



OPEN ACCESS

EDITED BY

Longxiang Zhu,
Chongqing University, China

REVIEWED BY

Bin Zhang,
Xi'an Jiaotong University, China
Chuntian Gao,
Northeast Electric Power University, China
Zeyun Wu,
Virginia Commonwealth University,
United States

*CORRESPONDENCE

Yinxiang Ma,
✉ ylfeng0501@163.com

RECEIVED 03 November 2023

ACCEPTED 27 March 2024

PUBLISHED 09 May 2024

CITATION

Huang J, Li J and Ma Y (2024), Development of a fully implicit ODE-solver for containment analysis code.

Front. Energy Res. 12:1332476.

doi: 10.3389/fenrg.2024.1332476

COPYRIGHT

© 2024 Huang, Li and Ma. This is an open-access article distributed under the terms of the [Creative Commons Attribution License \(CC BY\)](https://creativecommons.org/licenses/by/4.0/). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.

Development of a fully implicit ODE-solver for containment analysis code

Jun Huang, Jinggang Li and Yinxiang Ma*

China Nuclear Power Technology Research Institute Co., Ltd., Shen Zhen, China

The thermal–hydraulic dynamics in containment are governed by a system of stiff ordinary differential equations (ODEs). A fully implicit discretization scheme is adopted to discretize these ODEs in order to mitigate the effects of stiffness. In comparison with explicit or semi-implicit discretization schemes that are subject to Courant limits on time steps, the fully implicit discretization scheme is more suitable for a containment analysis code that focuses on predicting both short-term and long-term thermal–hydraulic parameters after an accident. This study introduces a general-purpose ODE solver for the containment analysis code. The outline of the solver is as follows: The fully implicit discrete equations lead to a large set of nonlinear equations that need to be solved using Newton's iterative method. The partial derivative components in the Jacobi matrix are calculated by the perturbation method using finite difference approximation, which avoids the complicated derivation of partial derivatives. The scaling modification technique is incorporated into this ODE solver to deal with significant differences in unknown variable magnitudes, and the line search method is introduced to address the difficulty of obtaining an accurate root estimate with Newton's method when the initial guess is far from the actual root. This proposed ODE solver was applied to two typical stiff ODE problems to test its stiffness-suppressed ability and to demonstrate that this proposed solver can perform calculations with a very large time step. Then, the CASSIA code, a containment analysis code developed by China Nuclear Power Technology Research Institute Co., Ltd (CNPRI), equipped with this ODE solver, was applied to the CSNI (Committee on the Safety of Nuclear Installations) benchmark problem and the Carolinas Virginia Tube Reactor (CVTR) test 3 problem to preliminarily demonstrate that the proposed ODE solver can perform containment thermal–hydraulic analysis correctly. This study could provide references for the development of a home-made containment analysis code.

KEYWORDS

ODE solver, Newton's iterative method, perturbation method, line search method, containment thermal hydraulic

1 Introduction

The widespread utilization of nuclear energy has placed a heightened emphasis on nuclear safety, particularly following the Fukushima nuclear power plant accident in Japan (Chen et al., 2018). The containment structure serves as the ultimate safeguard against the release of radioactive fission products into the surrounding environment. Maintaining the structural integrity of the containment system during loss of coolant accidents (LOCA) or main steam line breaks (MSLB) necessitates the control of pressure and temperature within

the structure to ensure that they remain below acceptable limits (Brosche D, 1972). It is crucial to gain a precise understanding of the time- and space-dependent pressure and temperature distribution within a containment to ensure reactor safety. Thus, the development of a reliable mathematical model and corresponding containment thermal–hydraulic code is crucial for accurate prediction of the time- and space-dependent pressure distribution within the containment structure.

The mathematical simulation of thermal–hydraulic dynamics within containment structures involves stiff initial value problems (IVPs) in ordinary differential equations (ODEs), which is shown as follows:

$$\begin{cases} \frac{d\vec{y}}{dt} = \vec{F}(t, \vec{y}) \\ \vec{y}(t_0) = \vec{y}_0. \end{cases} \quad (1)$$

Efficient and accurate numerical methods are essential for solving these types of problems. Conventional ODE solving methods, such as Euler (Butcher, (2008)), explicit Runge–Kutta (Butcher, (2008)), and Adams (Wille, (1998)), are restricted to a very small step size in order to achieve a stable solution. This means that substantial computer time could be required. To address this challenge, an implicit scheme, which is unconditionally stable, is required to allow for larger time steps and maintain stability regardless of the stiffness of the problem.

