# Highlights

**Validation of an integrated data-driven surrogate model and a thermo-hydraulic network based model to determine boiler operational loads using a fully connected mixture density network**

B.T. Rawlins,Ryno Laubscher,Pieter Rousseau

- Computational Fluid Dynamics capture trend of boiler heat uptake at reduced loads.

- Computational Fluid Dynamics capture trend of boiler heat uptake at reduced loads.

- Computational Fluid Dynamics capture trend of boiler heat uptake at reduced loads.

# Validation of an integrated data-driven surrogate model and a thermo-hydraulic network based model to determine boiler operational loads using a fully connected mixture density network

B.T. Rawlins[a,*], Ryno Laubscher[b] and Pieter Rousseau[a]

[a]*Department of Mechanical Engineering, Applied Thermal-Fluid Process Modelling Research Unit, University of Cape Town, Library Road, Rondebosch, Cape Town, 7701, South Africa*
[b]*Department of Mechanical Engineering, Stellenbosch University, Banghoek Road, Stellenbosch, 7600, South Africa*

ARTICLE INFO

*Keywords*:
Mixture density model
Steady State

ABSTRACT

ss fhvosjv
sfjvnsfklnvjksfn
skfnvkpsfn
sjkvnsf skfnv
ksfvn;sfv,

## 1. Introduction

## 2. Applicable machine learning theory

The present work makes use of various machine learning architectures to develop a surrogate model that can predict the thermal and combustion characteristics of a utility scale boiler using high level inputs. The current section discusses the details of the respective architectures, namely linear regression, ANN and ANN-MDN models.

### 2.1. Linear regression

The present work makes use of a multiple linear regression model as a base model for comparative purposes. The main assumption of a multiple linear regression model is that the output/s can be calculated from a linear combination of the variable inputs. In other words a linear regression models aims to determine the quantitative relationship between the dependent and independent variables [12]. The representation of the $i$-th dependent variable ($y_i$) can be written as follows for $m$-number of independent variables ($x_{mi}$):

$$y_i = \beta_0 + \beta_1 x_{1i} + ... + \beta_m x_{mi}$$
$$i = 1, 2, 3...n \tag{1}$$

where $\beta_0$ is a constant term, $\beta_m$ is the $m$-th coefficient, and $n$ is the total number of observations.

The optimal solution can be estimated by minimizing the cost function ($J$). A cost function usually calculates the difference between the estimated and the desired values and is reported as a single number. Multiple linear regression problems namely utilize the mean square error (MSE) between the desired and estimated ($\hat{y}_i$) values [5], which is

given in Equation 2.

$$J_{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{2}$$

To minimize the cost function, the gradient descent algorithm is the preferred method [12], which is an iterative procedure used to find the local minimum/maximum. Considering the cost function as a function of the weight the algorithm can be written as shown in Equation 3.

$$\beta_m = \beta_m - \eta \frac{\partial}{\partial \beta_m} J(\beta_m) \tag{3}$$

where $\eta$ is known as the learning rate.

In most cases the relationship between the dependent and independent variables are not always linear. Special linear basis models, such as polynomial, sinusoidal and radial can be used to optimize the training results [8]. For the current work a multiple linear progression model is utilized as a base model for comparison.

### 2.2. Multilayer perception networks (MLPs)

Artificial neural networks (ANN) are machine learning systems inspired by biological animal brains [8]. There are many classifications of ANNs, with multilayer perception networks (MLP) being the standard representation [4]. Typically MLPs are adapted for supervised learning problems where the input variables are mapped to labelled output variables. The relationship between the input and output variables are learned by optimizing the weights ($\overline{w}$) and biases ($\overline{b}$) to minimize a selected cost function, which for most cases is the MSE given in equation 2. A standard MLP schematic is given in figure 1, illustrating the common features of a MLP, that being the input, hidden and output layers.

