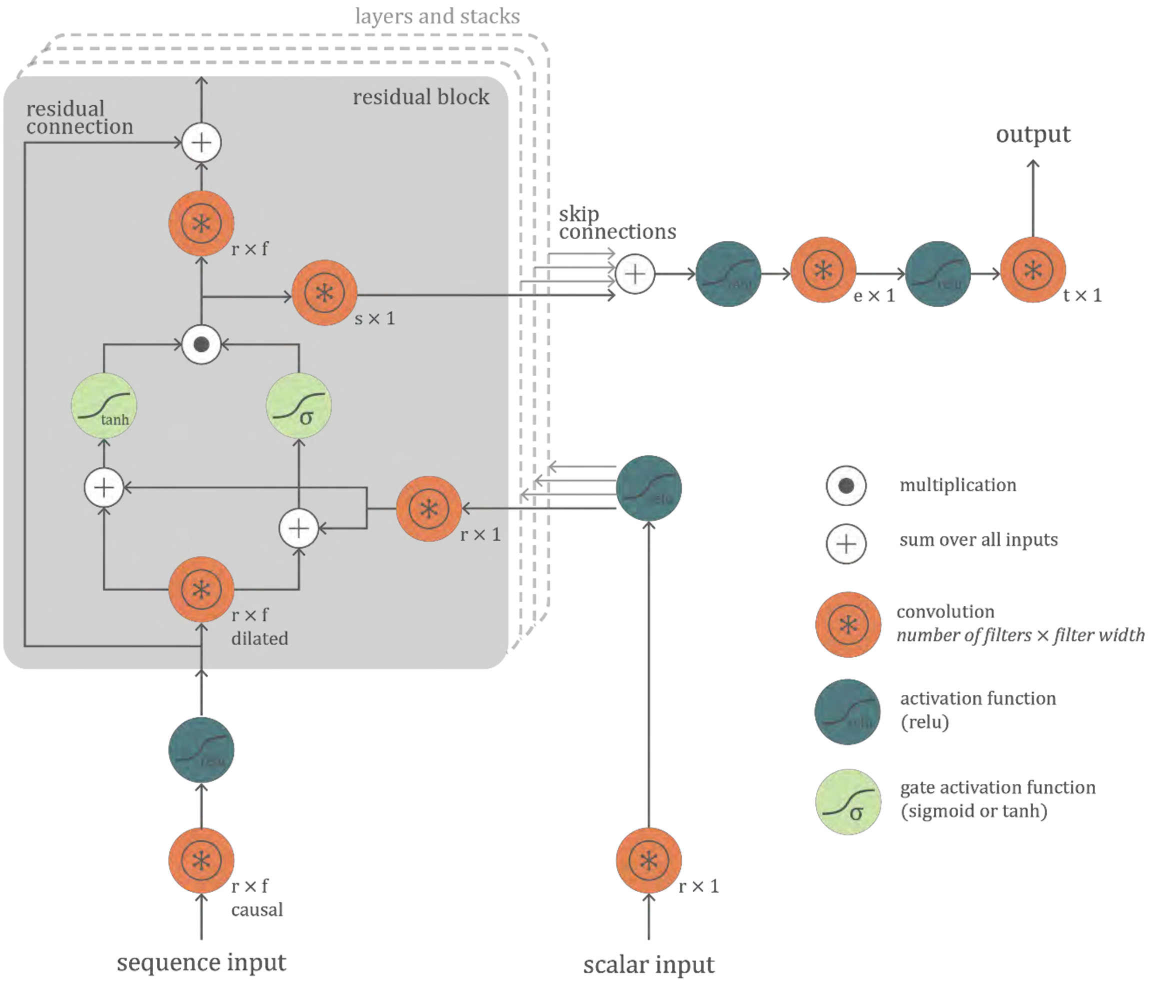


Figure 52: The used CNN architecture with residual blocks and skip connections, based on the Wavenet architecture.

The network is trained to predict the hygrothermal time series given in Table 12, based on the input in Table 9, Table 7 and Table 10. Note that, instead of the moisture content, the saturation degree is now learned, as this was found to result in better predicting performance. The inputs are pre-processed to facilitate learning. The internal and external climate inputs are pre-processed to contain time series only; the categorical parameters start year and interior humidity load are incorporated and cut out, whereas the scalar parameters wall orientation, solar absorption and rain exposure are included in the climate time series but also preserved as scalar input parameter, as they provide extra useful information. Additionally, the different brick types, which is a categorical parameter, are replaced by the characteristics in Table 10, which are scalar parameters. This simplifies the network architecture, and allows more flexibility on using multiple brick types. This results in 6 input time series (exterior temperature, exterior relative humidity, wind-driven rain load, short-wave radiation, interior temperature and interior relative humidity) and 10 scalar inputs (exterior heat transfer coefficient slope, rain exposure factor, solar absorption, wall orientation, brick wall thickness and the 5 brick characteristics from Table 10).