Consequently, either the semi-implicit method (the convection term is linearized) or the fully implicit method is adopted by the traditional containment thermal–hydraulic analysis code. Vapor, liquid pool, and droplets are simulated in the CONTAIN 2.0 code (Murata et al., 1997). An implicit numerical method was used to calculate flow rates based on pressure and temperature in control volumes, and fluid inertia was neglected in the momentum equation. Complicated 1-D and 3-D conservation equations were considered in the advanced integral containment code, GOTHIC code (George et al., 2001). The semi-implicit method is adopted to solve momentum equations, and then mass and energy conservation equations are discretized by the semi-implicit method and solved by the one-step Newton–Raphson method. The conservation equations in the COCOSYS code (Allelein et al., 2008; O Dell and Wong, 2010) are discretized by the explicit or implicit method, and the Richardson extrapolation method is used to enhance the accuracy of results and improve convergence. Some severe accident analysis codes, like MELCOR (Gauntt et al., 2005), can also be used to analyze containment thermal hydraulic behaviors. Conservation equations in MELCOR were discretized by the semi-implicit method and solved by the fixed-point iteration method.

The velocity in the convection term of the momentum equation is linearized in the CONTAIN code, GOTHIC code, and MELCOR code, which means the time step used in these codes is restricted to the Courant limit, and its value should not exceed a certain value (Cai, 2022). The time step used in the COCOSYS code can reach a large value due to an optional numerical scheme adopted in this code, i.e., the fully implicit method, but the Richardson extrapolation method used to enhance the accuracy of results and improve convergence requires the calculation of additional solutions at different grid resolutions to estimate the error and refine the final solution. This process increases the computational

cost of the simulation as it involves running the simulation multiple times with different grid sizes (Florez et al., 2017; Merga and Chemedda, 2021).

The system of differential equations for thermal–hydraulic dynamics is discretized via fully implicit schemes, yielding a set of nonlinear equations. Common methods for solving nonlinear equation systems include Newton’s iteration method (Press et al., 2007), the fixed-point iteration method (Hoffman and Frankel, 2001), and the trust-region method (Yuan, 2000). Considering the rapid convergence and effectiveness of Newton’s iteration method, this study adopts it for solving the nonlinear equation system resulting from fully implicit discretization. Consequently, in the present work, a Newton-based fully implicit ODE solver with a convergence-enhanced method is proposed for the containment analysis code CASSIA, which has been developed by China Nuclear Power Technology Research Institute Co., Ltd. (CNPRI).

Section 2 introduces the governing equations of the CASSIA code, necessary closure models, a discretization scheme, and a Newton-based solution procedure. Section 3 shows the application of this proposed ODE solver to two typical stiff ODE problems to test its stiffness-suppressed ability and to demonstrate that this proposed solver can perform calculations with a very large time step. Then, Section 4 shows the application of the CASSIA code equipped with this ODE solver to the CNSI benchmark problem and the Carolinas Virginia Tube Reactor (CVTR) test 3 problem to preliminarily demonstrate that the proposed ODE solver can perform containment thermal–hydraulic analysis correctly.

2 Overview of a fully implicit ODE solver for containment code

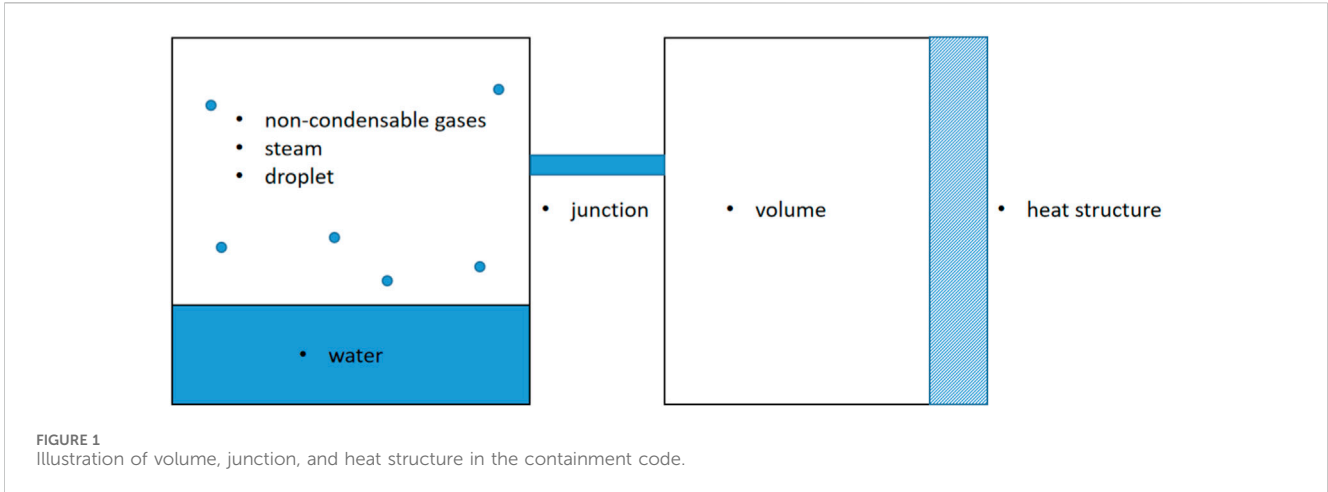
2.1 Governing equations for containment analysis code

2.1.1 Basic conservation equations

The basic computational components in the containment analysis code are control volume, junction, and heat structure. They are used to model the compartments, connections between compartments, and compartment walls in containment, respectively. These computational components are illustrated in Figure 1; the left component represents a control volume with a pool, while the right component represents a control volume without a pool. The junction connects these two control volumes. On the right side of the right control volume, there is a heat structure component used to represent heat transfer between this control volume and another control volume or with the environment. The sump water and gas mixture in a control volume is based on the thermal non-equilibrium assumption, which allows them to have different temperatures. The droplet within the containment compartment is conveyed through the gas junction, taking into account the thermal–hydraulic state (saturated or superheated) of the target volume. If the target volume becomes superheated, the droplet undergoes evaporation.