To calculate the output values ($\hat{y}_i$) the forward propagation algorithm is utilized, which calculates the output for

*Corresponding author
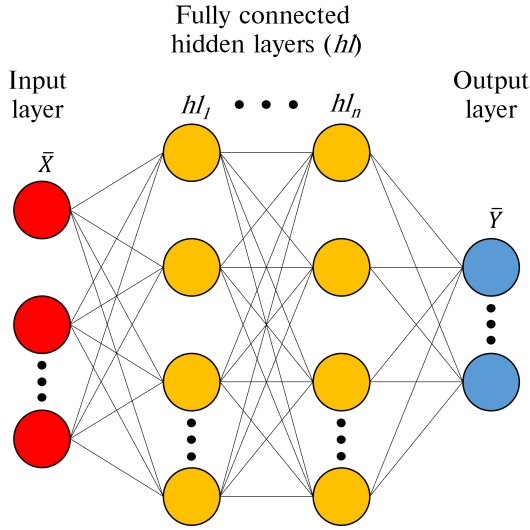✉ rwlbra001@myuct.ac.za (B.T. Rawlins)
ORCID(s):

Figure 1: Traditional MLP schematic

each layer and moves sequentially through the network till an output is determined. Each network layer output is calculated using two steps, the first being the calculation of the summed signal $\overline{z}_l$ and secondly the use of an activation function to generate the output signal $\overline{h}_l$. Equation 4 highlights the first step, where $\overline{h}_{l-1}$ is the output signal from the previous layer.

$$\overline{z}_l = \overline{h}_{l-1} \cdot \overline{w}_l + \overline{b}_l \tag{4}$$

The result of the equation 4 is subsequently passed to an activation function ($\overline{h}_l = \sigma_l(\overline{z}_l)$). There are various activation functions that can be utilized, such as linear, ReLu, Elu, and the hyperbolic-tangent [4], with the final layer activation function usually being linear for regression models. The current work makes use of ReLu and linear activation functions for the hidden layers. The ReLu and linear activation functions are shown in Equation 5.

$$\overline{h}_l = \sigma_{ReLu}(\overline{z}_l) = \overline{z}_l = \begin{cases} \overline{h}_{l-1} \cdot \overline{w}_l + \overline{b}_l & if \;\; \overline{z}_l > 0 \\ 0 & if \;\; \overline{z}_l < 0 \end{cases}$$
$$\overline{h}_l = \sigma_{linear}(\overline{z}_l) = \overline{z}_l = \overline{h}_{l-1} \cdot \overline{w}_l + \overline{b}_l \tag{5}$$

When the forward propagation step is complete, the network weights and biases can be updated to minimize the cost function (refer to Equation 2) using the backward propagation method **??**. The methodology calculates the gradient of the cost function with respect to the weights and biases for each layer. Once the gradients have been calculated the weights and biases are updated using the gradient descent algorithm. The current work makes use of the Adam [4] alternative to the gradient descent algorithm and is illustrated in Equation 6. Forward- and backward-propagation algorithms would be iteratively conducted until the cost function is reduced to below a desired threshold.

**Algorithm 1.**

$$\overline{m} \leftarrow \beta_1 \overline{m} + (1 - \beta_1)\nabla_\theta J_{MSE}(\overline{\theta})$$
$$\overline{s} \leftarrow \beta_1 \overline{s} + (1 - \beta_2)\nabla_\theta J_{MSE}(\overline{\theta}) \otimes \;\; \nabla_\theta J_{MSE}(\overline{\theta})$$
$$\overline{m} \leftarrow \frac{\overline{m}}{1 - \beta_1^t} \tag{6}$$
$$\overline{s} \leftarrow \frac{\overline{s}}{1 - \beta_2^t}$$
$$\overline{\theta} \leftarrow \overline{\theta} - \eta\overline{m} \otimes (\sqrt{\overline{s} + \epsilon})^{-1}$$

The variable $\overline{\theta}$, of Equation 6, represents the model parameters weights ($\overline{w}$) and biases ($\overline{b}$) for each layer. Scaling ($\overline{s}$) and momentum ($\overline{m}$) matrices are initialized to zero when beginning the training phase, $t$ is the iterative counter, $\beta_1$ and $\beta_2$ are the momentum and scaling decay hyper-parameters set to values of 0.9 and 0.999 respectively. Lastly $\epsilon$ is a smoothing term set to a value of $10^{-8}$.

## 2.3. Mixture density networks (MDNs)

Mixture density networks are fundamentally built from two components, that being an ANN and a mixture model, this allows for multi-modal predictions. The ANN can be a standard feed-forward MLP or a recurrent neural network (RNN), with RNNs traditionally being used in dynamic applications with at least one feedback loop [7]. For this study the traditionally MLP is used.

