Technical Report

Surrogate-based Shape Optimization of Immersion Nozzle in Continuous Casting

Tokinaga NAMBA* Nobuhiro OKADA

Abstract

In continuous casting, molten steel is fed from the tundish into the mold through an immersion nozzle (submerged entry nozzle, SEN). Alumina-based inclusions in the molten steel can adhere to and accumulate in the immersion nozzle, thereby limiting continuous casting operation. To prevent nozzle clogging, argon (Ar) gas is injected into the immersion nozzle. However, Ar bubbles can enter the mold with the molten steel and become trapped in the solidifying shell, causing bubble-related defects in the slab. To suppress such defects, it is effective either to keep Ar bubbles away from the solidification interface or to wash away bubbles adhering to the solidification interface by molten steel flow. The molten steel flow in the mold is strongly affected by the immersion nozzle geometry. In this paper, we investigate shape optimization of the immersion nozzle to reduce Ar bubbles trapped in the solidifying shell. A numerical model of molten steel flow and heat-transfer solidification in the mold is combined with an optimization method. During the optimization process, the amount of Ar bubbles trapped in the solidifying shell is evaluated using a neural network to improve computational efficiency. We also report an application of this method to immersion nozzle shape exploration and discuss the effectiveness of the obtained nozzle shape in reducing trapped Ar bubbles.

1. Introduction

In continuous casting, molten steel is supplied from the tundish into the mold through the immersion nozzle (submerged entry nozzle, SEN). Alumina-based inclusions in the molten steel can adhere to and accumulate in the immersion nozzle, thereby limiting continuous casting operation. To suppress nozzle clogging, argon (Ar) gas is injected into the immersion nozzle. While Ar injection helps maintain casting, Ar bubbles entering the mold with the molten steel may become trapped in the slab, causing bubble-related defects. These defects can lead to plating defects, posing a particular problem for IF steel used in automotive exterior materials. Ar bubbles trapped near the surface region of the slab are referred to as pinhole defects. After rolling, they can appear as elongated surface scratches. In addition, Ar bubbles trapped a few millimeters below the slab surface can cause blistering, i.e., bulges on the surface of rolled steel sheets. Such subsurface bubble defects causing bulge defects are referred to as blowhole defects in the slab. To prevent these bubble defects, it is effective either to prevent Ar bubbles from approaching the solidification interface or to remove bubbles adhering to the solidification interface by molten steel flow. Therefore, controlling molten steel flow within the mold is crucial.

Technologies have been developed to control molten steel flow and prevent the trapping of Ar bubbles within the solidifying shell. Electromagnetic stirring (EMS) and electromagnetic brake (EMBr), both utilizing electromagnetic forces, are representative examples. To suppress bubble-related defects near the slab surface, EMS is installed at the top of the mold to stir the region near the molten steel surface. EMBr, on the other hand, is installed in the central part of the mold and near the tip of the immersion nozzle. It applies braking to the downward molten steel flow. By suppressing the downward flow of the discharge stream from the immersion nozzle, it promotes flotation and separation of inclusions and Ar bubbles.

Because molten steel flow within the mold is significantly influenced by the immersion nozzle shape, studies on immersion nozzles

* Researcher, Mathematical Science & Technology Research Lab., Advanced Technology Research Laboratories
20-1 Shintomi, Futtsu City, Chiba Pref. 293-8511

have also been conducted.[1–4] However, considering the overall balance of factors such as nozzle clogging, refractory strength, and cost, traditional two-hole nozzles with a pair of discharge holes on each side are frequently used in slab casting. Consequently, discussions on this topic have become less common in recent years. On the other hand, the introduction of thin slab continuous casting machines, which handle the entire process from casting to hot rolling, has been increasing in recent years. In thin slab continuous casting machines, because of the high casting speed, research on immersion nozzle shape and mold flow has been reported.[5–9] These reports suggest that there remains room for improvement in immersion nozzles even for conventional slab continuous casting machines. Therefore, this paper addresses the determination of an immersion nozzle shape that reduces Ar bubbles trapped in the solidifying shell in conventional continuous casting machines. Specifically, Ar bubbles trapped within the top 2 mm of the slab are termed pinholes, while those trapped between 2 mm and 15 mm below the surface are termed blowholes; the objective is to reduce both.

To evaluate pinholes and blowholes, we use numerical analysis models of molten steel flow and heat-transfer solidification within the mold, which have been developed and refined over time.[10, 11] This study combines the numerical model with an optimization framework using machine learning to optimize the immersion nozzle shape, aiming to minimize pinholes and blowholes. Shape optimization techniques combining computational fluid dynamics (CFD) and optimization have been developed and applied to aircraft wing design,[12] ship design,[13] and structural design of tall buildings,[14] but have rarely been applied to steelmaking processes. A key challenge in CFD-based shape optimization is its high computational cost, which limits practicality. Typically, the optimization process requires thousands to tens of thousands of simulations. Because CFD calculations demand significant computational time, direct application of optimization techniques is difficult. To reduce computational costs and improve optimization efficiency, research on surrogate models has been actively pursued in recent years. A surrogate model is an approximation model that replaces complex and time-consuming calculations during optimization. Application cases of machine learning models such as kriging,[14] radial basis functions,[15] and (deep) neural networks[16] have been reported.[17]

In this paper, because pinholes and blowholes arise from complex coupled phenomena involving molten steel flow and heat-transfer solidification, we evaluate pinholes and blowholes using a neural network model in place of CFD during the optimization loop. Specifically, the neural network is trained using a dataset comprising pinhole and blowhole values obtained from a feasible number of CFD calculations, along with their calculation conditions. The trained model is then used to predict pinholes and blowholes for given nozzle conditions. Using this prediction, an objective function is evaluated and an optimization algorithm is applied to find the nozzle shape that minimizes pinholes and blowholes. Predictions using the neural network take only a few seconds, enabling rapid optimization.

The optimization framework employs sequential approximation optimization,[18, 19] which iteratively performs optimization to find an optimal solution. In this framework, data consisting of calculation conditions and pinhole/blowhole values obtained during the iterative process are progressively added to the dataset, updating the neural network model. The prediction accuracy of the neural network strongly depends on the dataset. With insufficient data, prediction accuracy may be inadequate, leading to discrepancies with CFD results; thus, the optimal solution is not necessarily obtained in a single optimization run. By contrast, sequential approximation optimization improves prediction accuracy as the dataset expands, enabling iterative updates to the optimal solution to reduce pinholes and blowholes.

The optimization algorithm employed is particle swarm optimization (PSO).[20] This method uses multiple particles (search points) that move through the search space according to defined rules to find an optimal solution. It is characterized by resistance to local optima and strong global search capability even for complex multimodal objective functions.

In addition, we incorporate a method for adding data points using a density function based on a radial basis function network (RBFN).[21] During sequential approximation optimization, if only optimal nozzle conditions are added to the dataset, the neural network accuracy improves near the current optimum but remains insufficient in regions without data points. Re-optimizing under such biased sampling tends to yield new optimal solutions only around the initial optimum, potentially missing better solutions in regions that were originally sparse. Therefore, it is effective to identify regions with low data density and add data points there. An RBFN places basis functions at existing data points and sums them over the space to measure data density; this paper uses Gaussian functions as the basis functions. Adding the minimum point of the RBFN-based density function to the dataset helps the neural network learn the overall landscape of the search space.