The basic governing equations for containment thermal–hydraulic calculations are shown as follows (if droplets exist, the governing equations for droplets are also needed):

The mass conservation equations are as follows:



$$\frac{dM_{steam}}{dt} = G_{steam}, \tag{2}$$

$$\frac{dM_i}{dt} = G_i \quad (i = 1, \dots, m), \tag{3}$$

$$\frac{dM_{water}}{dt} = G_{water}. \tag{4}$$

Eqs (2)–(4) delineate the mass conservation of the steam component (denoted by the subscript “steam”) and all m types of non-condensable gas components (denoted by the subscript “ i ”) within the upper gas phase sub-volume. The subscript “water” signifies the water component within the lower liquid phase sub-volume. Furthermore, M represents mass, while G denotes the mass flow rate.

The energy conservation equations are as follows:

$$\frac{d}{dt} \left(\sum_{i=1}^m M_i h_i + M_{steam} h_{steam} \right) - V_{gas} \tag{5}$$

$$\frac{dp}{dt} = \sum_k \left(\sum_{j=1}^{N_{in}} G_{k,j} h_{i,j} - \sum_{j=1}^{N_{out}} G_{k,j} h_{k,j} \right) + \dot{Q}_{gas,in} - \dot{Q}_{gas,out},$$

$$\begin{aligned} & \frac{d}{dt} (M_{water} h_{water}) - V_{water} \frac{dp}{dt} \\ & = \sum (G_{water,in} h_{water,in} - G_{water,out} h_{water,out}) \\ & + \dot{Q}_{water,in} - \dot{Q}_{water,out}. \end{aligned} \tag{6}$$

Eqs (5) and (6) describe the conservation of energy within the gas phase sub-volume (denoted by the subscript “gas”) and the liquid phase sub-volume (denoted by the subscript “water”). In Eq (5), the first term on the left-hand side represents the change rate of total energy in the gas phase sub-volume, while the second term represents the work performed by volume expansion. On the right-hand side of the equation, the first term accounts for the energy exchange due to the flow of all gas components, where the subscript “in” denotes inflow to the control volume and “out” denotes outflow from the control volume. The summation variable “ k ” encompasses all gas components, while “ N_{in} ” and “ N_{out} ” represent the number of fluids flowing into and out of the control volume, respectively. The term “ \dot{Q} ” denotes energy exchange, such as evaporation and condensation at the gas–liquid interface, and energy transfer into

and out of the control volume via heat structures. Eq. (6) is similar to Eq. (5), and its details are omitted here for brevity. In addition, h is the specific enthalpy.

There are two different kinds of junctions (gas junctions and water junctions), considering the characteristics of flow inside the containment. Taking a gas junction as an example, the mass flow rate is calculated by the following equations:

$$\frac{d\vec{G}}{dt} = \frac{A}{l} \left[\rho g (el_{out} - el_{in}) + (p_{out} - p_{in}) - \xi \frac{1}{2\rho A^2} \vec{G} |\vec{G}| \right], \tag{7}$$

$$\rho = \frac{1}{2} (\rho_{out} + \rho_{in}), \tag{8}$$

$$\xi = \xi_0 + \xi_a, \tag{9}$$

where A represents the cross-sectional area of the junction, l is the length of the junction, and ρ is the density of the fluid inside the junction, which is calculated as the arithmetic mean of the gas mixture density inside the control volume at the two ends of the junction (as shown in Eq. 8). el_{out} and el_{in} are the elevations of the outlet and inlet of the junctions, respectively; p_{out} and p_{in} are the pressures at the outlet and inlet of the junction, respectively; and ξ is the resistance coefficient of the junction, which is composed of the user-input local resistance coefficient and the frictional resistance coefficient along the junction (calculated by empirical formulas).

The volume conservation equation is as follows:

$$V_{gas} + V_{water} = const. \tag{10}$$

Eq. 10 indicates that the total volume of the control volume remains constant.

Physically, the entire equation system of differential equations from Eqs (2–7) exhibits rapid pressure responses with slower temperature and energy responses, leading to a significant disparity in the time scales of the dynamics, which qualitatively characterizes the system as a stiff equation set. Mathematically, if the condition number of the Jacobian matrix of the discretization equation for the ODEs is excessively large, the equation system can be considered stiff [(Butcher, (2008)); Thohura and Rahman, (2013)]. As an example, the condition number of the Jacobian matrix (the matrix is shown in Table 1) for the initial time step of the CNSI benchmark problem, which is simulated in Section 4.1, is

calculated (the condition number is equal to $4.5737 \cdot 10^{13}$), indicating that the problems to be solved by the containment code constitute a stiff equation set as well.