MDNs are used to predict the parameters of a probability distribution ($P(\overline{X} \mid \overline{Y})$) allowing for non-Gaussian distributions to be modelled, thus making MDNs a probabilistic machine learning framework. MDNs estimate the conditional probability distribution as a mixture of Gaussian distributions where the mixing coefficients ($\overline{\pi}_k$) and component densities are flexible functions of the input data ($\overline{X}$). Equation 7 illustrates the conditional probability function.

$$P(\overline{X} \mid \overline{Y}) = \sum_{k=1}^{K} \pi_k(\overline{X}) \cdot \mathbb{N}(\overline{Y} \mid \overline{\mu}_k(\overline{X}), \overline{\sigma}_k^2(\overline{X})) \tag{7}$$

where $K$ represents the number of selected normal distributions, while $\overline{\mu}_k$ and $\overline{\sigma}_k^2$ are the predicted means and variances for each distribution $k$ given the input data $\overline{X}$ respectively.

A schematic of a simple ANN-MDN network is given in Figure 2, highlighting the mixing coefficients, predicted means and deviations. It can be shown that modifications are made to the output layer by splitting the network output into three parts in order to calculate the $\overline{\pi}_k$, $\overline{\mu}_k$ and $\overline{\sigma}_k^2$ for each $k$ distribution. This enables the MDN network to learn the $P(\overline{X} \mid \overline{Y})$.

Bishop [3] proposed the following restrictions for the mixing coefficients and variance components of the MDN output layer. Since the mixing coefficients contain the discrete probabilities of an output belonging to each $K$ normal
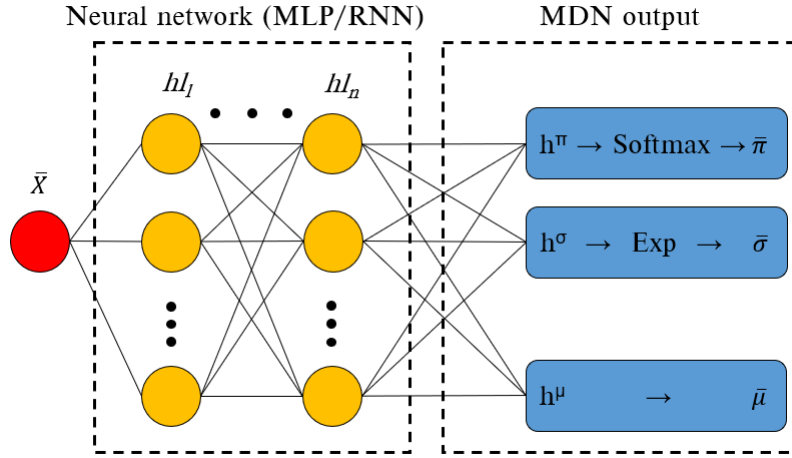
**Figure 2:** Simple MDN network

distribution for all observations, the mixing coefficients must satisfy the constraints listed in Equation 8.

$$\sum_{k=1}^{K} \overline{\pi}_k^n = 1$$
$$0 \le \overline{\pi}_k^n \le 1 \tag{8}$$

These constraints are met by using a Softmax function on the output of the hidden layer $h^{\pi}$. The mixing coefficient is calculated using Equation 9.

$$\overline{\pi}_k^n(\overline{x}^n) = \frac{exp(h_k^{\pi,n})}{\sum_{k=1}^{K} exp(h_k^{\pi,n})} \tag{9}$$

Similarly a constraint is applied to the standard deviation values ensuring a positive value, this is achieved through the use of an exponential function being applied to the standard deviation leg of the MDN output layer, namely $h^{\sigma}$. Equations 10 shows the imposed constraint for the standard deviation leg.

$$(\overline{\sigma}_k^n(\overline{x}^n))^2 \ge 0$$
$$\sigma_k^n(\overline{x}^n) = exp(h_k^{\sigma,n}) \tag{10}$$

The MDN weights and biases are optimized by minimizing the negative log-likelihood for all observations in a batch, this is shown in Equation 11.

$$J_{NLL}(\overline{Y}, \overline{\pi}, \overline{\sigma}, \overline{\mu}) =$$
$$-\sum_{n=1}^{N} ln \left\{ \sum_{k=1}^{K} \overline{\pi}_k(\overline{X}^n, \overline{\theta}) \cdot \mathbb{N}(\overline{Y}^n \mid \overline{\mu}_k(\overline{X}^n, \overline{\theta}), \overline{\sigma}_k^2(\overline{X}^n, \overline{\theta})) \right\} \tag{11}$$

## 3. Data generation

A steady-state multiphase non-thermal equilibrium CFD model was used to generate the target data, which was subsequently used for training/development of an appropriate surrogate model.