The following sections describe the mathematical model, the numerical analysis model for molten steel flow and heat-transfer solidification within the mold, and the method for optimizing the immersion nozzle shape. An implementation example of immersion nozzle shape optimization is then presented.

2. Mathematical Model

2.1 Governing equations

For numerical analysis of molten steel flow and heat-transfer solidification within the mold, the mathematical model developed by Takatani et al.[10, 11] is employed. The governing equations are the large eddy simulation (LES) model for fluid flow and the unsteady energy equation for heat-transfer solidification. The assumptions in this model are as follows.

(1) The fluid is incompressible, and the densities and specific heats of the liquid and solid phases are assumed to be constant.
(2) Darcy's law is applied as the fluid resistance in the solid-liquid coexisting phase.
(3) The solid phase is treated as a rigid body (its velocity is equal to the casting pulling speed), and regions where the solid phase volume fraction exceeds 0.8 are considered rigid bodies.
(4) Gas bubbles are treated as incompressible dispersed phases with spherical shapes and constant diameters.
(5) The momentum equation for the gas phase is the Basset-Boussinesq-Oseen-Tchen equation[22]. However, the Basset term is assumed to be negligible.
(6) In the energy balance, the enthalpy of the gas phase is assumed to be negligible.
(7) The turbulence model applied is LES.
(8) The turbulent Prandtl number is set to 1.
(9) The solidification temperature is calculated using the lever rule.[23]

The governing equations are as follows.

$$f_g + f_l + f_s = 1 \tag{1}$$

$$\frac{\partial f_g}{\partial t}+\nabla\cdot(f_g\mathbf{u}_g)=0 \tag{2}$$

$$\frac{\partial f_l}{\partial t}+\nabla\cdot(f_l\mathbf{u}_l)=-R_v \tag{3}$$

$$\frac{\partial f_s}{\partial t}+\nabla\cdot(f_s\mathbf{u}_s)=R_v \tag{4}$$

$$\rho_g\frac{D\mathbf{u}_g}{Dt}=\rho_l\frac{D\mathbf{u}_l}{Dt}+\frac{1}{2}\rho_l\left[\frac{D\mathbf{u}_l}{Dt}-\frac{D\mathbf{u}_g}{Dt}\right]-\frac{3\mu_l}{4d_g^2}\mathrm{Re}_g C_{Dg}(\mathbf{u}_g-\mathbf{u}_l)+(\rho_l-\rho_g)\mathbf{g} \tag{5}$$

$$\frac{\partial(f_l\rho_l\mathbf{u}_l)}{\partial t}+\nabla\cdot(f_l\rho_l\mathbf{u}_l\mathbf{u}_l)$$
$$=-f_l\nabla p+\nabla\cdot[f_l(\mu_l+\mu_t)(\nabla\mathbf{u}_l+\nabla\mathbf{u}_l^T)]+f_l[\beta(T-T_0)\rho_l\mathbf{g}+\gamma(\mathbf{u}_s-\mathbf{u}_l)+\mathbf{F}_{lg}] \tag{6}$$

$$\frac{\partial T}{\partial t}+\nabla\cdot(f_l\mathbf{u}_l T+f_g\mathbf{u}_g T)=\nabla\cdot[(\alpha+\alpha_t)\nabla T]+\frac{1}{C_p}R_v(-\Delta H) \tag{7}$$

$$T=\frac{(T_l-T)/m_l}{(T-T_s)/m_s+(T_l-T)/m_l} \tag{8}$$

Here, t is time, $\mathbf{u}$ is flow velocity, f is volume fraction, p is pressure, and T is temperature. ρ is density, μ is the dynamic viscosity, C_p is specific heat, α is thermal conductivity, β is thermal expansion coefficient, $\mathbf{g}$ is gravitational acceleration, and R_v is the solidification rate of molten steel. ΔH is latent heat; m_l represents the slope of the liquidus line; and m_s represents the slope of the solidus line. Subscripts l, s, and g denote the liquid phase, solid phase, and gas phase, respectively. The coefficients Re_g and C_{Dg} are given by Equations (9) and (10).

$$\mathrm{Re}_g=d_g|\boldsymbol{u}_l-\boldsymbol{u}_g|\frac{\rho_l}{\mu_l} \tag{9}$$

$$C_{Dg}=0.4+\frac{24}{\mathrm{Re}_g}+\frac{6}{1+\sqrt{\mathrm{Re}_s}} \tag{10}$$

Furthermore, μ_t and α_t represent the turbulent viscosity and turbulent thermal conductivity, respectively. The turbulent viscosity is given by Equation (11).

$$\mu_t=\rho_l(C_s\Delta^{1/3})^2\sqrt{2|\mathbf{S}||\mathbf{S}|} \tag{11}$$

Here, the coefficient $C_s=0.1$, Δ is the cell volume, and $\mathbf{S}$ is the strain-rate tensor defined by Equation (12).

$$\mathbf{S}=\frac{1}{2}(\nabla\mathbf{u}_l+\nabla\mathbf{u}_l^T) \tag{12}$$

The third term on the right-hand side of Equation (6) represents buoyancy due to temperature, where T_0 is the average temperature of the liquid phase. The fourth term represents fluid resistance in the solid-liquid coexisting phase, with the coefficient γ given by Equation (13). Here, D_2 is the second-order dendrite arm spacing.

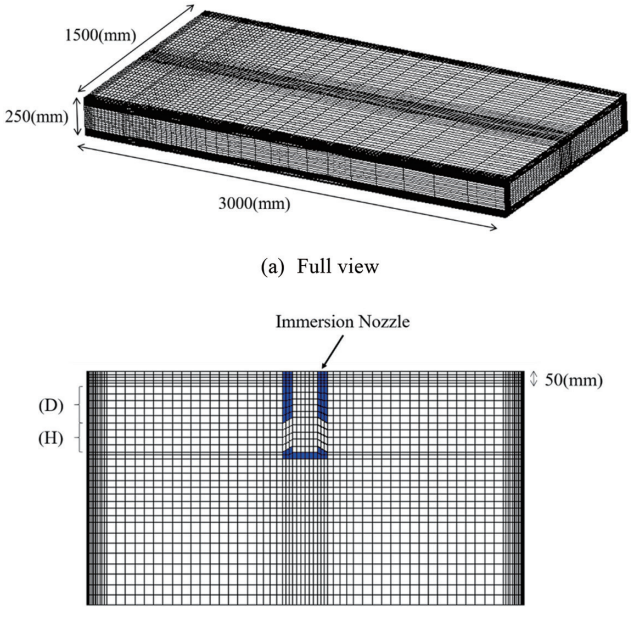
$$\gamma=\frac{\mu_l}{\rho_l}\frac{150(1-f_l)^2}{D_2^2 f_l^3} \tag{13}$$

The fifth term represents momentum exchange between the liquid and gas phases, with $\mathbf{F}_{lg}$ given by Equation (14).

$$\mathbf{F}_{lg}=f_g\frac{3\mu_l}{4d_g^2}\mathrm{Re}_g C_{Dg}(\mathbf{u}_g-\mathbf{u}_l) \tag{14}$$