To mitigate the effects of stiffness, a fully implicit discretization scheme is employed for solving the ordinary differential equations (ODEs) involved in the containment analysis code. In contrast to the explicit or semi-implicit discretization schemes, which are subject to Courant limits on time steps, the fully implicit discretization scheme is better suited for predicting both short-term and long-term thermal-hydraulic parameters following an accident.

2.1.2 Wall heat transfer equations

The heat transfer between containment walls and the gas mixture in the containment is taken into account by heat conduction through the wall, which is modeled by a heat structure as shown in Figure 1. The one-dimensional heat conduction model is adopted here. Although the three-dimensional heat conduction model considering the axial and circumferential temperature variations of the containment walls and internal heat sink is more accurate, only the heat conduction in the direction perpendicular to the containment wall (i.e., the radial direction) is considered here due to the axial temperature and circumferential gradients being minimal compared with radial gradients. The code uses the finite difference method to solve the heat conduction problem.

The one-dimensional heat conduction equation (as mentioned above, the radial direction is the direction of heat conduction) is presented as follows:

$$\rho C_p \frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial x^2}, \tag{11}$$

where k is the thermal conductivity, t is the time, T is the temperature, and ρC_p is the volumetric heat capacity.

Within the heat structure model in the CASSIA code, diverse boundary conditions can be simulated, encompassing adiabatic heat flow calculated by different heat transfer coefficient correlations, tables of surface temperature *versus* time, heat transfer rate *versus* time, and heat transfer coefficient *versus* time or surface temperature.

The boundary condition can be expressed in general as follows:

$$A(T)T(t) + B(T) \frac{\partial T(T)}{\partial n} = D(T, t). \tag{12}$$

The symbol n represents the unit normal vector oriented away from the boundary surface. Therefore, the prescribed boundary condition dictates that the heat flux leaving the surface equals a heat transfer coefficient, h , multiplied by the temperature difference between the surface temperature, T , and the temperature of the sink, T_{sk} , that is,

$$-k \frac{\partial T}{\partial n} = h(T - T_{sk}), \tag{13}$$

where $A = h, B = k, D = hT_{sk}$.

Additional constitutive models are required to solve the above conservation equations. These models include physical properties, heat transfer between the water pool and gas mixture, and wall drag/heat transfer models.

2.1.3 Physical properties

The physical properties of water and steam come from WSPIF-97 (Cooper and Dooley, (2007)), and the relationships among physical properties are presented as follows:

$$\rho_g = \rho_g(P, T_g), \tag{14}$$

$$\rho_f = \rho_f(P, T_f), \tag{15}$$

$$T^s = T^s(P). \tag{16}$$

The non-condensable gases are treated as ideal gases, so the physical properties of ideal gases come from the ideal gas equation of state. The partial derivatives of physical properties are derived based on the state of the equations and Maxwell correlation (Moran and Shapiro, (1992)).

The total gas pressure is calculated by Dalton's law as follows:

$$P = P_{steam} + \sum_{i=1}^m P_i = P_{steam} + \sum_{i=1}^m \frac{R_{g,i} T_g}{V_g}. \tag{17}$$

2.1.4 Wall-gas heat transfer model

The wall heat transfer coefficient is calculated by the Tagami correlation (Tagami, 1965) during the blowdown phase and by the Uchida correlation (Uchida et al., 1964) during the following heat transfer coefficient decrease phase. The change from the Tagami correlation to the Uchida correlation is realized with an exponential function.

The equation for the maximum heat transfer at the end of the blowdown phase is as follows:

$$h_{max} = 0.7858 \cdot \left(\frac{E}{t_p \cdot V} \right)^{0.6}, \tag{18}$$

where E is the energy-related term, t_p is the blowdown time span, and V is the free volume of the containment.

For the period until the end of the blowdown phase, the heat transfer coefficient is determined by the following equation:

$$h_{g-w} = h_{max} \cdot \sqrt{\frac{t}{t_p}}. \tag{19}$$

The heat transfer coefficient for the following period in the containment volume, according to Uchida is as follows:

$$h_{ep} = 11.362 + 284.05 \cdot \min \left(\frac{m_{steam}}{\sum_{i=1}^n m_i}, 5 \right). \tag{20}$$

The heat transfer coefficient calculated by the Uchida correlation is strongly related to the ratio of steam mass to the mass of non-condensable gases.