Before presenting the input and output data to the neural network, all data are standardised (zero mean, unit variance). This ensures that all input features are on the same scale, which allows weighting all features equally in their representation. To ensure errors are penalised equally for all targets, the output data are standardised as well. Once the networks are trained on the train dataset, their performance is evaluated on the test dataset, as this gives an indication of the networks' performance when used as a replacement of the original model.

Before the network can be trained, the hyper-parameters need to be set. After a limited number of experiments, it was found that the filter width, number of layers and number of stacks did not really matter, as long as the receptive field is larger than about one year (8760 time steps). Also the number of filters appears of less importance, once larger than 64. The network is trained using the Adam learning algorithm. The learning rate is the amount of change to the network during each step of the learning process. At extremes, a learning rate that is too large will result in too large weight updates and the performance of the network will oscillate over training epochs. A learning rate that is too small may never converge or may get stuck on a suboptimal solution. Experiments showed that a learning rate between 0.0005 and 0.0015 is optimal. In this study, the following hyper-parameter were used:

- Number of filters in skip connections s: 512
- Number of filters in residual connections r: 256
- Number of filters in end connections e: 128
- Filter width f: 12
- Number of layers: 3
- Number of stacks: 1
- Learning rate: 0.001

4.6.3 Performance evaluation

The prediction accuracy of the different metamodels is evaluated using the same three performance indicators as before: the root mean square error (RMSE), the mean absolute error (MAE), and the coefficient of determination (R2), quantified by Equation 4-19. The performance indicators are calculated both for all outputs together as well as for each output separately. Additionally, the accuracy of the post-processed damage predictions is evaluated using the indicators quantified by Equation 4-18, as the damage predictions are not standardized.

4.6.4 Results and discussion

In Table 13, the performance indicators for the network are given. Compared to the performance on the second simple case study (Figure 48), this is an improvement. The total train time of the networks was two hours on a computer with a dual NVIDIA RTX 2070 GPU.

Table 13: Performance indicators on test set

Performance indicator	Value
RMSE	0.079
MAE	0.035
R2	0.994

Figure 53 shows the predictions for the temperature, relative humidity and saturation degree at the wooden beam ends and exterior surface for a sample from the test set. At the top of each graph, the performance indicators for that output is given. Not all outputs are shown, as the previous case studies showed that the other outputs were easy to predict. According to these results, the network is able to predict all outputs accurately.