2.2 Calculation conditions and method

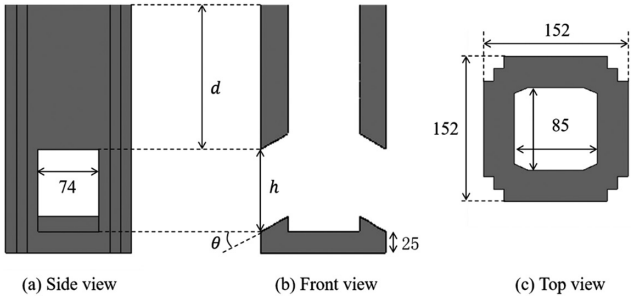
The computational domain was defined as a rectangular prism with dimensions 250 mm × 1 500 mm × 3 000 mm (thickness × width × length), as shown in **Fig. 1**. The region extending 900 mm downward from the molten steel surface was defined as the mold, with a heat transfer coefficient of 800 W/m^{2}K. Below this region, secondary cooling was assumed, with a heat transfer coefficient of 400 W/m^{2}K. The immersion nozzle geometry is shown in **Fig. 2**. The immersion depth d (mm), discharge-hole height h (mm), and nozzle angle θ (degree) were treated as continuous parameters to be optimized. The number of computational grid cells was fixed at 36 × 80 × 23 (thickness × width × length) for the region beyond 516 mm in the length direction. For the region up to 516 mm, the dimensions 36 × 80 (thickness × width) were fixed. However, the number of subdivisions in the lengthwise direction was adjusted for regions (D) and (H) in Fig. 1 (b), respectively, based on the immersion depth and discharge-hole height values. For region (D), the minimum number of divisions was used such that the grid spacing was less than 25 mm. For region (H), the minimum number of divisions was used such that the spacing was 25 mm or less. If the number of divisions for region (H) was less than 4, it was set to 4.



(a) Full view

(b) Enlarged view of the top

Fig. 1 Computational grid



(a) Side view (b) Front view (c) Top view

Fig. 2 Geometry of the immersion nozzle. All dimensions are in mm, except angle degree.

Discretization was performed on a staggered grid, and calculations were carried out using a solution method based on the SOLA method[24] for arbitrary shapes. A second-order central difference scheme was used for the advection and diffusion terms. The advection term was advanced in time using the Euler method, and the diffusion term was calculated implicitly.

The casting pulling speed was 1.4 m/min. The amount of Ar bubbles flowing from the immersion nozzle into the mold was determined based on Reference 25), with a bubble diameter of 0.5 mm at 2.668 Nl/min, 1.0 mm at 0.944 Nl/min, and 1.5 mm at 0.493 Nl/

min. Note that the optimization targets pinholes and blowholes, both of which are most prevalent at a bubble diameter of 0.5 mm.

2.3 Mechanism of bubble trapping

The mechanisms by which bubbles are trapped at the solidification interface are considered to involve Saffman forces due to velocity gradients,[26] interfacial-tension differences caused by concentration gradients of surfactant elements,[27, 28] and electromagnetic forces,[29] but the details remain unclear. For example, it has been reported that pinholes are more likely to occur at higher sulfur concentrations,[30] suggesting that interfacial-tension differences caused by sulfur concentration gradients can have a significant influence. Equation (15) shows the force $\mathbf{F}_I$ acting on a bubble in molten steel due to interfacial-tension differences.[27]

$$\mathbf{F}_I = \frac{8}{3}\pi R^2 C_0 \frac{\partial \sigma}{\partial C_L}\frac{V_s}{D_L}(1-K_\varepsilon)\exp\left(-\frac{V_s(x-\delta)}{D_L}\right) \quad (15)$$

Here, C_L is the solute concentration in the boundary layer, C_0 is the solute concentration in the liquid phase before solidification, R is bubble radius, σ is interfacial tension, K_ε is the effective partition coefficient, V_s is the growth rate of the solidification interface, x is the distance from the solidification interface, δ is the thickness of the concentration boundary layer, and D_L is the diffusion coefficient of the solute. The effective partition coefficient K_ε is expressed by Equation (16), where K_0 is the equilibrium distribution coefficient.

$$K_\varepsilon = \frac{K_0}{K_0+(1-K_0)\exp(-V_s\delta/D_L)} \quad (16)$$

To account for the force exerted on a bubble due to interfacial-tension differences via Equation (16) in numerical analysis of a continuous caster, it is sufficient to compute sulfur concentration distribution and segregation due to solidification. However, resolving a concentration boundary layer thickness on the order of approximately 0.01 to 1 mm is difficult with current computational capabilities. On the other hand, this force becomes negligible if the concentration boundary layer thickness is sufficiently smaller than the bubble diameter. The concentration boundary layer thickness is given by the relationship reported by Burton et al. in Equation (17).[31]

$$\delta = 0.285 D_L^{1/3} \nu^{1/6} \omega^{-1/2} \quad (17)$$

Here, ν is the kinematic viscosity coefficient, and ω is the angular velocity due to liquid-phase rotation. Thus, the concentration boundary layer thickness becomes a function solely of the flow velocity across the solidification interface: the higher the flow velocity, the thinner the boundary layer.

From the above, it is appropriate to model the number of bubbles trapped within the solidifying shell as a function of molten steel flow velocity when evaluating pinholes in numerical simulations of continuous casting. The rate η_ε at which bubbles are trapped into the solidifying shell is assumed as shown in Equation (18).

$$\eta_\varepsilon = n_\varepsilon R_s P_\varepsilon(|\mathbf{u}|) \quad (18)$$

where n_ε is the bubble volume fraction, R_s is the solidification velocity, and $P_\varepsilon(|\mathbf{u}|)$ is the bubble capture probability dependent on molten steel flow velocity. The volume fraction n_ε of bubbles trapped by the shell is transported by the shell withdrawal velocity $\mathbf{u}_s$, as shown in Equation (19).

$$\frac{\partial n_\varepsilon}{\partial t} + \nabla\cdot(\mathbf{u}_s n_\varepsilon) = -\eta_\varepsilon \quad (19)$$

Values such as 0.05 m/s[32] and 0.15 m/s[33] have been reported as threshold bubble capture velocities. However, due to differences in the definition of the solidification interface position and the location where velocity is evaluated, the exact value remains unclear. Moreover, a model where bubbles are captured at 0.05 m/s but not captured at all at 0.051 m/s is unrealistic. Therefore, the capture probability was modeled as an exponential function that decreases with increasing velocity, as shown in Equation (20).

$$P_\varepsilon(|\mathbf{u}|) = \exp(-C_0|\mathbf{u}|) \quad (20)$$

In this exponential model, the constant C_0 in Equation (20) was modified to 100, ensuring that the capture probability at 0.15 m/s approaches 1×10^{-8}. In addition, because the flow velocity in elements where solidification begins and the solid fraction increases significantly decreases due to Darcy's law, the flow velocity at a solid phase fraction of 0.1 was used as the solidification-interface flow velocity.