### 3.1. CFD model setup

The current study makes use of the commercial CFD software package ANSYS® Fluent 2019 R3 to resolve the fluid flow, heat transfer and combustion processes for a $620MW_e$ utility scale coal fired boiler. The computational domain is modelled on a symmetry plane half way through the depth of the boiler. Figure 3 highlights the computational domain and defines the important boundary conditions.

The general conservation equations, which include, continuity, momentum, energy and species, were solved using a Eulerian approach. The subsequent equations are stated in Equation (12).

$$\frac{\partial}{\partial x_i}(\rho \overline{u}_i) = S$$

$$\frac{\partial}{\partial x_i}(\rho_{eff} u_i u_j) + \frac{\partial \overline{p}}{\partial x_j} =$$

$$\frac{\partial}{\partial x_i}\left[ \mu \left\{ \frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\delta_{ij}\frac{\partial u_i}{\partial x_i} \right\} \right] + \frac{\partial}{\partial x_i}(-\rho \overline{u_i' u_j'}) + S_m$$

$$\frac{\partial}{\partial x_i}(u_i[\rho E + p]) = \frac{\partial}{\partial x_j}\left[ \lambda \frac{\partial T_g}{\partial x_j} \right] + S_h \tag{12}$$

$$\frac{\partial}{\partial x_i}(\rho u_j Y_k) = -\frac{\partial}{\partial x_j}(\vec{J}_k) + \sum_r R_{j,r} + S_k$$

The resolution of the Reynolds stress term found in the momentum equation, $-\rho \overline{u_i' u_j'}$, was approximated using the Boussineq equation [11]. In the present study the realizable k-$\varepsilon$ turbulence model was utilized to address the turbulence closure problem, this model was selected for its applicability
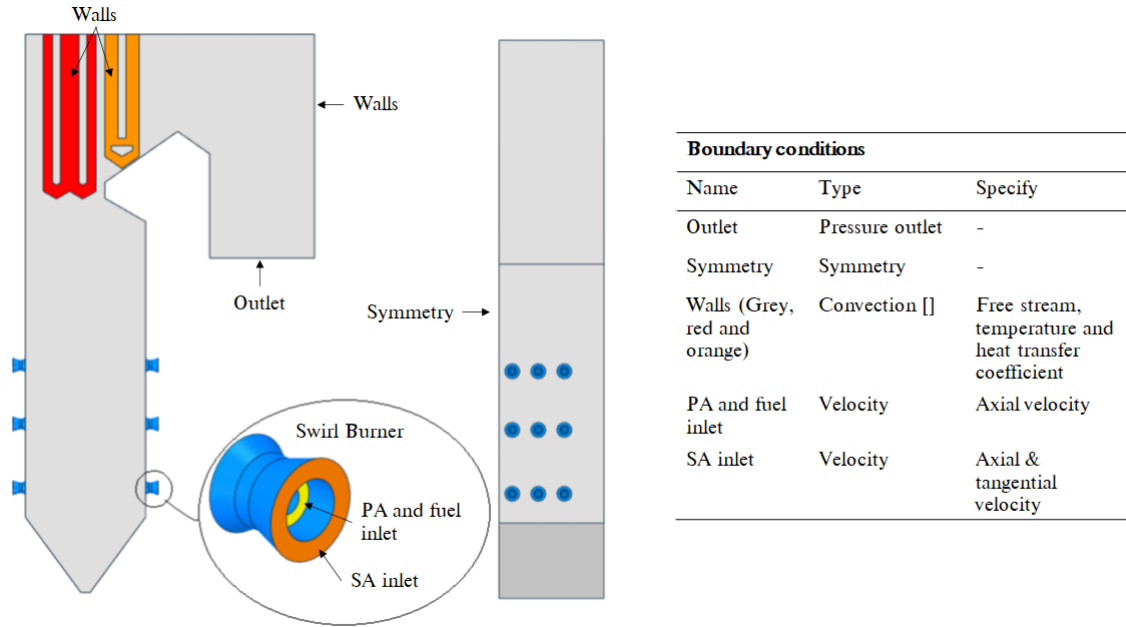
**Figure 3:** CFD model geometry and boundary condition descriptions.

in modelling the effects of coal-fired swirl burners [6].