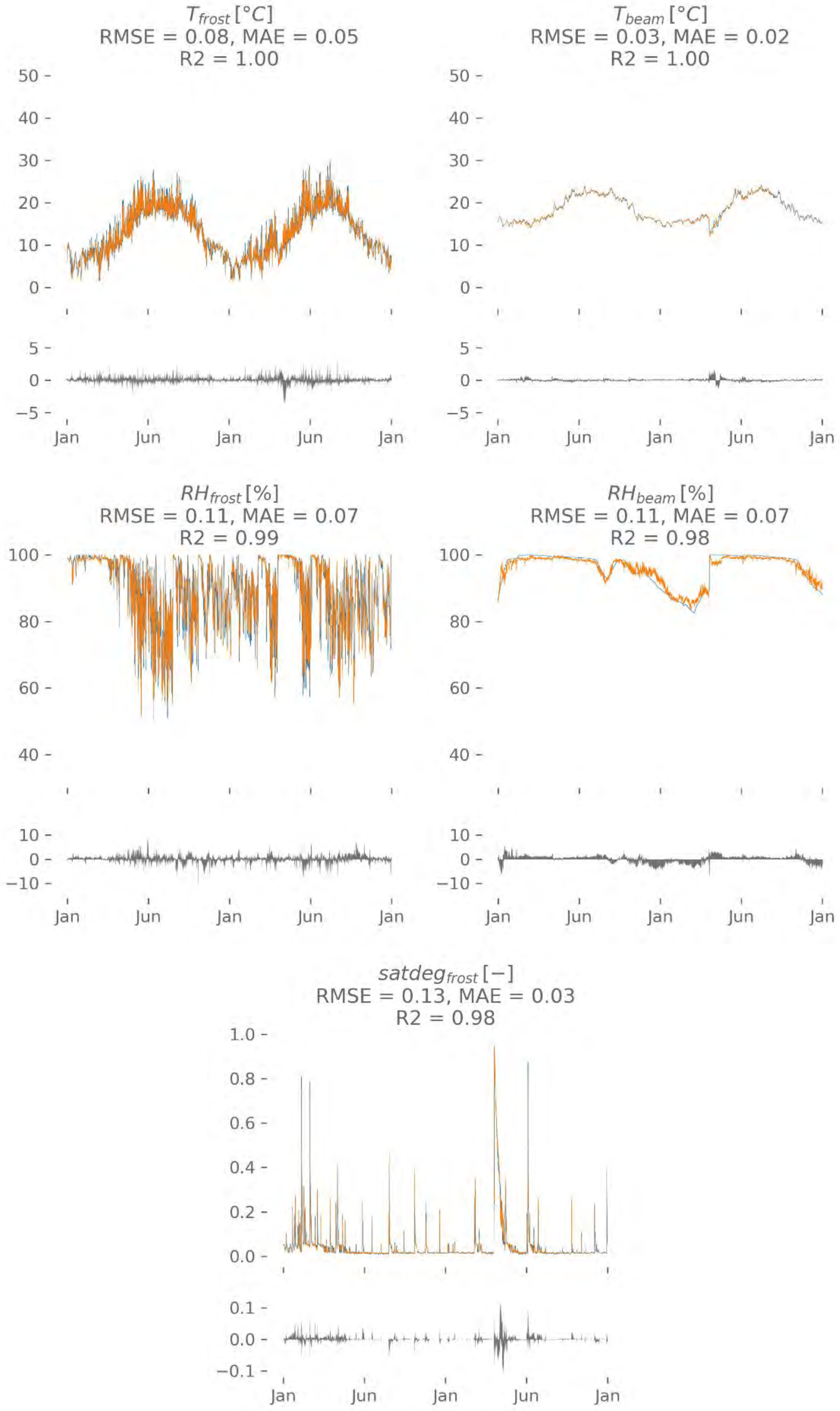


Figure 53: Example of a prediction (orange) of some of the hygrothermal time series, compared to the true output (blue) and the residuals (grey).

Figure 54 shows the damage prediction for the same test set sample, based on the true hygrothermal output (orange) and the predicted hygrothermal output (blue). Again, the performance indicators are given at the top of each graph. This indicates that both the Mould index at the wooden beam ends and the wood mass loss can be predicted quite accurately based on the network prediction. The indicators for frost damage score less well. Figure 55 shows the damage prediction for one of the worst performing samples. The number of moist freeze-thaw cycles is clearly overestimated here. These results should be put in perspective, however, as a difference of two moist freeze-thaw cycles is not much. Because the total number of cycles is low, a deviation of two cycles results in a high error according to the indicators. However, whether 6 or 4 moist freeze-thaw cycles occur, both are likely not high enough to cause frost damage. Thus, although the performance indicators score less well, these results are unlikely to influence decision making significantly. Note that these are the results for Belgian climates. In more severe climates, where a larger number of moist freeze-thaw cycles occurs, different results might be obtained. This has to be tested still.