2.4 Calculation method for pinhole and blowhole

Pinhole and blowhole values were evaluated by averaging over 100 s after reaching steady-state conditions, i.e., 1 000 s after the start of the CFD calculation in real-time equivalent. Specifically, pinholes y_P and blowholes y_B were calculated using Equations (21) and (22).

$$y_P = \left(\sum_{n=1}^{N}\sum_{i=1}^{M_P}\frac{C_{i,P}^n V_{i,P}}{100}\delta t\right)\Big/\left(V_c\sum_{i=1}^{M_P}V_{i,P}\right) \quad (21)$$

$$y_B = \left(\sum_{n=1}^{N}\sum_{i=1}^{M_B}\frac{C_{i,B}^n V_{i,B}}{100}\delta t\right)\Big/\sum_{i=1}^{M_B}V_{i,B} \quad (22)$$

Here, N is the maximum number of time steps, δt is the time step width, M_P and M_B are the total numbers of cells within 2 mm from the surface and between 2 mm and 15 mm on the shell side, respectively, and $V_{i,P}$ and $V_{i,B}$ are the volumes of the i-th cell in the respective regions. $C_{i,P}^n$ and $C_{i,B}^n$ represent the volume fraction of Ar bubbles trapped within the i-th cell in the respective region at time step n. Furthermore, V_c is the volume of a sphere with a diameter of 0.5 mm, i.e., $V_c=\pi(0.5\times10^{-3})^3/6$. Therefore, pinholes are evaluated by number density, and blowholes by volume fraction.

3. Method for Optimizing Immersion Nozzle Shape

3.1 Problem setting

In the immersion nozzle shape optimization considered in this paper, the immersion depth d, discharge-hole height h, and nozzle angle θ are design variables. Therefore, the search space is set as in Equation (23).

$$S=\{(d,h,\theta)^T\in\mathbb{R}^3 : d_{lb}\le d\le d_{ub}, h_{lb}\le h\le h_{ub}, \theta_{lb}\le\theta\le\theta_{ub}\} \quad (23)$$

Due to design constraints, lower bounds d_{lb}, h_{lb}, θ_{lb} and upper bounds d_{ub}, h_{ub}, θ_{ub} were established for the search space. The upper and lower bounds used in this study are shown in **Table 1**.

Once these three variables are determined, pinholes and blowholes can be calculated using CFD. This process is expressed as a function F, as shown in Equation (24).

$$(y_P, y_B)^T = F(d,h,\theta) \quad (24)$$

The immersion nozzle shape optimization problem for reducing pinholes and blowholes can then be formulated as in Equation (25).

$$\mathrm{Find}\,(d^*,h^*,\theta^*)^T = \underset{(d,h,\theta)^T\in S}{\operatorname{argmin}}\, L(F(d,h,\theta)) \quad (25)$$

where $L=L(y_P,y_B)$ represents the objective function. Because the objective is to simultaneously minimize pinholes and blowholes, the

Table 1 Upper and lower bounds of the search space

d_{lb} (mm)	h_{lb} (mm)	θ_{lb} (degree)
125	60	15
d_{ub} (mm)	h_{ub} (mm)	θ_{ub} (degree)
225	120	40

problem is classified as multi-objective optimization. To simplify, this paper reduces the two objectives into a single objective function defined by their average, as shown in Equation (26), converting it into a single-objective optimization problem.

$$L(y_P, y_B) = \frac{1}{2}|y_P| + \frac{1}{2}|y_B| \tag{26}$$

In general, optimization requires repeated evaluations of the model F to compute the objective function. Solving the problem defined by Equation (26) directly is impractical from the standpoint of computational cost. Therefore, this paper uses a neural network as a low-cost model $\tilde{F}$ that approximates F, without using F itself. In this case, the optimization problem becomes the approximation problem expressed by Equation (27).

$$\mathrm{Find}\,(\tilde{d}^*, \tilde{h}^*, \tilde{\theta}^*)^T = \underset{(\tilde{d},\tilde{h},\tilde{\theta})^T \in S}{\operatorname{argmin}} L(\tilde{F}(d,h,\theta)) \tag{27}$$

Furthermore, to improve the reliability of the optimal solution, we adopt the framework of sequential approximation optimization,[18, 19] which iteratively updates the surrogate model and the optimal solution while adding data points. Ultimately, we solve the problem expressed by Equation (28).

$$\mathrm{Find}\,(\tilde{d}^k, \tilde{h}^k, \tilde{\theta}^k)^T = \underset{(\tilde{d},\tilde{h},\tilde{\theta})^T \in S}{\operatorname{argmin}} L(\tilde{F}^k(d,h,\theta)),\ k = 1,2,\cdots \tag{28}$$

3.2 Prediction of pinhole and blowhole using neural networks

The neural network model takes the immersion nozzle design variables (h, d, θ) as inputs and outputs the predicted pinhole and blowhole values (y_P, y_B). The model was constructed by supervised learning. Given a dataset $\Omega = \{(\boldsymbol{x}_j, \boldsymbol{y}_j) = (h_j, d_j, \theta_j, y_{P,j}, y_{B,j}) | j = 1, \cdots, m\}$, the dataset Ω is first split into a training dataset Ω_{train} and a test dataset Ω_{test} in a 7:3 ratio. Subsequently, a dataset Ω'_{train} is created by standardizing the input and output variables of Ω_{train} using Equation (29).

$$\boldsymbol{x}' = \frac{\boldsymbol{x} - \mu}{\sigma} \tag{29}$$

where $\boldsymbol{x}'$ is the standardized value of $\boldsymbol{x}$, and μ and σ are the mean and variance of each variable, respectively. Using the corresponding means and variances, Ω'_{test} is created by transforming the input and output variables of Ω_{test} using Equation (29).

The neural network $\tilde{F}$ is a function composed of multiple nested (layered) functions $\hat{F}_i$, as shown in Equation (30).

$$\tilde{F}(\boldsymbol{x}) = \hat{F}_N\left(\hat{F}_{N-1}\left(\cdots\left(\hat{F}_1(\boldsymbol{x})\right)\right)\right) \tag{30}$$

The function $\hat{F}_i$ is more specifically expressed by Equation (31).

$$\hat{f}_i(\hat{\boldsymbol{y}}'_{i-1}) = a(\mathbf{W}_i \hat{\boldsymbol{y}}'_{i-1}) \tag{31}$$

Here, $\mathbf{W}_i$ is the weight matrix, and $\hat{\boldsymbol{y}}'_{i-1}$ is the vector obtained by inserting 1 into the first row of the output vector $\hat{\boldsymbol{y}}_{i-1}$ of layer $i-1$, defined in Equation (32).

$$\hat{\boldsymbol{y}}_{i-1} = \begin{cases} \hat{F}_{i-1}\left(\hat{F}_{i-2}\left(\cdots\left(\hat{F}_1(\boldsymbol{x})\right)\right)\right) & \text{if } i \geq 2 \\ \boldsymbol{x}, & \text{if } i = 1 \end{cases} \tag{32}$$

The function a is an activation function; various types have been proposed, including the hyperbolic tangent function and the rectified linear unit (ReLU).[34]

Because weights cannot be predetermined, the neural network updates its weights (learns) by minimizing a predefined error function so that the model outputs match the training data Ω'_{train}. Stochastic gradient descent[35] was employed for learning. The adopted error function consisted of the mean squared error plus a regularization term, as defined in Equation (33).

$$E(\boldsymbol{y}, \tilde{\boldsymbol{y}}; \mathbf{W}) = \frac{1}{m}\sum_{j=1}^{m}\left\{(y_{P,j} - \tilde{y}_{P,j})^2 + (y_{B,j} - \tilde{y}_{B,j})^2\right\} + \lambda\left(\sum_{i,j}|w_{i,j}|^p\right)^{\frac{1}{p}} \tag{33}$$

Table 2 Parameters of neural network structure and search range

N	n_{unit}	r
2 to 7	1 to 100	0.0 to 0.9
p	λ	η
1 or 2	10^0 to 10^{-3}	10^0 to 10^{-4}

where $\boldsymbol{y}$ and $\tilde{\boldsymbol{y}}$ represent the given output data and the predicted value, respectively; $w_{i,j}$ denotes components of the weight matrices; and λ is a parameter representing the degree of regularization. The sum of the regularization terms is taken over all weights in the neural network, and $p \geq 1$.

