In-House Developed Thermal-Hydraulic Code for Research Reactor Thermal Margin Analysis

Junyoung Lim [a], Kiwon Song [b], Jaehyun Cho [a*]
[a]*Energy System Engineering, Chung-ang University, 84 Heukseok-ro, Dongjak-gu, Seoul, Republic of Korea*
[b]*Korea Atomic Energy Research Institute, 989-111 Daedeok Daero, Yuseong Gu, Daejeon, Republic of Korea*
[*]*Corresponding author: jcho@cau.ac.kr*

Keywords : Thermal-hydraulic design, Thermal Margin, Research Reactor, Python

1. Introduction

During the preliminary design stage of a research reactor, a computational tool is required to rapidly evaluate key thermal-hydraulic quantities in a coolant channel—such as pressure drop, bulk temperature, wall temperature, and the onset of nucleate boiling—based on the available design(input) data. The TMAP(Thermal hydraulic and Margin Analysis code for Plate type fuel assembly), developed at KAERI, was designed to analyze thermal-fluid characteristics and thermal margins for plate-type research reactors under steady-state forced convection and natural circulation conditions[1].

The original TMAP, implemented in MATLAB, is primarily configured for forced-convection downflow and natural-circulation upflow conditions. However, open-pool type research reactors with high heat flux, such as OPAL(Australia) and RA-10(Argentina), operate with upflow, which motivates the implementation of forced-convection upflow capability[2]. Accordingly, this study develops a Python-based thermal-hydraulic code tailored to plate-type fuel research reactor channels and extends the framework to include forced-convection upflow by generalizing the treatment of gravity terms and boundary conditions with respect to flow direction. In addition, the code is modularized into preprocessing–solving–postprocessing components to facilitate correlation selection and design-parameter changes, enabling repeated calculations and sensitivity analyses in the preliminary design phase.

The code developed in this study is not intended to directly perform a full 3-D analysis of the research reactor. Instead, it provides a 1-D axial thermal-hydraulic analysis for representative coolant channels, while incorporating core- and assembly-level design inputs such as total power, number of fuel plates, and number of assemblies. Thus, the overall reactor configuration and operating conditions are reflected in the input data, whereas the actual thermal-hydraulic calculations are carried out for the average channel and the intermediate(peak) channel. Fig. 1. presents a schematic of the representative-channel-based analysis concept.

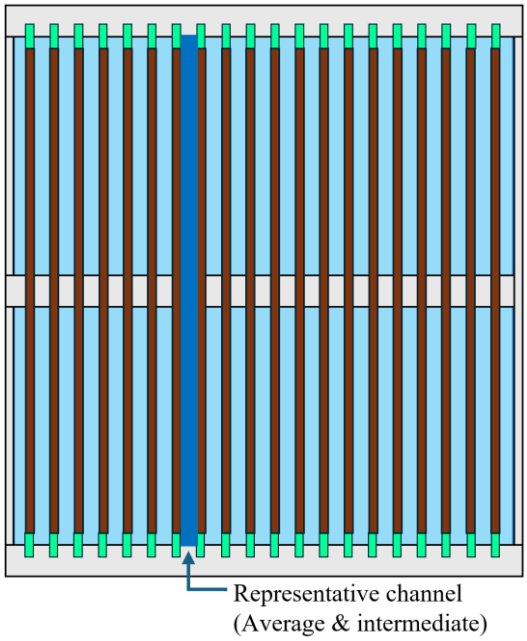

Fig. 1. Concept of the plate-type fuel assembly and the representative-channel

2. Methods(Thermal-Hydraulic Code)

This section describes the computational structure of the developed code, the solution procedure, and the main governing relationships implemented.