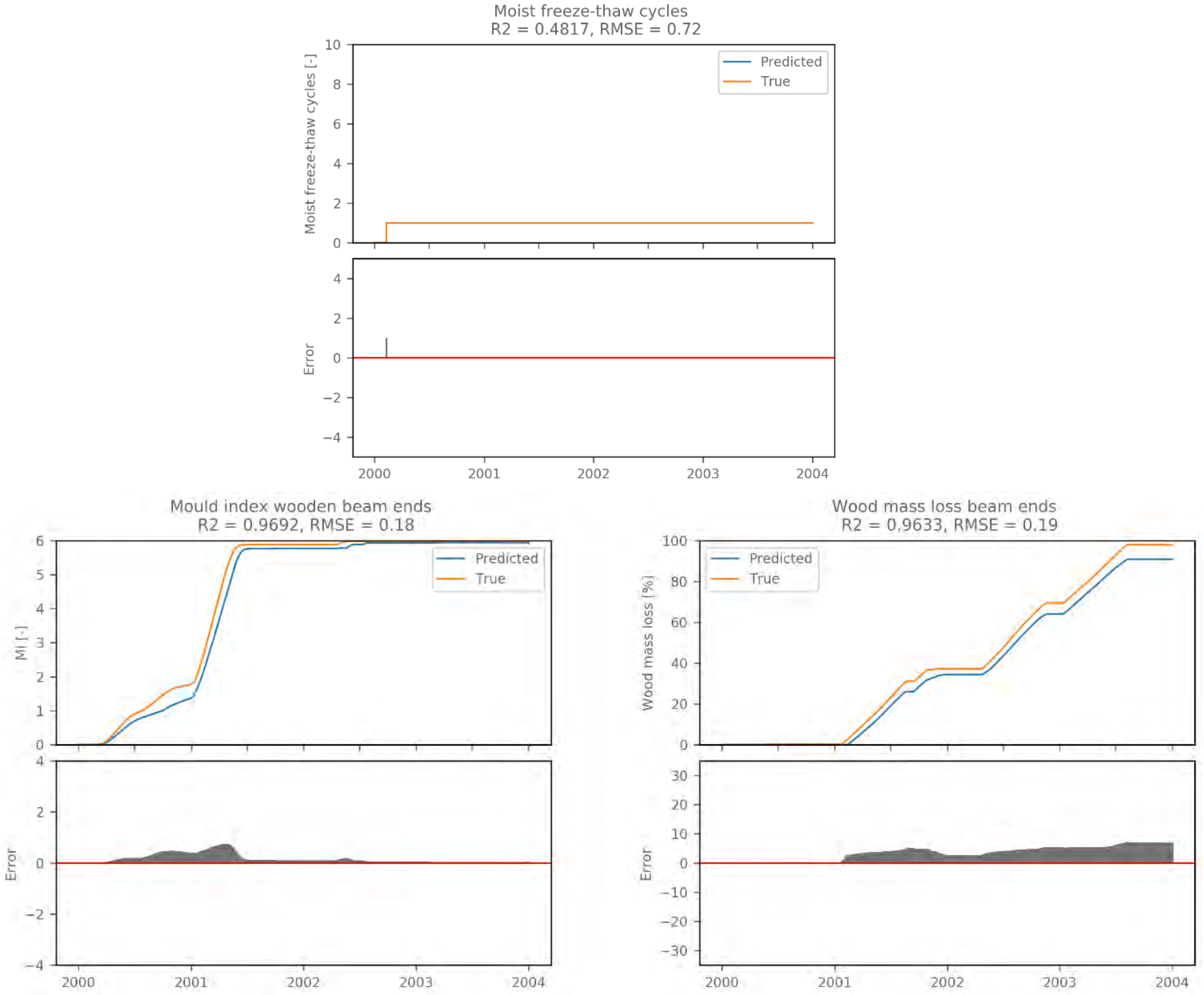


Figure 54: Damage prediction for the same sample used in Figure 53.