While the regularization term aims to prevent overfitting, dropout was also employed. Dropout randomly removes a fixed proportion of components from the output vector of each layer, thereby reducing the dimension of the inputs to the next layer.[36] This helps prevent stochastic gradient descent from getting stuck in local optima. To accelerate convergence, initial weights were set following Reference 37). The neural network was implemented using the Python library TensorFlow.[38]

The parameters determining the neural network structure include the number of layers N, the number of units n_{unit} in each layer, the dropout rate r, the type of activation function, the parameters p and λ in the regularization term, and the learning rate η in stochastic gradient descent. These must be determined before training. This paper employs Bayesian optimization to tune these parameters. **Table 2** shows the parameters to be adjusted and their search ranges. For implementation, the Python library Optuna[39] was used. The process of selecting candidate parameter values, training the network, and evaluating predictions on Ω'_{test} was repeated. The model with the best prediction accuracy on the test data was selected as the final model. The number of Bayesian optimization trials was set to 1 000, and prediction accuracy on the test data was measured using the coefficient of determination R^2.

3.3 Particle swarm optimization

Particle swarm optimization (PSO)[20, 40] is a method in which multiple particles (search points) move through the search space according to defined rules to find an optimal solution. Here, we describe PSO combined with neural-network prediction of pinholes and blowholes.

The position of particle i is represented by the vector $\boldsymbol{x}_i = (d_i, h_i, \theta_i)^T$ within the search space defined by the immersion nozzle design variables. Here, i is the particle index. Each particle also has a velocity vector $\boldsymbol{v}_i = (v_{i1}, v_{i2}, v_{i3})^T$. Furthermore, each particle stores the best position found so far, $\boldsymbol{lbest}_i = (lbest_{i1}, lbest_{i2}, lbest_{i3})^T$, and the corresponding objective value $L(\tilde{F}(\boldsymbol{lbest}_i))$. The swarm stores the best position found among all particles, $\boldsymbol{gbest} = (gbest_1, gbest_2, gbest_3)^T$.

Figure 3 shows the computational flowchart. First, particle positions $\boldsymbol{x}_i^{(0)}$ and velocities $\boldsymbol{v}_i^{(0)}$ are generated randomly within the search space for the specified number of particles. Next, the objective value $L(\tilde{F}(\boldsymbol{x}_i^{(0)}))$ is computed for each particle using the neural network, and $\boldsymbol{lbest}_i$ is updated. Subsequent processing is repeated until the maximum number of iterations is reached or convergence is achieved. Let t denote the iteration count. From the current position $\boldsymbol{x}_i^{(t)}$, the vectors pointing toward $\boldsymbol{lbest}_i^{(t)} - \boldsymbol{x}_i^{(t)}$ and $\boldsymbol{gbest}^{(t)} - \boldsymbol{x}_i^{(t)}$, together with the previous velocity $\boldsymbol{v}_i^{(t)}$, are used to update the velocity according to Equation (34).

$$\boldsymbol{v}_i^{(t)} = w\boldsymbol{v}_i^{(t)} + c_1 r_1 (\boldsymbol{lbest}_i^{(t)} - \boldsymbol{x}_i^{(t)}) + c_2 r_2 (\boldsymbol{gbest}^{(t)} - \boldsymbol{x}_i^{(t)}) \tag{34}$$

Here, r_1 and r_2 are uniform random numbers distributed between 0 and 1, while w, c_1, and c_2 are pre-set weight parameters. The particle position is then updated by Equation (35).

$$\boldsymbol{x}_i^{(t+1)} = \boldsymbol{x}_i^{(t)} + \boldsymbol{v}_i^{(t)} \tag{35}$$

If the j-th component $\boldsymbol{x}_{ij}^{(t+1)}$ falls below the lower bound, it is set to the lower bound; if it exceeds the upper bound, it is set to the upper bound.

In this paper, the maximum iteration count was used as the termination condition, and optimization was performed under the conditions listed in **Table 3**.

3.4 Adding sample points using density functions via radial basis function networks

A radial basis function network (RBFN) is a neural network consisting of three layers: an input layer, one hidden layer, and an output layer. Let the existing sample points consisting of design variables be $\{\boldsymbol{x}_j\}_{j=1,\cdots,m}$, with no duplicates. The output of the RBFN is given by Equation (36).

$$z(\boldsymbol{x};\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i K(\boldsymbol{x},\boldsymbol{x}_i) \tag{36}$$

where $K(\boldsymbol{x},\boldsymbol{x}_i)$ is the basis function for the i-th unit and $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_m)$ is the vector of weights for the basis functions. This paper uses the Gaussian function defined in Equation (37) as the basis function.

$$K(\boldsymbol{x},\boldsymbol{x}_i) = \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}_i)^T(\boldsymbol{x}-\boldsymbol{x}_i)}{r_i^2}\right) \tag{37}$$

Here, r_i is the radius of the i-th data point. Various definitions of the radius have been proposed,[21, 41, 42] but this paper follows Reference 42) and uses Equation (38).

$$r_i = \frac{d_{i,\max}}{\sqrt{n^n}\sqrt{m-1}} \tag{38}$$

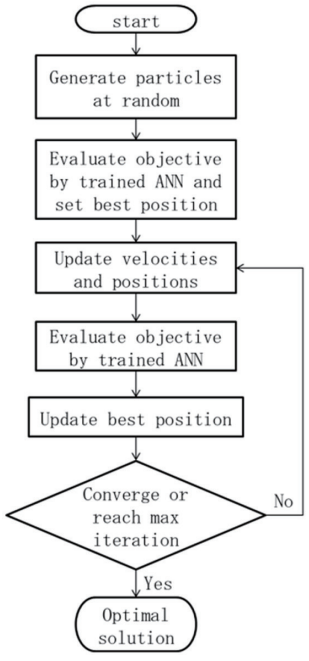


Fig. 3 Neural network-based particle swarm optimization process

Table 3 Particle swarm optimization parameters

Number of particles	w	c_1	c_2	Max iteration
100	0.5	0.14	0.14	500

where n is the dimension of the search space and $d_{i,\max}$ is the maximum distance between the i-th data point and other data points. The Euclidean distance is adopted, as shown in Equation (39).

$$d_{i,\max} = \max_{j=1,\cdots,m} \|\boldsymbol{x}_j - \boldsymbol{x}_i\| = \max_{j=1,\cdots,m} \sqrt{|d_j-d_i|^2+|h_j-h_i|^2+|\theta_j-\theta_i|^2} \tag{39}$$

The density function is a function that generates local minima in sparse regions of the sample points using an RBFN. The method for determining additional sample points using the density function is as follows.