The P1 radiation model was used to resolve the radiative field in the domain. Particle transport was modelled using a multiphase approach, further details on the approach are provided in the validation study of Rawlins et al [9]. The combustion follows a four step sequential process, beginning with the heating and evaporation of the moisture present in the fuel, followed by the devolatilization process where the volatiles are liberated from the solid particle, succeeded by the phenomena of char burnout, and finally the gas phase reactions would commence. The char oxidation reaction product species was set to that of carbon monoxide ($CO$). For the gas-phase reactions the turbulence-chemistry interaction was approximated using the eddy dissipation model (EDM). A summary of the combustion equations and constants are provided in Table 1.

The simulations were solved using the SIMPLE pressure–velocity coupling scheme. The pressure term was discretized using the PRESTO! scheme. Momentum, species and energy equations were discretized using the second-order upwind scheme and the turbulent kinetic energy and dissipation rate using the first-order upwind scheme. The numerical mesh was generated using quadrilateral elements consisting of 6 million cells. The convergence criteria for the simulation model was set to $1 \times 10^{-3}$ for the continuity equation, $1 \times 10^{-4}$ for the velocity equations, $1 \times 10^{-6}$ for the remaining transport equations, and $1 \times 10^{-4}$ for monitored key parameters.

### 3.2. Simulated dataset

As previously mentioned the aim this study is to illustrate the use of a data-driven surrogate model, integrated with a 1D process model, to predict the heat loads to the various heat exchanging components, the flue-gas composition and exit gas temperatures for a utility scale boiler using various high-level inputs. The inputs include the the following, the excess air ratio per burner, the total mill flowrate for the six mills in operation, the average steam temperatures for the platen and final superheaters, the fouling resistance for the platen and final superheaters, the composition of ash and moisture of the fuel and the gross calorific value of the fuel. Thus the input field has a dimensionality of $d_{inputs} = 14$.

A design of experiments (DOE) was conducted to generate a set of 180 simulation cases to obtain a representative set of results. The various model input ranges used in the DOE are given in Table 2. The ranges where selected to cover a wide range of operational loads with maximum continuous ratings (MCR) between 100% and 30%. A total of 27 output target values ($d_{targets}$) are realised for each CFD simulation case. Due to the quasi-steady nature of the CFD simulations, output target values for were taken every 50 iterations for the last 2500 iterations for a converged CFD simulation. This results in each CFD simulation case having a data solution matrix size of ($\bar{Y} \in 50 \times 27$).

The DOE matrix was populated using the Latin Hypercube Sampling (LHS) method provided in the Python *pyDOE* library. Once the CFD simulations achieved convergence, the target data, comprising of the discretized heat loads to the furnace, platen and final SHs, the exit flue-gas temperatures and combustion characteristics, was stored