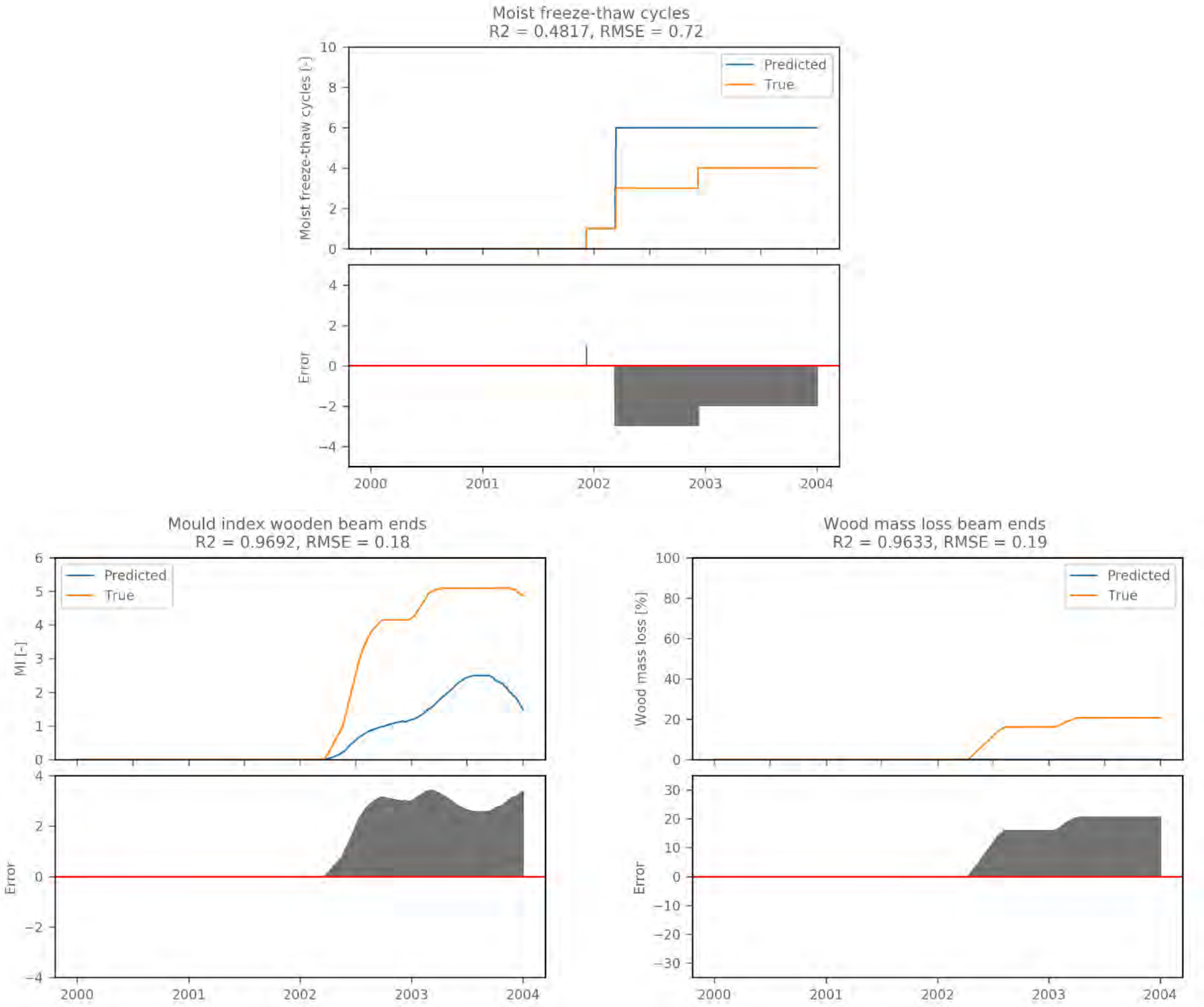


Figure 55: Damage prediction for one of the worst performing samples.

In previous reports, the damage risk was assessed by looking at the total number of moist freeze-thaw cycles, the maximum mould index and the total wood mass loss after a certain period. For the current case study, these risk indicators are plotted in Figure 56, both for the true output (x-axis) and the network output (y-axis). This shows that most predictions are accurate but deviations occur in some cases. The mould index and wood mass loss at the wooden beam ends tends to be underestimated by the network predictions. In these cases, the fluctuations of the predicted relative humidity influence the damage models activation process and growth/decay rate negatively, thus resulting in an underestimation compared to the true output.