(1) Prepare a column vector with all components set to 1.

$$\boldsymbol{y} = (1, \cdots, 1)^T_{m\times 1} \tag{40}$$

(2) Determine the coefficients $\boldsymbol{\alpha}^* = (\alpha_1^*, \cdots, \alpha_m^*)$ using Equation (41).

$$\boldsymbol{\alpha}^* = (\boldsymbol{K}^T\boldsymbol{K}+\boldsymbol{\varepsilon})^{-1}\boldsymbol{K}^T\boldsymbol{y} \tag{41}$$

Here, $\boldsymbol{K}$ and $\boldsymbol{\varepsilon}$ are matrices defined by Equations (42) and (43), respectively. The ε in Equation (43) is a weight parameter, which was set to $\varepsilon = 1.0\times10^{-2}$ in this paper.

$$\boldsymbol{K} = \begin{bmatrix} K(\boldsymbol{x}_1,\boldsymbol{x}_1) & K(\boldsymbol{x}_1,\boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_1,\boldsymbol{x}_m) \\ K(\boldsymbol{x}_2,\boldsymbol{x}_1) & K(\boldsymbol{x}_2,\boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_2,\boldsymbol{x}_m) \\ \vdots & \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_m,\boldsymbol{x}_1) & K(\boldsymbol{x}_m,\boldsymbol{x}_2) & \cdots & K(\boldsymbol{x}_m,\boldsymbol{x}_m) \end{bmatrix} \tag{42}$$

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon & 0 & \cdots & 0 \\ 0 & \varepsilon & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \varepsilon \end{bmatrix} \tag{43}$$

(3) Solve the following minimization problem (44), and use the obtained minimum solution as an additional sample point. Note that the density function refers to the objective function in this problem.

$$\text{Find } \boldsymbol{x}^* = \underset{\boldsymbol{x}\in S}{\operatorname{argmin}} \sum_{i=1}^{m} \alpha_i^* K(\boldsymbol{x},\boldsymbol{x}_i) \tag{44}$$

3.5 Implementation method

Figure 4 shows the overall calculation flowchart. At the start of optimization, there are no data available for training the neural net-

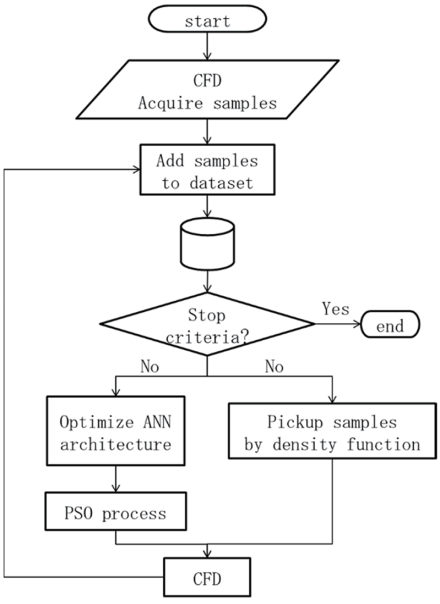


Fig. 4 Sequential approximate optimization process

work. Therefore, CFD is used to calculate pinholes and blowholes, creating an initial dataset corresponding to the nozzle conditions. Because the extrapolation performance of the neural network is limited, the initial dataset consists of a small number of points selected mainly around conditions corresponding to the boundaries of the search range. Specifically, CFD calculations were performed for the following combined conditions: immersion depth (mm) of 60, 100, 120; discharge-hole height (mm) of 125, 155, 175, 195, 225; and nozzle angle (degrees) of 15, 20, 25, 30, 35, 40.

Subsequently, using this dataset, we searched for an optimal neural network structure, performed training, and then executed PSO using the obtained model. PSO yielded candidate nozzle conditions with potential for improvement. In parallel, additional nozzle conditions were derived from sparse regions using the RBFN-based density function to create new data points. For nozzle conditions obtained via PSO and the density function, CFD calculations were performed to obtain accurate pinhole and blowhole values. All nozzle conditions and the corresponding pinhole and blowhole values obtained up to this point were compiled and added to the dataset. To obtain as many candidate solutions as possible and expand the dataset—thereby improving surrogate prediction accuracy within a short timeframe—the addition of sample points via PSO and the density function was repeated three times each. This process was iterated, and all processing was terminated upon reaching the predetermined number of iterations.

4. Optimization Results and Discussion

Figure 5 shows the CFD results for the initial immersion nozzle conditions prepared before optimization (black circles; base conditions), the neural network predictions for immersion nozzle conditions obtained after 10 iterations of sequential approximation optimization (black triangles), and the CFD results for candidate optimal immersion nozzle conditions obtained using the RBFN-based density function under the same framework (white circles). Because the ideal minimum values of both pinholes and blowholes are assumed to be zero, conditions toward the lower-left of the figure are better. The neural network predictions (black triangles) indicate improved conditions compared with the initial conditions (black circles). While the CFD results (white circles) do not always coincide with the neural network predictions (black triangles), they show similar overall trends. Therefore, by repeatedly performing neural network-based search and CFD-based verification, improved conditions can be identified efficiently.

Condition A in Fig. 5 is the optimal point obtained manually (black circle), while Condition B is the optimal point confirmed by CFD calculation (white circle). Hereafter, these are referred to as the basic condition and the improved condition, respectively. **Table 4** shows pinholes and blowholes for the basic and improved conditions. Both values are smaller under the improved condition, corresponding to reductions of approximately 56% for pinholes and approximately 10% for blowholes.

Figure 6 shows the corresponding nozzle shapes. Under the basic condition, because the discharge-hole height and nozzle angle are close to the lower limits of the explored range, a nearly horizontal, high-velocity discharge flow tends to reduce pinholes and blowholes. **Figure 7**(a) shows the flow field under the basic condition: the discharge jet impinging on the narrow face splits into upward and downward branches. This flow pattern is known as the double-roll flow pattern.[43] The upward flow reaches the molten steel surface and agitates the surface, which suppresses trapping of Ar bubbles at the solidification interface and can reduce pinholes. In addi-