**Table 1**

Summary of combustion models and constants used in the CFD model

| Model | Equation/s | Constant/s |
|---|---|---|
| *Devolatilization* | | |
| Single rate kinetic model | $\frac{dm_{vol}}{dt} = R_{vol}(m_{0,vol} - m_{vol})$, | $A_{vol} = 2 \times 10^5 [s^{-1}]$, |
| | $R_{vol} = A_{vol} exp\left(\frac{E_{a,vol}}{RT_p}\right)$ | $E_{a,vol} = 6.7 \times 10^7 [J/kmol]$ - [10] |
| *Char oxidation* | | |
| Diffusion/kinetic - [2] | $\frac{dm_{char}}{dt} = -A_p p_{O_2} \frac{R_{diff} R_c}{R_{diff} + R_c}$, | $A_c = 0.0053 [kg/m^2 s Pa]$, |
| | $R_c = A_c exp\left(\frac{E_{a,c}}{RT_p}\right)$, | $E_{a,c} = 8.37 \times 10^7 [J/kmol]$ - [10] |
| | $R_{diff} = \frac{5 \times 10^{-12}}{d_p}\left(\frac{T_g + T_p}{2}\right)^{0.75}$ | |
| *Gaseous reactions of volatiles and CO* | | |
| Eddy dissipation model - [1] | $R_{k,r,P} = \vartheta_{k,r} M_{w,k} A B \rho \frac{\varepsilon}{k} min\left(\frac{\sum_p Y_p}{\sum_j \vartheta_{j,r} M_{w,j}}\right)$, $\quad A = 4.0, \; B = 0.5$ | |
| | $R_{k,r,R} = \vartheta_{k,r} M_{w,k} A \rho \frac{\varepsilon}{k} min\left(\frac{Y_R}{\varepsilon_{R,r} M_{w,R}}\right)$ | |

**Table 2**

Design of experiments input ranges for simulations

| Input variables | Min | Max | Units |
|---|---|---|---|
| Total fuel flow rate for mills 1 to 6 | 39.5 | 120.2 | $kg/s$ |
| Fuel proximate analysis moisture mass fraction, $Y_{H_2O}$ | 0.025 | 0.085 | $kg/kg$ |
| Fuel proximate analysis ash mass fraction, $Y_{ash}$ | 0.259 | 0.559 | $kg/kg$ |
| Platen SH fouling thermal resistance, $R_{platen}$ | 0.004 | 0.007 | $m^2 K/W$ |
| Final SH fouling thermal resistance, $R_{final}$ | 0.01 | 0.017 | $m^2 K/W$ |

| Dependent variables | Function of | Units |
|---|---|---|
| Higher heating value | Fuel constituents $Y_{H_2O}$ and $Y_{ash}$ | $J/kg$ |
| Excess air | Total fuel flow rate for mills | $\%$ |
| Platen SH internal steam temperature | $R_{platen}$ | $K$ |
| Final SH internal steam temperature | $R_{final}$ | $K$ |

for each case. The target dataset was split with 80% being used for training purposes and the remainder for testing. To reduce the training time of the neural networks a min-max normalization was utilized scaling all the dataset entries to values between $0 \rightarrow 1$.

## 4. Model development

The present work makes use of two types of machine learning models, namely a standard artificial neural network (ANN/MLP) and a mixture density designated model connected to a standard ANN (MDN-ANN). The following section will discuss the hyper parameter tuning and final selected model configuration. The programming language Python 3.7.8 and the Tensorflow machine learning libraries were utilized in the present study.

### 4.1. Model configuration

The goal of the final machine learning configuration is to be able to predict the heat load distributions to the furnace, platen SH and final SH walls, the combustion characteristics and flue-gas temperatures at the exit, using the high level

inputs given in Table 2.

The outputs for the MLP will consist of $y = 50 \times 180 = 9000$ outputs. Thus, for a data batch size, $m_b$, the output tensor for the MLP model will be $\hat{Y} \in \mathbb{R}^{m_b \times y \times 27}$ since $d_{targets} = 27$. However, the MDN-ANN models output data will consist of three parts, namely; the mixing coefficients tensor of shape $\overline{\pi} \in \mathbb{R}^{m_b \times y \times K}$, the output standard deviation tensor of shape $\overline{\sigma} \in \mathbb{R}^{K \times m_b \times y \times 27}$ and the predicted means of tensor shape $\overline{\mu} \in \mathbb{R}^{K \times m_b \times y \times 27}$. The input data fed into both the MLP and MDN-ANN will have shape $\overline{X} \in \mathbb{R}^{m_b \times d_{inputs}}$. The input features will be varied based on the DOE, to account for burner mill biassing, fuel quality and SH heating component fouling.

### 4.2. Hyper-parameter tuning & final model selection

Table 4 highlights the hyper-parameter search spaces for both the ANN and ANN-MDN model. The ANN-MDN has an added parameter namely the number of additional distributions the ANN-MDN would fit to the output data.