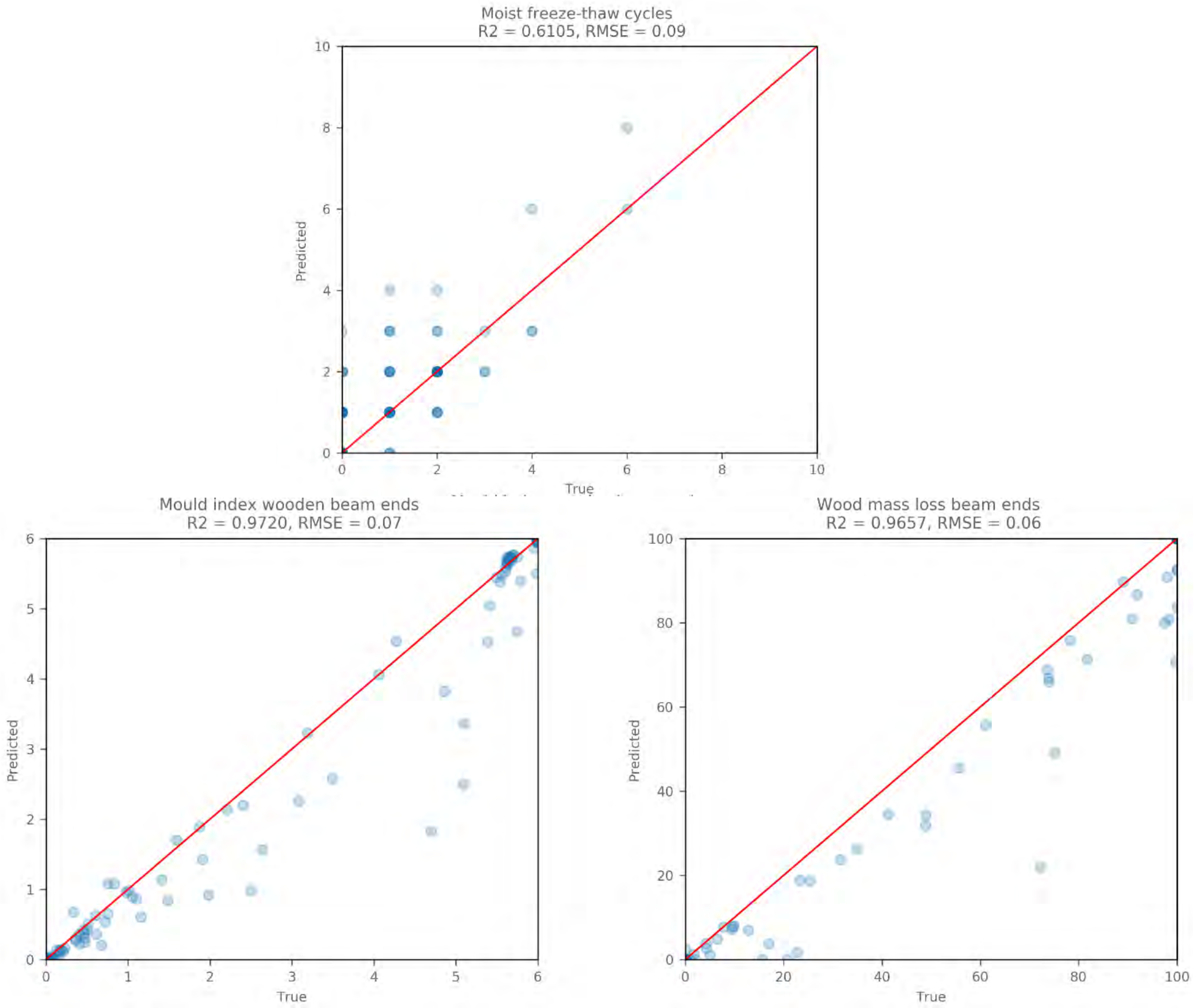


Figure 56: Comparison of the risk indicators predicted using the network output (y-axis) and by the true output (x-axis).

4.6.5 Conclusions

In this study, a convolutional neural network (CNN) is trained to predict the hygrothermal behaviour of a 1D masonry wall, taking into account all probabilistic input parameters. The network is trained to predict hygrothermal time series (temperature, relative humidity and saturation degree), which are subsequently used to predict damage using existing damage prediction models (moist freeze-thaw cycles, mould growth, wood decay). The results showed that the CNN is able to predict the hygrothermal outputs accurately. The damage predictions, based on the hygrothermal predictions by the network, are accurate in most cases as well. However, in some cases, the mould index and wood mass loss at the wooden beam ends are underestimated, due to the fluctuations of the predicted relative humidity which influence the mathematical damage models activation process and growth/decay rate negatively. The indicators for frost damage score low, due to relatively high errors. Because the total number of cycles is low, a deviation of a few cycles results in a high error according to the indicators. However, the total number of moist freeze-thaw cycles is low and likely not high enough to cause frost damage. Thus, although the performance indicators score less well, these results are unlikely to influence decision making significantly.

4.7 References

Bai S., Kolter J.Z., Koltun V. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling, http://arxiv.org/abs/1803.01271.

Bakker B. 2002. Reinforcement learning with long short-term memory. In Advances in Neural Information Processing Systems, 14.

Bengio Y., Simard P., Frasconi P. 1994. Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks, 5:157-166.

Bengio Y., LeCun Y. 2007. Scaling learning algorithms towards AI. In Bottou L., Chapelle O., DeCoste D., and Weston J., editors, Large-Scale Kernel Machines. MIT Press.

Blocken, B.; Carmeliet, J. 2002. Spatial and temporal distribution of driving rain on a low-rise building, Wind and Structures, 5:441–462.

Borovykh A., Bohte S., Oosterlee C.W. 2017. Conditional Time Series Forecasting with Convolutional Neural Networks, http://arxiv.org/abs/1703.04691.

Cottrell M., Girard B., Girard Y., Mangeas M., Muller, C. 1995. Neural modeling for time series: A statistical stepwise method for weight elimination. IEEE Transactions on Neural Networks, 6:1355-1364

Chauvin Y. (Ed.), Rumelhart D.E. (Ed.). 1995. Backpropagation: Theory, architectures, and applications. New York: Psychology Press.

Cho K., van Merrienboer B., Gulcehre C., Bahdanau D., Bougares F., Schwenk H., Bengio Y. 2014a. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv preprint arXiv:1406.1078v3.

Cho K., Van Merrienboer B., Bahdanau D., Bengio Y. 2014b. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.

Duchi J., Hazan E., Singer Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research, 12:2121–2159.

Elman J.L. 1990. Finding structure in time. Cognitive Science, 14: 79-211.

Eck D., Schmidhuber J. 2002. Finding temporal structure in music: blues improvisation with LSTM recurrent networks. In H. Bourlard, editor, Neural Networks for Signal Processing XII, Proceedings of the 2002 IEEE Workshop, New York, USA.