Table 4 Pinholes and blowholes by condition

	Base condition	Improved condition
Pinhole	2 322	1 002
Blowhole	13.8	12.3

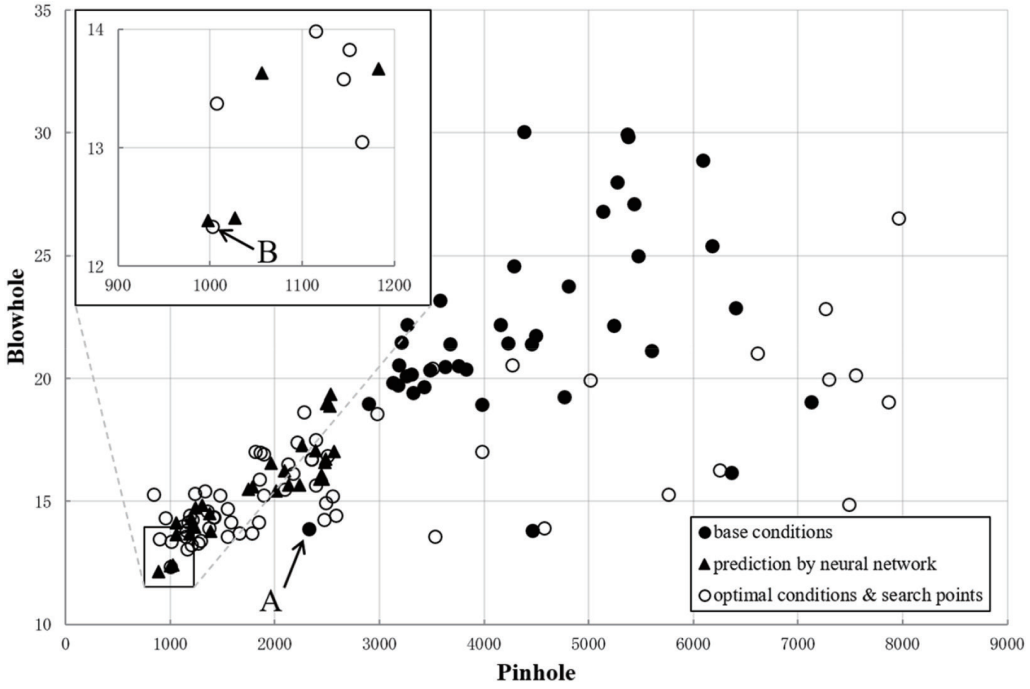


Fig. 5 Scatter plot for pinhole and blowhole

tion, the recirculating flow associated with the discharge jet and the double-roll pattern produces a cleaning effect on bubbles adhering to the solidification interface, which can reduce blowholes.

The improved condition enhances these effects. In terms of immersion nozzle geometry, while the discharge-hole height is comparable to that of the basic condition, the smaller nozzle angle results in a more horizontally directed discharge jet. As shown in Fig. 7 (b), this promotes a stronger upward flow, increasing flow velocity near the molten steel surface and likely contributing to pinhole reduction. The increased immersion depth is considered to induce flow over a wider region of the mold, thereby helping to wash bubbles trapped in the solidifying shell and reduce blowholes. **Figure 8** shows the magnitude of flow velocity. Compared with the basic condition, the improved condition yields higher flow velocities overall. Furthermore, if flow reaching the molten steel surface turns downward near the mold center where the immersion nozzle is located, Ar bubbles are transported together with the flow, which can worsen blowholes.

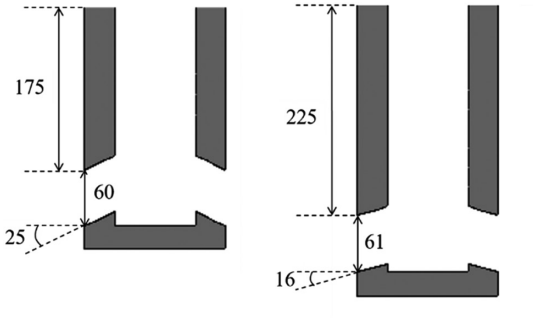


Fig. 6 Comparison of immersion nozzle shapes

Therefore, increasing the immersion depth is considered to prevent excessively strong flow toward the center of the molten steel surface.

5. Conclusion

This paper reported an immersion nozzle shape optimization method that combines CFD calculations of molten steel flow and heat-transfer solidification in a continuous casting mold with a neural-network-based optimization approach, together with an implementation example. The proposed method integrates a mathematical model for CFD calculations, sequential approximation optimization as the optimization framework, a neural network surrogate model to replace CFD evaluations during optimization, and particle swarm optimization as the search algorithm. Applying this method to the exploration of immersion nozzle shapes aimed at reducing pinholes and blowholes resulted in reductions of approximately 56% in pinholes and approximately 10% in blowholes compared with the basic condition obtained by prior CFD-based investigation. In addition, nozzle shapes with deeper immersion depth, lower discharge-hole height, and a smaller nozzle angle showed favorable trends. These trends are likely associated with a nearly horizontal, high-velocity discharge jet and the resulting strong upward flow that agitates the molten steel surface and reduces pinholes. Furthermore, enhanced flow over a wider region of the mold is considered to wash away bubbles adhering to the solidification interface, contributing to blowhole reduction.

Future challenges include optimizing nozzle shape while considering EMS and EMBr. Moreover, although this paper converted the multi-objective problem into a single-objective one by averaging the evaluation values for pinholes and blowholes, it is necessary to investigate multi-objective optimization in future work.