2.1 Code Structure and Execution Workflow

A one-dimensional axial thermal-hydraulic Python code was developed to support the preliminary design of plate-type fuel research reactors. The code constructs operating conditions and geometric inputs from an Excel file, generates an axial local heat-flux distribution from a power-shape function, and then solves both the intermediate(peak) channel and the average channel. The outputs include pressure $P(z)$, bulk temperature $T_b(z)$, wall temperature $T_w(z)$, heat transfer coefficient $h(z)$, and thermal-margin indicators based on ONB/OFI/CHF(ONBR, OFIR, and DNBR).

Fluid properties(saturation temperature, viscosity, thermal conductivity, specific heat, density, etc.) are evaluated as functions of pressure and bulk temperature. The code updates state variables using a water/steam property library(steam tables / IAPWS-based functions).

Pressure is handled in Pa by default, while unit conversion(e.g., to bar) is applied when required by specific correlations(e.g., the ONB correlation). Heat flux is also internally converted and consistently treated in W/m² to ensure unit consistency with the energy equation.

The major code modules and the corresponding flow diagram are summarized in Table I and Fig. 2.

Fig. 2. illustrates the overall execution procedure. After loading the Excel inputs, the axial power shape is generated, and the preprocessing stage computes the axial grid, hydraulic diameter and flow area, the local heat-flux distribution, and the power normalization factor. Next, depending on the flow-direction flag(FDIR), a top-loss correction(downflow) or a natural-circulation iteration(if applicable) is performed. The solver then computes $P(z)$, $T_b(z)$, and $T_w(z)$ using one-dimensional marching and a 2-pass property update scheme, and the postprocessing stage evaluates ONB/OFI/CHF-based margin indicators.

2.2 Input data and Variable Definitions

Input data are loaded from Sheet1(Input1) and Sheet2(Input2) of the Excel file, and the main input variables are summarized in Table II.

2.3 Preprocessing: Grid generation and heat-flux distribution

The axial grid is defined based on the channel height CHANHT. The number of axial nodes is set as

$$NNT = round(CHANHT \cdot 1000) + 1 \quad (1)$$

and the axial spacing is

$$\Delta z = CHANHT/(NNT - 1) \quad (2)$$

For a rectangular channel, the flow cross-sectional area A_{flow}, heated area A_{heat}, and hydraulic diameter D_h are computed as follows. When HOPTION=2(one-side heated), CHANTKO is used and the geometric quantities are recomputed accordingly.

$$A_{flow} = CHANWT \cdot CHANTK \quad (3)$$
$$A_{heat} = 2(FUELWT \cdot FUELHT) \quad (4)$$
$$D_h = \frac{4(CHANWT \cdot CHANTK)}{2CHANWT + 2CHANTK} \quad (5)$$

HOPTION is a variable used to distinguish the heating boundary condition. HOPTION=1 denotes both-sides heated, whereas HOPTION=2 denotes one-side heated. In both-sides heated condition, both surfaces of the fuel plate exchange heat with the coolant, while in the one-side heated condition, only one surface participates in heat transfer. Therefore, the two conditions differ in terms of heated area and heated perimeter, which leads to differences in the local heat flux, wall temperature,

Table I: Module structure and role of the Python thermal-hydraulic code

Module(File name)	Role
io.py/units.py	Input loading and unit conversion
power_shape.py	Load axial power shape
preprocess.py	Grid generation; Hydraulic diameter/flow area/heated area calculation; heat-flux normalization
top_losses.py	Top-loss application for downflow
solver_intermediate.py /solver_average.py	1D marching solver for pressure/temperature/heat transfer
post_margin.py	ONB, OFI, CHF(DNBR) evaluation
save_results.py /plot_results.py	Save results and plot generation
natural_circulation.py	Natural circulation condition mass-flux solver