European Commission, Climate for Culture: damage risk assessment, economic impact and mitigation strategies for sustainable preservation of cultural heritage in times of climate change, 2014.

Fischer T., Krauss C. 2018. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270:654-669.

Gers F.A., Schmidhuber J., Cummins F. 2000. Learning to forget: Continual prediction with LSTM. Neural computation, 12:2451-2471.

Gers F.A. 2001. Long short-term memory in recurrent neural networks. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.

Graves A. and Schmidhuber J. 2005. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. Neural Networks, 18:602-610.

Graves A., Fernandez S., Gomez F., Schmidhuber J. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In Proceedings of the International Conference on Machine Learning (ICML 2006), Pittsburgh, USA.

Hinton G.E., Osindero S., Teh Y.W. 2006. A Fast Learning Algorithm for Deep Belief Nets. Neural Computation, 18:1527-1554.

Hinton G.E., Srivastava N., Swersky K. 2012. Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Hochreiter S., Schmidhuber J. 1997. Long short-term memory. Neural Computation, 9:1735-1780.

Jaeger H. 2001. The “Echo State” Approach to Analysing and Training Recurrent Neural Networks. Technical Report GMD Report 148, German National Research Center for Information Technology.

Janssen H., Roels S., Van Gelder L. Das P. 2015. Annex 55 - Reliability of Energy Efficient Building Retrofitting, Probability Assessment of Performance and Cost. Final Report. Chalmers University Of Technology Gothenburg, Sweden.

Jordan M.I. 1990. Attractor dynamics and parallelism in a connectionist sequential machine. IEEE Press.

Jung M., Lee H., Tani J. 2018. Adaptive detrending to accelerate convolutional gated recurrent unit training for contextual video recognition. Neural Networks, 105: 56-370

Kemajou A., Mba L., Meukam P. 2012. Application of artificial neural network for predicting the indoor air temperature in modern building in humid region, British Journal of Applied Science and Technology, 2:23-34.

Kingma D.P., Ba L.J. 2015. Adam: A Method for Stochastic Optimization. In: Proceedings of International Conference on Learning Representations. arXiv preprint arXiv:1412.6980

Lang K.J., Waibel A.H., Hinton G.E. 1990. A Time-delay Neural Network Architecture for Isolated Word Recognition. Neural Networks, 3:23-43.

McCulloch W.S., Pitts W. 1988. A Logical Calculus of the Ideas Immanent in Nervous Activity, pages 15-27. MIT Press.

Ojanen T., Viitanen H., Peuhkuri R., Lähdesmäki K., Vinha J., Salminen K. 2010. Mold Growth Modeling of Building Structures Using Sensitivity Classes of Materials. In Proceedings of the Thermal Performance of the Exterior Envelopes of Buildings XI; Clearwater Beach, Florida: 1–10.

Pascanu R., Mikolov T., Bengio Y. 2013. On the difficulty of training Recurrent Neural Networks. arXiv preprint arXiv:1211.5063

Ravanelli M., Brakel P., Omologo M., Bengio Y. 2017. Improving speech recognition by revising gated recurrent units. arXiv preprint arXiv:1710.0064

Rosenblatt F. 1963. Principles of Neurodynamics. Spartan, New York.

Rumelhart D.E., Hinton G.E., Williams R.J. 1986. Learning internal representations by error propagation. MIT Press: 318-362.

Shen G., Tan Q., Zhang H., Zeng P., Xu J. 2018. Deep learning with gated recurrent unit networks for financial sequence predictions. Procedia Computer Science, 131: 895-903

Soleimani-Mohseni M., Thomas B., Fahlén P. 2006. Estimation of operative temperature in buildings using artificial neural networks. Energy and Buildings, 38:365-640.

Sundermeyer M., Schluter R., Ney H. 2012. LSTM neural networks for language modeling. In Proceedings of the ISCA's 13th Annual Conference (INTERSPEECH 1012), September 9-13 2012, Portland, USA.

Sutskever I., Martens J., Hinton G.E. 2011. Generating text with recurrent neural networks. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), June 28 - July 2 2011, Washington, USA.

Sutskever I., Martens J., George D., Hinton G.E. 2013. On the importance of initialization and momentum in deep learning. In Proceedings of the 30th International Conference on Machine Learning (ICML-13), June 16 - 21 2013, Atlanta, USA.

Van den Oord A., Dieleman S., Zen H., Simonyan K., Vinyals O., Graves A., Kalchbrenner N., Senior A., Kavukcuoglu K. 2016. WaveNet: A Generative Model for Raw Audio, doi:10.1109/ICASSP.2009.4960364.