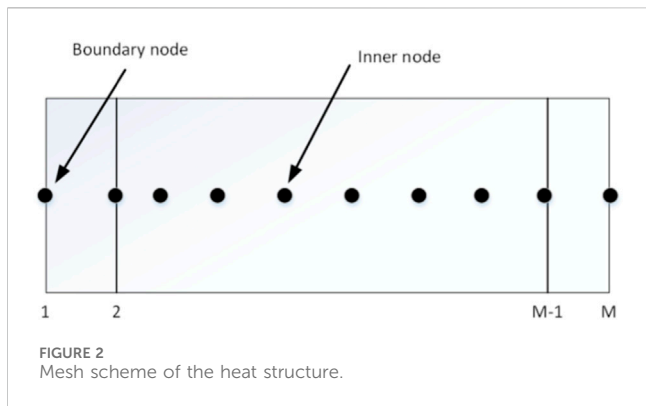
The connection between the Tagami correlation and the Uchida correlation is shown as follows:

$$h_{g-w} = h_{ep} + (h_{max} - h_{ep}) e^{-0.05(t-t_p)}. \tag{21}$$

2.1.5 Discretization scheme

The discretization scheme can be expressed as follows:

The mass conservation equations are as follows:



$$\frac{M_{steam}^{n+1} - M_{steam}^n}{\Delta t} = G_{steam}^{n+1}, \tag{22}$$

$$\frac{M_i^{n+1} - M_i^n}{\Delta t} = G_i^{n+1} \quad (i = 1, \dots, m), \tag{23}$$

$$\frac{M_{water}^{n+1} - M_{water}^n}{\Delta t} = G_{water}^{n+1}. \tag{24}$$

The energy conservation equations are as follows:

$$\begin{aligned} & \frac{(M_i^{n+1} h_i^{n+1} + M_{steam}^{n+1} h_{steam}^{n+1}) - (M_i^n h_i^n + M_{steam}^n h_{steam}^n)}{\Delta t} - V_{gas}^{n+1} \frac{p^{n+1} - p^n}{\Delta t} \\ & = \sum (G_{g,in}^{n+1} h_{g,in}^{n+1} - G_{g,out}^{n+1} h_{g,out}^{n+1}) + \dot{Q}_{g,in}^{n+1} - \dot{Q}_{g,out}^{n+1}. \end{aligned} \tag{25}$$

$$\begin{aligned} & \frac{(M_{water}^{n+1} h_{water}^{n+1} + M_{water}^{n+1} h_{water}^{n+1}) - (M_w^n h_w^n + M_w^n h_w^n)}{\Delta t} - V_{water}^{n+1} \frac{p^{n+1} - p^n}{\Delta t} \\ & = \sum (G_{w,in}^{n+1} h_{w,in}^{n+1} - G_{w,out}^{n+1} h_{w,out}^{n+1}) + \dot{Q}_{w,in}^{n+1} - \dot{Q}_{w,out}^{n+1}. \end{aligned} \tag{26}$$

The governing equations for the mass flow rate in gas junctions are as follows:

$$\frac{\bar{G}^{n+1} - \bar{G}^n}{\Delta t} = \frac{A^{n+1}}{l^{n+1}} \left[\rho^{n+1} g^{n+1} (h_{out}^{n+1} - h_{in}^{n+1}) + (P_{out}^{n+1} - P_{in}^{n+1}) - \xi^{n+1} \frac{1}{2\rho^{n+1} (A^{n+1})^2} \bar{G}^{n+1} |\bar{G}^{n+1}| \right]. \tag{27}$$

The wall heat conduction equation is as follows:

Two types of nodes are considered when discretizing the wall conduction equation, namely, boundary nodes and inner nodes, as demonstrated in Figure 2.

Assuming that the thermal conductivity k and volumetric heat capacity ρC_p are constant, the difference equation of the m^{th} inner node can be expressed as (28) and the difference equations of boundary nodes can be expressed as (29) and (30).

$$\rho_m^{n+1} C_{p,m}^{n+1} \frac{T_m^{n+1} - T_m^n}{\Delta t} = k_m^{n+1} \frac{T_{m+1}^{n+1} - 2T_m^{n+1} + T_{m-1}^{n+1}}{(\Delta x)^2}. \tag{28}$$

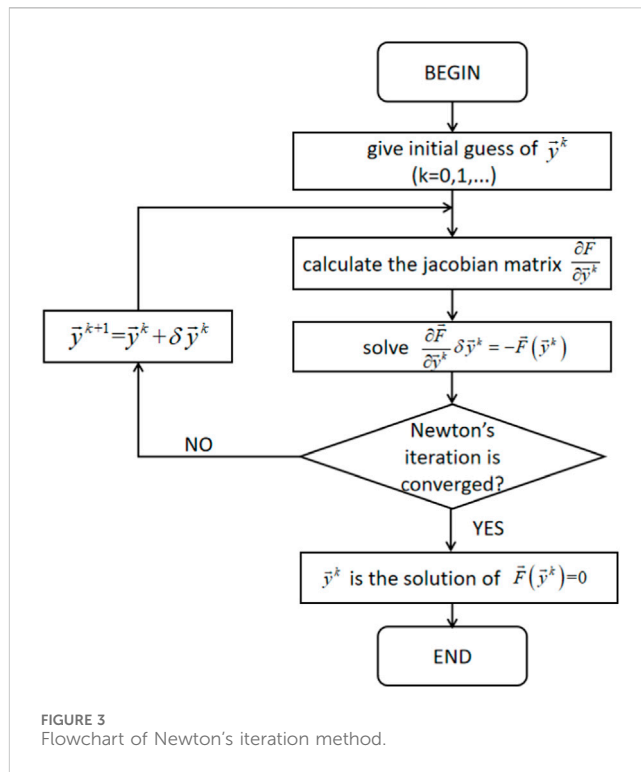
$$\rho_1^{n+1} C_{p,1}^{n+1} \frac{T_1^{n+1} - T_1^n}{\Delta t} = k_1^{n+1} \frac{T_2^{n+1} - 2T_1^{n+1}}{(\Delta x)^2}. \tag{29}$$