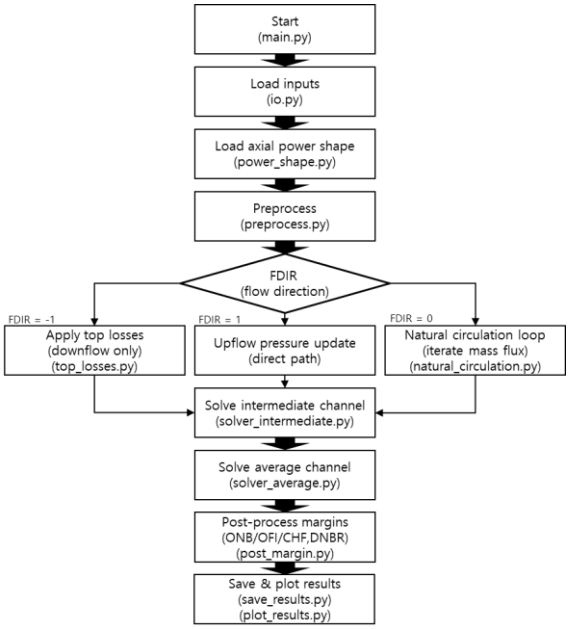

Fig. 2. Flow diagram of the Python thermal hydraulic code

bulk coolant temperature distribution, and thermal-hydraulic safety margin results.

The axial local heat flux is constructed by linearly interpolating the shape data (ZR, QVZ) from the input Excel file onto the normalized coordinate $z/CHANHT$ to generate $shape(z)$, and then scaling it with AFLUX and the peak factor R_{peak}:

Intermediate channel:

$$q''_{loc}(z) = shape(z) \cdot AFLUX \cdot R_{peak} \quad (6)$$

Average channel:

$$q''_{loc,avg}(z) = shape(z) \cdot AFLUX \quad (7)$$

To satisfy the total power consistency condition, a normalization factor Covar is computed and applied to the heat-flux distributions:

$$Covar = \frac{TOP \cdot 10^6 \cdot POD}{\sum[q''_{loc,avg} \cdot (2 \cdot \Delta z \cdot FUELWT) \cdot (NRD \cdot NAS)]} \quad (8)$$

Finally, the corrected heat-flux distributions are obtained by

$$q''_{loc} \cdot Covar \rightarrow q''_{loc} \quad (9)$$

$$q''_{loc,avg} \cdot Covar \rightarrow q''_{loc,avg} \quad (10)$$

2.4 One-Dimensional Model and Numerical Solution Procedure

The channel inlet is defined as $z = 0$ and the outlet as $z = CHANHT$. The code marches axially from $i = 0$ to $NNT - 1$ to compute $P(z)$, $T_b(z)$, and $T_w(z)$. The inlet boundary condition (P_{in}, T_{in}) is prescribed. For downflow($FDIR = -1$), a top reference pressure is first evaluated using the top-loss(Stop) inputs and then incorporated into the channel pressure calculation. For upflow($FDIR = 1$) and natural circulation($FDIR = 0$), the gravity-term sign convention is applied consistently in the same coordinate system.

The solvers(solver_intermediate.py and solver_average.py) march axially from $i = 1$ to NNT and compute pressure, bulk temperature, and thermal properties at each node. To improve numerical stability and handle the coupling between properties and state variables, a 2-pass scheme(itrpr = 1, 2) is employed. In the first pass, bulk temperature is predicted using the averaged heat flux between $(i-1)$ and i, and in the second pass, the temperature is recomputed using the heat flux at node i to finalize the values. This 2-pass approach mitigates numerical oscillations arising from pressure–temperature–property coupling and improves solution robustness.