Van Gelder L., Das P., Janssen H. Roels S. 2014. Comparative study of metamodelling techniques in building energy simulation: Guidelines for practitioners. Simulation Modelling Practice and Theory, 49:245-257.

Vereecken E., Van Gelder L., Janssen H., Roels S. 2015. Interior insulation for wall retrofitting - A probabilistic analysis of energy savings and hygrothermal risks, Energy and Buildings, 89:231-244.

Viitanen H. 2011. Moisture and Bio-Deterioration Risk of Building Materials and Structures. In Mass Transfer - Advanced Aspects.

Viitanen H., Toratti T., Makkonen L., Peuhkuri R., Ojanen T., Ruokolainen L.; Räisänen J. 2010. Towards modelling of decay risk of wooden materials. European Journal of Wood and Wood Products, 68, 303–313.

Werbos P.J. 1990. Backpropagation through time: what it does and how to do it. Proceedings of the IEEE, 78:1550-1560.

Williams R.J., Zipser D. 1989. A learning algorithm for continually running fully recurrent neural networks. Neural Computation, 1:270-280.

Li X., Peng L., Yao X., Cui S., Hu Y., You C., Chi T. 2017, Long short-term memory neural network for air pollutant concentration predictions: Method development and evaluation. Environmental Pollution, 231:997-1004.

Zeiler M.D. 2012. Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

5 Summary and conclusions

This report presents the second of two deliverables related to RIBuild's WP 4 '*Probabilistic assessment of interior insulation solutions*'. The key objective of WP4 is the development of an efficient strategy for the probabilistic hygrothermal assessment of interior insulation solutions. That strategy involves Monte-Carlo-based repetitions of hygrothermal simulations, and hence requires an efficient deterministic hygrothermal simulator and an efficient probabilistic assessment approach. Prior to the initiation of the RIBuild project, both a deterministic simulator and a probabilistic methodology were already available. Their joint application to interior insulation solutions does however require further developments, primarily in relation to their numerical efficiency.

The various tasks in WP4 hence focus on efficiency improvements of both the deterministic simulator and the probabilistic methodology. Tasks 4.1 and 4.2 target the efficient one-, two- and three-dimensional simulation of heat, air and moisture transfer in building components, Tasks 4.3 and 4.4 aim for the efficient probabilistic assessment of hygrothermal performances based on these simulations. In the final Task 4.5, the developments of Tasks 4.1 to 4.4 are to be brought together in a final application example. The first deliverable reported on the general developments in WP4 and on the specific results from Task 4.1 and Task 4.3. This deliverable reports the findings from the specific activities on model-order reduction in hygrothermal simulations in Task 4.2 (Section 2) and on neural-network-based metamodelling in Task 4.4 (Section 3).

Section 2 reports on the investigation on the performance of two model order reduction methods, Proper Orthogonal Decomposition (POD) and Proper Generalised Decomposition (PGD), with linear and nonlinear calculation objects. It is demonstrated there that the PGD method does not perform well, and the method did hence not get further consideration. The POD method, contrarily, shows to perform very well for the linear thermal case, as only a limited number of construction modes and a small amount of snapshots are required to obtain accurate results. This method furthermore does well when extrapolated to different problems and when applied in a probabilistic simulation study. For the nonlinear hygrothermal case, the accuracy and performance of POD appears to strongly depend on the degree of nonlinearity of the calculation object. Relative to the linear cases, (much) more construction modes and snapshots are required in order to obtain an accurate result. Since the inefficiency of POD for nonlinear simulations arises from the high computational cost of repeatedly evaluating the nonlinear terms, the Discrete Empirical Interpolation method is studied to further decrease the computational cost of the hygrothermal simulation. This approach again brings a significant reduction of the computational complexity.

Section 3 describes the examination of the performance of neural networks for metamodeling of hygrothermal behaviour. To that aim, a convolutional neural network (CNN) is trained to predict the behaviour of a 1D masonry wall, taking into account all probabilistic input parameters. The network is trained to predict hygrothermal time series (temperature, relative humidity and saturation degree), which are subsequently used to predict damage using existing damage prediction models (moist freeze-thaw cycles, mould growth, wood decay). The results show that the CNN is able to predict the hygrothermal outputs accurately. The damage predictions, based on the hygrothermal predictions by the network, are accurate in most cases as well. However, in some cases, the mould index and wood mass loss at the wooden beam ends are underestimated, and the indicators for frost damage score low, due to relatively high errors. But, although the performance indicators score less well, these deviations are unlikely to influence decision making significantly.