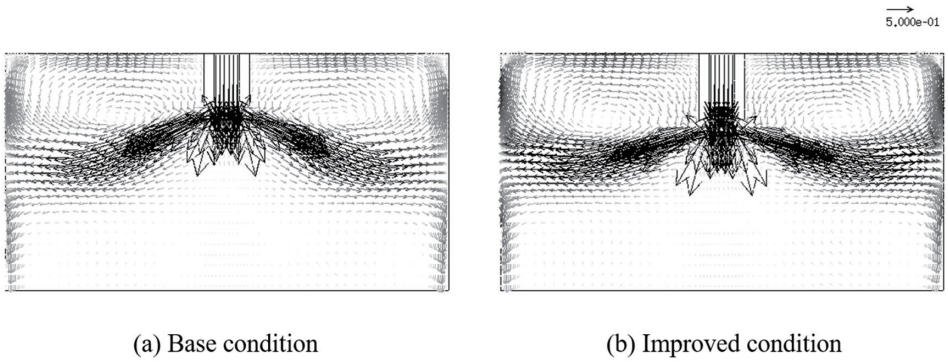


Fig. 7 Comparison of velocity in the mold

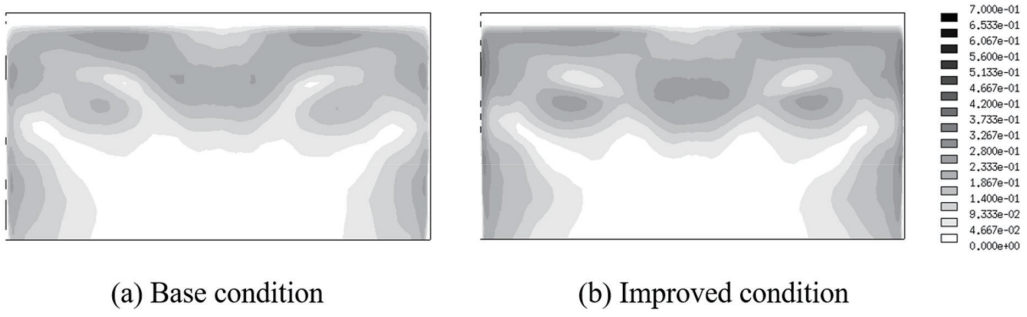


Fig. 8 Comparison of velocity magnitude in the mold

References

1) Yokotani, S., Haseo, S., Asako, Y., Takagi, S., Ayata, K, Szekely, J., Hara, S.: Tetsu-to-Hagané. 82, 581 (1996) (in Japanese), https://doi.org/10.2355/tetsutohagane1955.82.7_581
2) Yoshida, J., Iguchi, M., Yokota, S.-I.: Tetsu-to-Hagané. 89, 264 (2002) (in Japanese), https://doi.org/10.2355/tetsutohagane1955. 88.5_264
3) Takatani, K.: Tetsu-to-Hagané. 90, 751 (2004) (in Japanese), https://doi.org/10.2355/tetsutohagane1955.90.10_751
4) Tsukaguchi, Y., Hayashi, H., Kurimoto, H., Yokoya, S., Marukawa, K., Tanaka, T.: Tetsu-to-Hagané. 95, 33 (2009) (in Japanese), https://doi.org/10.2355/tetsutohagane.95.33
5) Li, B.W., Tian, X.Y., Wang, E.G., He, J.C.: Acta Metall. Sin-Engl. 20, 15 (2007), https://doi.org/10.1016/S1006-7191(07)60003-9
6) Park, H.-S., Nam, H., Yoon, J.K.: ISIJ Int. 41, 974 (2001), https://doi.org/10.2355/isijinternational.41.974
7) Liu, H., Yang, C., Zhang, H., Zhai, Q., Gan, Y.: ISIJ Int. 51, 392 (2011), https://doi.org/10.2355/isijinternational.51.392
8) Morales, R.D., Tang, Y., Nitzl, G., Eglsäeer, C. Hckl, G.: ISIJ Int. 52, 1607 (2012), https://doi.org/10.2355/isijinternational.52.1607
9) Xuan, M.,Chen, M.: Metals. 11, 1223 (2021), https://doi.org/10.3390/met11081223
10) Takatani, K., Tanizawa, Y., Mizukami, H., Nishimura, K.: ISIJ Int. 41, 1252 (2001), https://doi.org/10.2355/isijinternational.41.1252
11) Takatani, K.: ISIJ Int. 43, 915 (2003), https://doi.org/10.2355/isijinternational.43.915
12) Leifsson, L., Koziel, S.: J. Comput. Sci. 10, 45 (2015), https://doi.org/10.1016/j.jocs.2015.01.003
13) Yang, C., Huang, F.: J. Hydrodyn. 28, 947 (2016), https://doi.org/10.1016/S1001-6058(16)60696-0
14) Bernardini, E., Spence, S.M.J., Wei, D., Kareem, A.: J. Wind Eng. Ind. Aerodyn. 144, 154 (2015), https://doi.org/10.1016/j.jweia. 2015.03.011
15) Song, X., Lv, L., Sun, W., Zhang, J.: Struct. Multidisc. Optim. 60, 965 (2019), https://doi.org/10.1007/s00158-019-02248-0
16) Wu, H., Liu, X., An, W., Chen, S., Lyu, H.: Compt. Fluids. 198, 104393 (2020), https://doi.org/10.1016/j.compfluid.2019.104393
17) Li, J., Du, X., Martins, J.R.R.A.: Prog. Aerosp. Sci. 134, 100849 (2022), https://doi.org/10.1016/j.paerosci.2022.100849
18) Nakayama, H., Yun, Y., Yoon, M.: Sequential Approximate Multiobjective Optimization Using Computational Intelligence. Springer, Berlin, Heidelberg, 113 (2009), https://doi.org/10.1007/978-3-540-88910-6
19) Kitayama, S.: Kougakukei No Saiteki Sekkeihou (Kikai Gakusyu Wo Katuyou Shita Riron To Jissen). Kyoritsu Syuppan, Tokyo, (2021) (in Japanese)
20) Kennedy, J., Eberhart, R.: Proceedings of ICNN'95 – International Conference on Neural Networks, 4, 1942 (1995), 10.1109/ICNN.1995.488968
21) Kitayama, S., Arakawa, M., Yamazaki, K.: Optim. Eng. 12, 535 (2011), https://doi.org/10.1007/s11081-010-9118-y
22) Hinze, J.O.: Turbulence. 2nd ed., McGraw Hill, New York, 460 (1975)
23) Wang, Z., Tanaka, M., Inoue, T.: Nihon-Kikai-Gakkai-Ronbunsyu A. 53, 1735 (1987)
24) Hirt, C.W., Nichls, B.D. Romero, N.C.: SOLA-A Numerical Solution Algorithm for Transient Fluid Flows. Los Alamos Sci. Lab, Los Alamos, LA-5852 (1975)
25) Toh, T., Hasegawa, H., Harada, H.: ISIJ Int. 41, 1245 (2001), https://doi.org/10.2355/isijinternational.41.1245
26) Saffman, P.G.: J. Fluid Mech. 22, 385 (1965), https://doi.org/10.1017/S0022112065000824
27) Mukai, K., Lin, W. : Tetsu-to-Hagané. 80, 533 (1994) (in Japanese), https://doi.org/10.2355/tetsutohagane1955.80.7_533
28) Mukai, K., Zhong, L., Zeze, M.: ISIJ Int. 46, 1810 (2006), https://doi.org/10.2355/isijinternational.46.1810
29) Taniguchi, S., Brimacombe, J.K.: Tetsu-to-Hagané. 80, 24 (1994), https://doi.org/10.2355/tetsutohagane1955.80.4_312
30) Miyake, T., Morishita, M., Nakata, H., Kokita, M.: ISIJ Int. 46, 1817 (2006), https://doi.org/10.2355/isijinternational.46.1817
31) Burton, J.A., Prim, R.C., Slichter, W.P.: J. Chem. Phys. 21, 1987 (1953), https://doi.org/10.1063/1.1698728
32) Miki, Y.. Ohno, H., Kishimoto, Y., Tanaka, S.: Tetsu-to-Hagané. 97, 423 (2011) (in Japanese), https://doi.org/10.2355/tetsutohagane.97.423
33) Mollinger, A.M., Nieuwstadt, F.T.M.: J. Fluid Mech. 316, 285 (1996), https://doi.org/10.1017/S0022112096000547
34) Apicella, A., Donnarumma, F., Isgrò, F., Prevete, R.: Neural Netw. 138, 14-32 (2021), https://doi.org/10.1016/j.neunet.2021.01.026
35) Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge, MA, (2016)
36) Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: J. Mach. Learn. Res. 15, 1929-1958 (2014)
37) Glorot, X., Bengio, Y.: Proc. 13th Int. Conf. on Artificial Intelligence and Statistics (AISTATS2010). PMLR, 249 (2010)
38) TensorFlow, Google LLC: https://www.tensorflow.org/ (accessed 2021-10-20)
39) Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Proc. 25rd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. 2623-2631 (2019), https://doi.org/10.1145/3292500. 3330701
40) Aiyoshi, E., Yasuda, K.: MetaHeuristics To Ouyou (MetaHeuristics and Applications). IEEJ. Tokyo, 69 (2007) (in Japanese)
41) Haykin, S.: Neural Networks (A Comprehensive Foundation). Macmillan College Publishing, New York, USA, (1994)
42) Nakayama, H., Arakawa, M., Sasaki, R.: Optim. Eng. 3, 201 (2002), https://doi.org/10.1023/A:1020971504868
43) Thomas, B.G., Zhang, L.: ISIJ Int. 41, 1181 (2001), https://doi.org/10.2355/isijinternational.41.1181

Tokinaga NAMBA
Researcher
Mathematical Science & Technology Research Lab.
Advanced Technology Research Laboratories
20-1 Shintomi, Futtsu City, Chiba Pref. 293-8511

Nobuhiro OKADA
Ph. D., General Manager, Head of Dept.
Mathematical Science & Technology Research Lab.
Advanced Technology Research Laboratories