(1) Energy equation (bulk temperature update)

Bulk temperature is updated as

$$T_{b,i} = T_{b,i-1} + \frac{[(q''^*/HOPTION) \cdot (2 \cdot FUELWT)]}{(\rho_0 \cdot u_0 \cdot c_{p,i} \cdot A_{flow})} \cdot \Delta z \quad (11)$$

where $q''^* = 0.5(q''_{i-1} + q''_i)$ is used in the first pass and $q''^* = q''_i$ is used in the second pass. For the Average channel, the initial velocity accounts for VRATIO:

$$u_0 = \frac{MFLUX}{\rho_0 \cdot VRATIO} \quad (12)$$

(2) Velocity update based on continuity

Table II: Input variables

Variable	Description	Unit	Remarks
AFLUX	Average heat-flux	[W/m²]	-
MFLUX	Mass flux	[kg/m² · s]	$u_0 = MFLUX/\rho_0$
VRATIO	Average channel VRATIO	[-]	$u_0 = \frac{MFLUX}{\rho_0 \cdot VRATIO}$
PIN/TIN	Inlet pressure /temperature	[MPa] [°C]	-
FDIR	Flow direction	[-]	-1: downflow 1: upflow 0: natural circulation
CHANHT	Channel height	[mm]	-
CHANWT /CHANTK	Channel width/gap	[mm]	HOPTION=1
CHANTKO	Channel gap(option)	[mm]	HOPTION=2
FUELWT/ FUELHT	Fuel width /height	[mm]	-
NRD /NAS	No. of plates per assembly /No. of assembly	[-]	-
PPF	Power Peaking Factor	[-]	$R_{peak} = \frac{PPF}{\max(QVZ)}$
TOP /POD	Total power and distribution factor	[MW] [-]	Covar
F_OPTION	Friction-factor correlation	[-]	See Table III
Nu _OPTION	Nu/HTC correlation	[-]	See Table III
CHF _OPTION	CHF correlation	[-]	See Table III
HOPTION	q/HOPTION and geometry re-calculation	[-]	Energy equation and geometry
Stop /Sbot	Top/bot. loss arrays	[-]	Sheet2
OOPTION/ OXIDETK/ OXIDEK	Oxide layer thermal resistance model	[-] [μm] [W/mK]	Optional

Velocity is updated according to the density change:

$$u_i = \frac{\rho_0 \cdot u_0}{\rho_i} \quad (13)$$

(3) Pressure equation (gravity term + friction loss)

Pressure is integrated axially as

$$P_i = P_{i-1} + \rho_i \cdot g \cdot (-FDIR \cdot \Delta z) - \rho_i \cdot \left(\frac{f_i u_i^2}{2D_h}\right) \cdot \Delta z \quad (14)$$

Thus, the sign of the gravity term is controlled by FDIR, and friction losses are modeled in a Darcy–Weisbach form.

The friction factor is implemented in correlations.py via selectable options. Laminar flow follows (Eckert & Irvine)[3]: $f = C_f/Re$ for $Re < 2500$, and turbulent options include Blasius and Colebrook formulations.

(4) Heat transfer coefficient (Nusselt correlations + flow-direction handling)

The Nusselt number/heat transfer coefficient is computed via correlations.py. In the current implementation, when $FDIR = 1$ (upflow), the FDIR used for HTC computation is forced to 0, so that the upflow case uses a branch distinct from the downflow branch($FDIR = -1$). Available options include Dittus–Boelter [4] and Sudo [5].

Wall temperature is computed as

$$T_{w,i} = T_{b,i} + \left(\frac{q_i''}{h_i}\right) \quad (15)$$

If the oxide option is enabled($OOPTION = 1$), an additional thermal resistance(oxide thickness and conductivity) is included. Fuel and cladding temperatures are computed axially by linearly interpolating the temperature-thermal conductivity tables provided in the input.

The friction factor, heat transfer coefficient(Nu/HTC), and CHF options are summarized in Table III.

In particular, for HTC evaluation, when $FDIR > 0$(upflow), the code redefines $FDIR = 0$ internally when evaluating the Nusselt correlation. Therefore, the upflow condition is implemented such that the "FDIR == 0 branch" correlation is used in the Nu/HTC computation.

2.5 ONB/OFI/CHF Margin Evaluation (Postprocessing)

The postprocessing module(post_margin.py) computes ONB, OFI, and CHF-based DNBR using the calculated P, T_b, T_w, h, and q'', thereby evaluating thermal margins.