$$\rho_M^{n+1} C_{p,M}^{n+1} \frac{T_M^{n+1} - T_M^n}{\Delta t} = k_M^{n+1} \frac{-2T_M^{n+1} + T_{M-1}^{n+1}}{(\Delta x)^2}. \tag{30}$$

2.2 Newton-based solution method

2.2.1 Basic solution procedure

The discrete process described above gives rise to a system of non-linear equations. To solve this system of discretized



equations, the iterative Newton method (Süli and Mayers, 2003) is employed. The resulting set of equations can be expressed as follows:

$$\frac{\bar{y}^{n+1} - \bar{y}^n}{\Delta t} = \bar{F}(t^{n+1}, \bar{y}^{n+1}), \tag{31}$$

where

$$\bar{F}(\bar{y}) = [f_1(\bar{y}), f_2(\bar{y}), \dots, f_n(\bar{y})]^T. \tag{32}$$

$$\bar{y} = [y_1, y_2, \dots, y_n]^T. \tag{33}$$

At the $(k+1)$ -th iteration, it is possible to express the equation using a first-order Taylor expansion as follows:

$$\bar{F}(\bar{y}^{k+1}) \approx \bar{F}(\bar{y}^k) + \frac{\partial \bar{F}}{\partial \bar{y}^k} \delta \bar{y}^k = 0. \tag{34}$$

The linear system of Eq. 35 should be solved at each iteration, where $\frac{\partial \bar{F}}{\partial \bar{y}^k}$ is the Jacobian matrix and $\bar{F}(\bar{y}^k)$ is the vector of residuals.

$$\frac{\partial \bar{F}}{\partial \bar{y}^k} \delta \bar{y}^k = -\bar{F}(\bar{y}^k). \tag{35}$$

The desired $\delta \bar{y}^k$ can be obtained by using the Gaussian elimination method. Then, we can get:

$$\bar{y}^{k+1} = \bar{y}^k + \delta \bar{y}^k. \tag{36}$$

The flow chart of the Newton-based solution method is as demonstrated in Figure 3.

The computational cost and memory for complex problems can be reduced by approximating the Jacobian matrix in Eq. 37 through a finite difference of the residuals, eliminating the need to explicitly construct the Jacobian matrix:

$$\left(\frac{\partial \vec{F}}{\partial \vec{y}^k}\right)_{ij} = \frac{\partial f_i}{\partial y_j} \approx \frac{f_i(y_1, y_2, \dots, y_j + \varepsilon, \dots, y_n) - f_i(y_1, y_2, \dots, y_j, \dots, y_n)}{\varepsilon} \tag{37}$$

The parameter ε is a perturbation parameter that is set to 1×10^{-6} in this study.

2.2.2 Scaling modification technique

As mentioned above, the unknown variables in this study include steam mass, mass flow rate, and temperature. These variables may vary significantly in magnitude under certain operating conditions (for instance, in scenarios where steam content is low, the magnitude of steam mass may be approximately 0.01 or less, while in situations with high pressure differences, the magnitude of the mass flow rate may exceed thousands or more.). Such differences in magnitude among variables can affect the convergence and stability of Newton’s method described in Section 2.2.1. Therefore, special numerical methods are required.

The scaling modification technique is used to improve the convergence and stability of the algorithm. This involves scaling the variables by a certain factor to make their magnitudes more comparable. By applying variable scaling, Newton’s method can more effectively handle variables with different magnitudes, improving the convergence and stability of the algorithm.

Here are the steps to perform variable scaling:

1. Estimate the magnitudes of each variable.
2. Divide each variable by its estimated magnitude.
3. Apply the scaled variables in the iterations of Newton’s method.
4. Restore the final solution to the original variable magnitudes.

By applying the scaling modification technique, Newton’s method can adapt better to variables with different magnitudes, enhancing the efficiency and stability of the algorithm. This technique is particularly useful when dealing with ODEs in the containment analysis code and significant differences in unknown variable magnitudes (mass, mass flow rate, temperature, and some other variables) and can improve the convergence behavior of the method.

2.2.3 Line search method

To avoid divergence caused by the excessively long step size of Newton’s iteration method, the line search method is combined with Newton’s iteration method to improve its efficiency and convergence stability. The main idea of the line search method is to adjust the step size at each iteration to reduce the residual error of nonlinear equations.