**Table 3**
Hyper-parameter search space for fully connected NN and MDN models

| Parameter | NN search space | MDN search space |
|---|---|---|
| Number of distributions | - | 1,2,3,4 |
| Number of layers | 2,3,4 | 2,3,4 |
| Number of neurons per layer | 10, 40, 80, 100 | 10, 40, 80, 100 |
| Learning rates | 1e-3, 1e-4, 1e-5, 1e-6 | 1e-3, 1e-4, 1e-5, 1e-6 |
| Mini batch sizes | 32, 64, 128, 256 | 32, 64, 128, 256 |

**Table 4**
ANN model selection results

| Parameter | NN search space | MDN search space |
|---|---|---|
| Number of distributions | - | 2,3,4 |
| Number of layers | 2,3,4 | 2,3,4 |
| Number of neurons per layer | 10, 40, 80, 100 | 10, 40, 80, 100 |
| Learning rates | 1e-3, 1e-4, 1e-5 | 1e-3, 1e-4, 1e-5 |
| Mini batch sizes | 16, 32, 64 | 16, 32, 64 |

The hyper-parameter search was conducted in a sequential manner with the number of layers and neurons per layer being the initial step, after which the learning rates were varied, followed by the batch sizes. The ANN-MDN hyper-parameter tuning was conducted in a similar manner only that an additional step is required to establish the best amount of distributions.

### 4.3. Process model integration

Th validation and the subsequent case-studies of Section 5, make use of an integrated surrogate and network based-process model. The process model is used to capture the thermodynamic response of the water/steam side of the utility boiler under investigation, the the developed data-driven surrogate model providing the gas-side thermal predictions.

A C# script is used to access the Python API available in the process modelling software, Flownex SE® 2021. This allows for predictions to be made using the trained MDN-ANN model (surrogate model). The most probable predictions are retrieved using the script and transferred to the respective process model components as inputs. A schematic of the surrogate and process models integration is provided in Figure 4. fhhf

## 5. Results and discussion

A illustrated in the previous section the MDN-ANN model is selected model used for the subsequent validation and fuel quality case studies.

### 5.1. Multiple load validation

Measured plant data for a 100%, 81% and 60% MCR load ratings was made available and is to be used compare the accuracy of the integrated model.

### 5.2. Utility boiler response to poor fuel combustion

The fuel quality is an important factor when determining the Fuels with total moisture contents exceeding The following results illustrate the effects poor quality fuel has on the case studies boiler operational. The study made use of the developed surrogate model to investigate the steady state operation of the boiler burning poor quality coal. In this section the results of a study conducted

## 6. Conclusion

The present work has shown it is possible

## CRediT authorship contribution statement

**B.T. Rawlins:** Methodology, Software, Validation, Formal analysis, Investigation,Writing original draft, Visualization.. **Ryno Laubscher:** Writing review & editing, Methodology, Resources, Conceptualization.. **Pieter Rousseau:** Writing review & editing, Resources, Conceptualization.
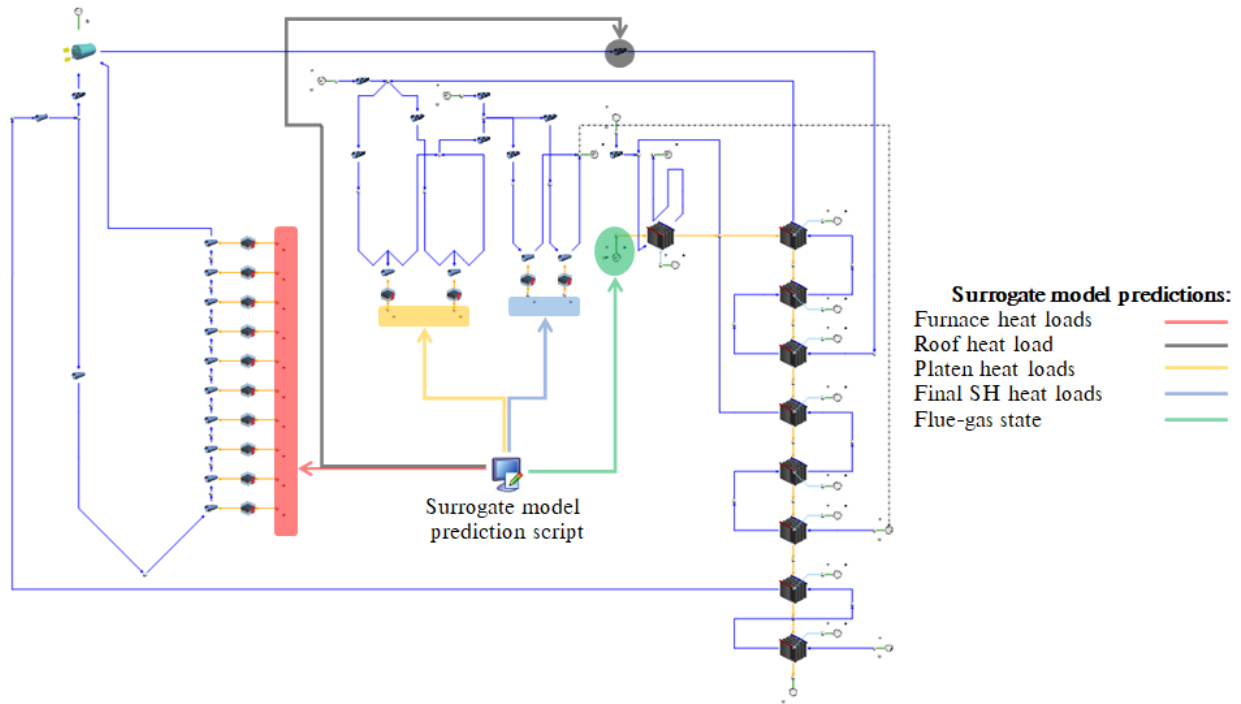
**Figure 4:** Schematic of integrated surrogate and network based process model