(1) ONB calculation: iterative convergence of q''_{ONB}

At each axial node, q''_{ONB} is initialized with the local heat flux and iteratively updated until the coupled

Table III: Correlations and options

	Option	Correlation /Implementation	Remarks
f	F_OPTION=1	Blasius: $f = 0.316Re^{-0.25}$	Re>2500
f	F_OPTION=2	Colebrook	Re>2500
Nu	Nu_OPTION=1	Dittus–Boelter: $Nu = 0.023 \cdot Re^{0.8} \cdot Pr^{0.4}$	$h = \frac{kNu}{D_h}$
Nu	Nu_OPTION=2	Sudo	FDIR branch
CHF	CHF_OPTION=1	Kaminaga[6]	FDIR branch
CHF	CHF_OPTION=2	Mirshak–Durant–Towell	
CHF	CHF_OPTION=3	Labuntsov	
CHF	CHF_OPTION=4	Bernath	

conditions are satisfied:

$$Initial: q_{ONB} \leftarrow q''_{loc} \quad (16)$$

$$T_{ONB} = T_{sat}(P) + \frac{5}{9}\left(\frac{q_{ONB}}{1082 \cdot P^{1.156}}\right)^{\left(\frac{P^{0.0234}}{2.16}\right)} \quad (17)$$

$$T_c = T_b + \frac{q_{ONB}}{h} \quad (18)$$

$$q_{ONB} = (T_{ONB} - T_b) \cdot h \quad (19)$$

$$Convergence: |T_{ONB} - T_c| < 10^{-3} \quad (20)$$

The ONB margin and ONB ratio are defined as

$$\Delta T_{ONB} = T_{ONB} - T_w, ONBR = \frac{q_{ONB}}{(q''_{loc}/HOPTION)} \quad (21)$$

(2) OFI

The code evaluates the OFI-limit using $\Delta T_{\text{sat}} = T_{\text{sat,out}} - T_{b,\text{in}}$ and a geometry correction factor $R = 1/(1 + \eta D_h/FUELHT)$ with $\eta = 32.5$, and defines

$$\text{OFIR} = \frac{q''_{\text{OFI}}}{q''/HOPTION} \quad (22)$$

(3) CHF and DNBR

Four CHF correlations are implemented as selectable options: Kaminaga, Mirshak–Durant–Towell, Labuntsov, and Bernath. DNBR is computed from the evaluated Q_{CHF} as

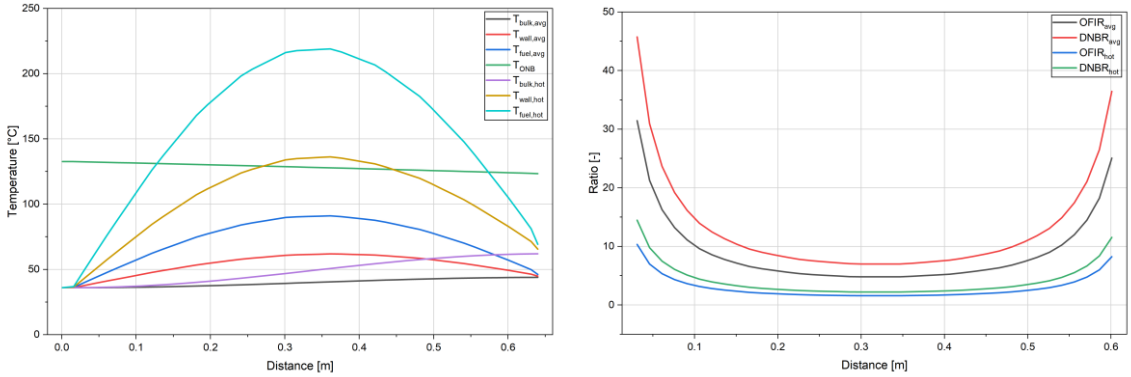

Fig. 3(a). Temperature results of the average and hot channels calculated by the Python code; (b) OFIR and DNBR results of the average and hot channels calculated by the Python code