In Newton’s method, the algorithm iteratively updates an initial guess for the solution by using the equation:

$$\vec{y}^{k+1} = \vec{y}^k + \delta \vec{y}^k \tag{38}$$

When applying the line search method, instead of using the above equation directly, the step size is scaled by a factor λ as follows:

$$\vec{y}^{k+1} = \vec{y}^k + \lambda \cdot \delta \vec{y}^k \tag{39}$$

The purpose of the line search method is to make Newton’s method more robust and efficient by taking into account the decreased residual as follows:

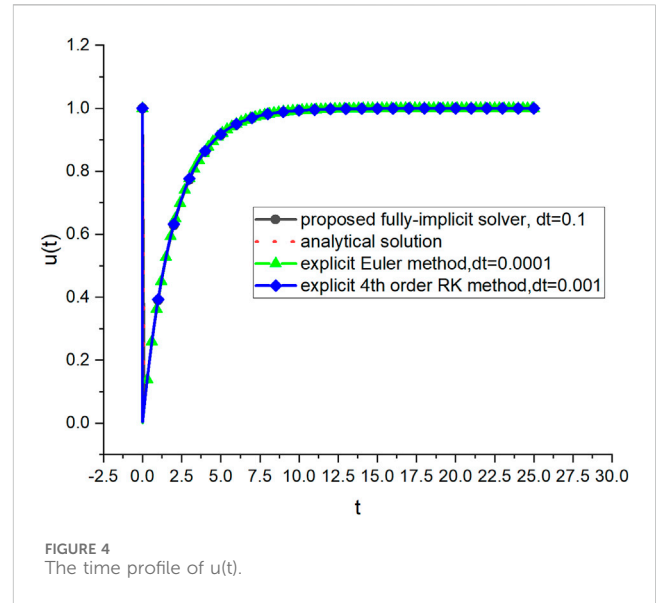


FIGURE 4 The time profile of u(t).

$$\|\vec{F}(\vec{x}^{k+1})\| \leq \|\vec{F}(\vec{x}^k)\| \tag{40}$$

It can improve the convergence rate, especially in cases where the initial guess is far from the solution.

By incorporating the line search method into Newton’s method, the algorithm can provide faster convergence and improved numerical stability, leading to more reliable results.

3 Application to typical stiff ODE problems

In this study, the proposed ODE solver was applied to some typical stiff ODE problems. The results of the computations were compared with the corresponding theoretical solutions or literature-derived values to verify that the solver is capable of solving the stiff ODE problems.

PROBLEM 1: To begin with, we consider an example of a stiff differential equation system that can be analytically solved, which is described in Eq. 41 as follows:

$$\begin{cases} \frac{du}{dt} = -1000.25u + 999.75v + 0.5 \\ \frac{dv}{dt} = 999.75u - 1000.25v + 0.5 \\ u(0) = 1.0 \\ v(0) = -1.0. \end{cases} \tag{41}$$

This set of equations has an analytical solution, as follows:

$$\begin{cases} u(t) = -e^{-0.5t} + e^{-2000t} + 1 \\ v(t) = -e^{-0.5t} - e^{-2000t} + 1. \end{cases} \tag{42}$$

The presence of the transient term e^{-2000t} in the solution causes this system to be stiff. The profiles of u and v calculated by the proposed fully implicit solver, explicit Euler method, and explicit 4th order Runge–Kutta method are compared with the analytical results, which are shown in Figures 4, 5. The figures above illustrate that

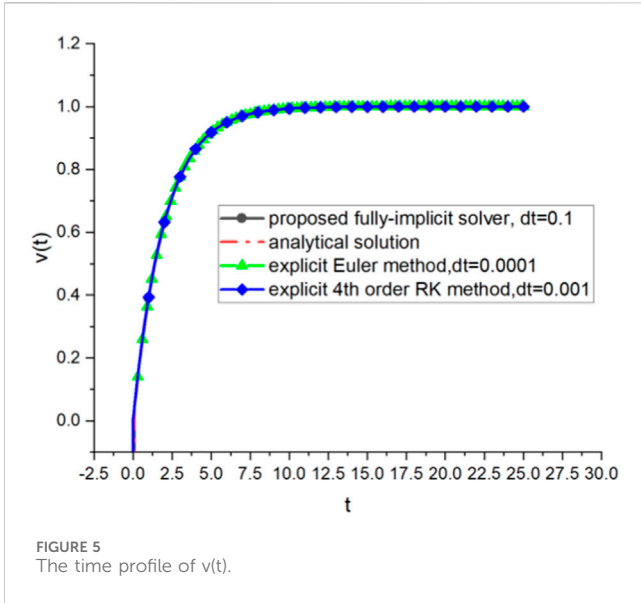


FIGURE 5 The time profile of v(t).

TABLE 2 Overall computational effort comparison among three solving methods.