# References

[1] , 2021. ANSYS Fluent Theory Guide. 20 ed., Ansys Inc.

[2] Baum, M., Street, P., 1971. Predicting the combustion behaviour of coal particles. Combustion science and Technolodgy 3, 231.

[3] Bishop, C., 1994. Mixture Density Networks. Technical Report. Aston University. doi:10.1007/978-3-322-81570-5_8.

[4] Goodfellow, I., Bengio, Y., Courville, A., 2017. Deep Learning. First edit ed., MIT Press, Chennai.

[5] Kochenderfer, M., Wheeler, T., 2019. Algorithms for optimization. 1 ed., The MIT Press, Cambridge, Massachusetts.

[6] Modlinski, N., 2010. Computational modeling of a utility boiler tangentially-fired furnace retrofitted with swirl burners. Fuel Processing Technology 91, 1601–1608. doi:10.1016/j.fuproc.2010.06.008.

[7] Oko, E., Wang, M., Zhang, J., 2015. Neural network approach for predicting drum pressure and level in coal-fired subcritical power plant. Fuel 151, 139–145. URL: http://dx.doi.org/10.1016/j.fuel.2015.01.091, doi:10.1016/j.fuel.2015.01.091.

[8] Rasmussen, C., Williams, C., 2006. Gaussian Processes for Machine Learning. 1 ed., The MIT Press, Cambridge, Massachusetts.

[9] Rawlins, B.T., Laubscher, R., Rousseau, P., 2021. Validation of a thermal non-equilibrium Eulerian-Eulerian multiphase model of a 620 MWe pulverized fuel power boiler., in: Skatulla, S. (Ed.), 12th South African Conference on Computational and Applied Mechanics (SACAM2020), MATEC Web of Conferences, Cape Town. doi:https://doi.org/10.1051/matecconf/202134700004.

[10] Sheng, C., Moghtaderi, B., Gupta, R., Wall, T.F., 2004. A computational fluid dynamics based study of the combustion characteristics of coal blends in pulverised coal-fired furnace. Fuel 83, 1543–1552. doi:10.1016/j.fuel.2004.02.011.

[11] Versteeg, H., Malalasekera, W., 2007. Introduction to Computational Fluid Dynamics, The finite volume method. Second ed., Pearson Prentice Hall. doi:10.1002/9781119369189.

[12] Wen, D., Pan, Y., Chen, X., Aziz, M., Zhou, Q., Li, N., 2022. Analysis and prediction of thermal stress distribution on the membrane wall in the arch-fired boiler based on machine learning technology. Thermal Science and Engineering Progress 28, 101137. doi:10.1016/j.tsep.2021.101137.