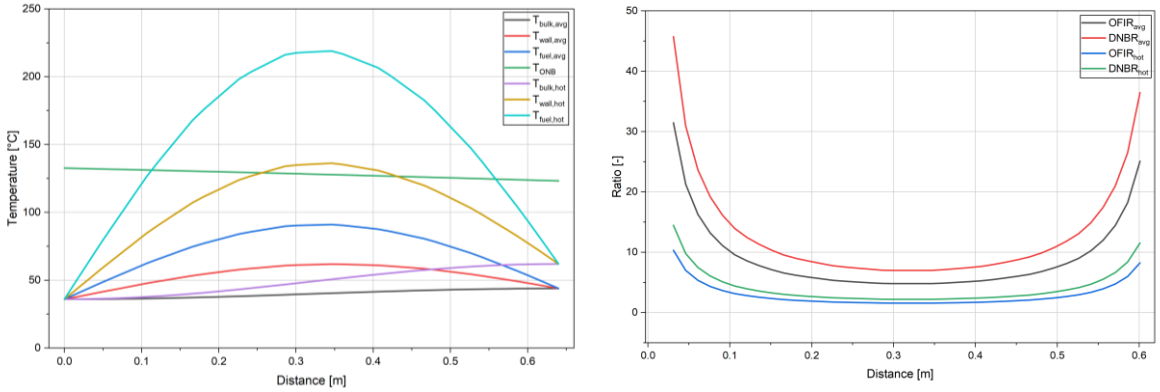

Fig. 3(c). Temperature results of the average and hot channels calculated by TMAP; (d) OFIR and DNBR results of the average and hot channels calculated by TMAP

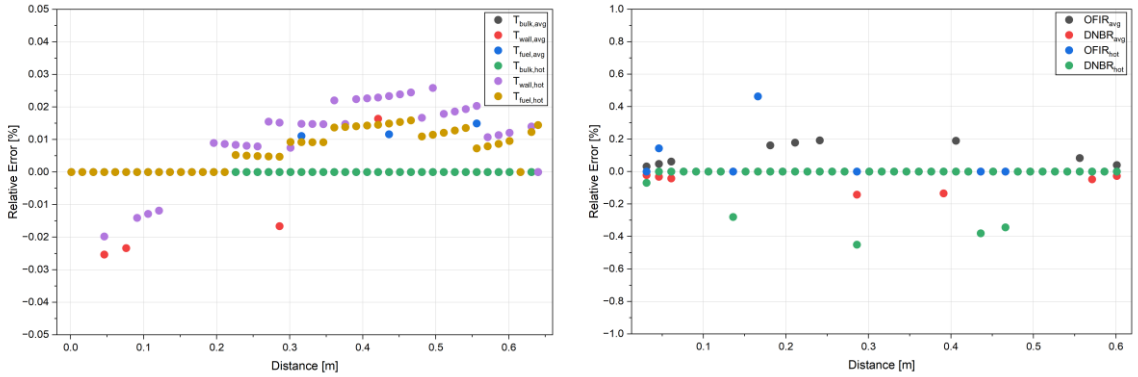

Fig. 4. Relative error of the comparison variables between the results calculated by Python code and TMAP

$$DNBR = \frac{Q_{CHF}}{(q''_{loc}/HOPTION)} \quad (23)$$

and a hot-channel-factor-adjusted quantity is also reported:

$$DNBR_{hc} = \frac{(Q_{CHF}/FQ)}{(q''/HOPTION)} \quad (24)$$

3. Verification

For the downflow condition, the developed Python thermal-hydraulic code was verified by comparison against TMAP results. The two calculations were performed using identical inputs(geometry, operating conditions, loss coefficients, and correlation options). After confirming that both result files share the same axial coordinate, axial distributions and representative

summary metrics were compared at identical locations. The comparison variables were $P(z)$, $T_b(z)$, $T_w(z)$, $h(z)$, $OFIR(z)$, and $DNBR(z)$, and quantitative assessment was conducted using the maximum absolute error and the mean absolute percentage error(MAPE).

As a result, for the average channel, the maximum absolute error in pressure $P(z)$ was 5.158Pa(MAPE 0.00162%), the maximum absolute error in $T_w(z)$ was 0.01°C(MAPE 0.00145%), and the maximum absolute error in the heat transfer coefficient $h(z)$ was 5.6 W/m²·K(MAPE 0.0113%), indicating excellent agreement with TMAP. In terms of representative metrics, the total pressure drop in the average channel was 56,365.554Pa(Python) versus 56,370.712Pa (TMAP), corresponding to a difference of 5.158Pa(0.00915%), and $DNBR_{min}$ was identical(6.99 vs 6.99). For the peak channel, the pressure-drop difference was 5.626 Pa(0.0126%), the peak inner-cladding temperature difference was 0.02°C(0.00574%), and the $DNBR_{min}$ was identical(2.76 vs 2.76). These results confirm that the developed code accurately reproduces TMAP calculations under downflow conditions. Fig. 3. presents the comparison plots between the Python code and TMAP. Fig 4. presents the relative error of the comparison variables between the results calculated by Python code and TMAP.

For the upflow conditions, a stepwise verification will be performed in future work using literature data or reference analysis results. In particular, the validity of the current implementation—where the common branch (FDIR = 0 branch) is applied in the heat-transfer-coefficient calculation—will be examined first.

4. Conclusions

In this study, a Python-based thermal-hydraulic calculation code was developed to support preliminary research reactor design, and the existing downflow-oriented logic was extended to include upflow conditions. The preprocessing–solving–postprocessing procedures were modularized to enable margin analyses under different correlation selections and design-parameter changes. Under downflow conditions, the pressure-drop difference was on the order of ~0.01% and the $T_{w,peak}$ difference was on the order of ~0.01 °C, showing close agreement with TMAP and confirming the applicability of the developed code for downflow preliminary design analyses. In future work, stepwise verification for upflow conditions will be conducted, and the code will be applied to preliminary design and thermal-margin evaluation for research reactor channels.

Acknowledgement

This work was supported by the Nuclear Safety Research Program through the Korea Foundation Of Nuclear Safety(KoFONS) using the financial resource granted by the Nuclear Safety and Security Commission(NSSC) of the Republic of Korea (RS-2025-02643022).

REFERENCES

[1] Daeseong Jo, Jonghark Park, Heetaek Chae, Development of thermal hydraulic and margin analysis code for steady state forced and natural convective cooling of plate type fuel research reactors, Progress in Nuclear Energy 71 (2014).
[2] Hyung Min Son, Kiwon Song, Jonghark Park, Core Thermal Hydraulic Characteristics of Open Pool Type Research Reactors, Transactions of the Korean Nuclear Society Virtual Spring Meeting, May 13–14, 2021.
[3] Eckert, E.R.G., Irvine, Jr., T.F., 1957. Incompressible friction factor, transition and hydrodynamic entrance length studies of ducts with triangular and rectangular cross sections. WADC Tech. Report.
[4] Dittus, F.W., Boelter, L.M.K., 1930. Heat Transfer in Automobile Radiator of the Tubular Type, vol. 2. University of California at Berkeley
[5] Sudo, Y., Miyata, K., Ikawa, H., Ohkawara, M., Kaminaga, M., 1985b. Experimental study of differences in single-phase forced-convection heat transfer characteristics between upflow and downflow for narrow rectangular channel. J. Nucl. Technol. 22
[6] Kaminaga, M., Yamamoto, K., Sudo, Y., 1998. Improvement of critical heat flux correlation for research reactors using plate-type fuel. J. Nucl. Sci. Technol. 35