Method	Time step (s)	Elapsed time* (s)
Explicit Euler method	0.001	0.014382
Explicit 4-th R-K method	0.001	0.0081875
Implicit_Newton method	0.1	0.0022122

these calculated values match the analytical solution very well. However, it is noteworthy that the explicit methods (Euler and RK methods) necessitate significantly smaller time steps compared to the fully implicit method. It is observed that when the time step of the explicit method is as large as that of the fully implicit method, the solutions are divergent. The computational results are illustrated in the figure below:

In addition, three different methods on a PC (13th Gen Intel(R) Core (TM) i9-13900H 2.60 GHz) were used to solve Eq. 41, and the total computational times required for the three methods are shown in Table 2. The result shows that the implicit method requires less overall computational effort compared to explicit methods.

PROBLEM 2: As described in Thohura and Rahman (2013), the following kinetics problem, originally proposed by Robertson, is commonly used as an illustrative instance in this field. The governing equations for this problem are described as follows:

$$\begin{cases} \frac{dy_1}{dt} = -0.04y_1 + 10^4 y_2 y_3 \\ \frac{dy_2}{dt} = 0.04y_1 - 10^4 y_2 y_3 - 3 \cdot 10^7 y_2^2 \\ \frac{dy_3}{dt} = 3 \cdot 10^7 y_2^2 \\ y_1(0) = 1.0 \\ y_2(0) = 0.0 \\ y_3(0) = 0.0. \end{cases} \quad (43)$$

Due to the long time span of this problem ($4 \cdot 10^9 s \approx 126.755 \text{ years}$), the time steps for this problem are variable to finish this calculation, which is shown in Table 3.

TABLE 3 Time steps used in problem 2.

Time span (s)	Time step (s)
0~0.4	0.01
0.4~4.0	0.1
4.0~40	1
40.~400	10
400.~4,000	100
4,000.~ $4 \cdot 10^4$	1,000
$4 \cdot 10^4 \sim 4 \cdot 10^5$	$1 \cdot 10^4$
$4 \cdot 10^5 \sim 4 \cdot 10^6$	$1 \cdot 10^5$
$4 \cdot 10^6 \sim 4 \cdot 10^7$	$1 \cdot 10^6$
$4 \cdot 10^7 \sim 4 \cdot 10^8$	$1 \cdot 10^7$
$4 \cdot 10^8 \sim 4 \cdot 10^9$	$1 \cdot 10^8$

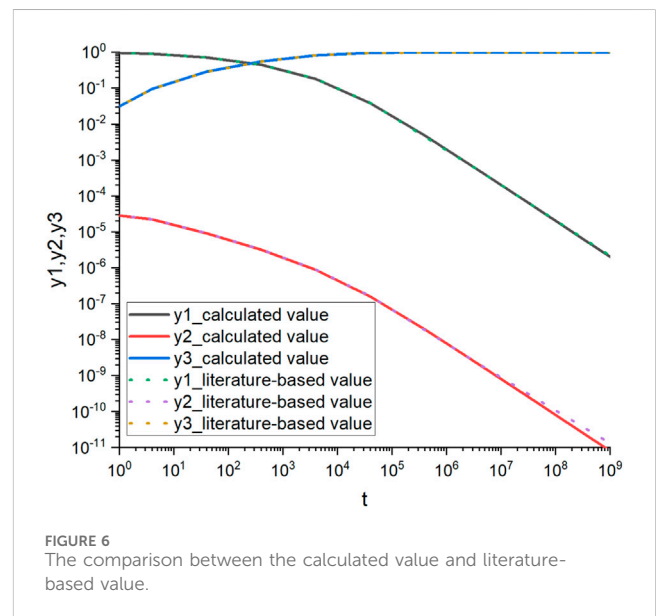


FIGURE 6 The comparison between the calculated value and literature-based value.

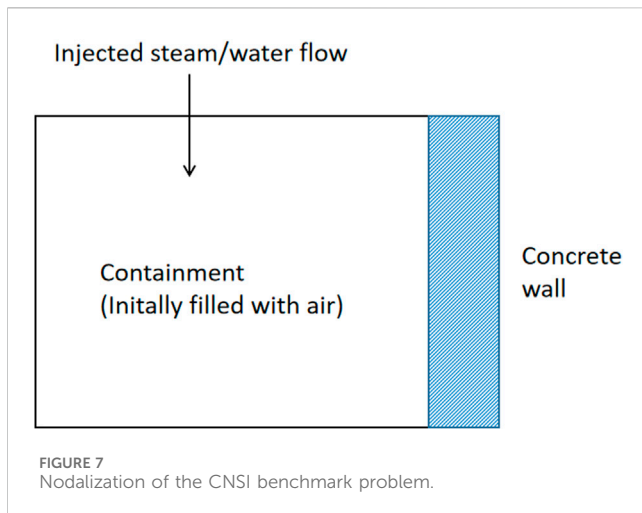
As a result, only the calculated value of the proposed ODE solver and the literature-based value for this problem are compared, as shown in Figure 6. The results show that the calculated values agree well with values in the literature considering the long time span.

Consequently, the capabilities of the proposed ODE solver to solve stiff ODE equations are proven by these two typical stiff ODE equations.

4 Application in the containment T-H analysis

4.1 Application to the CNSI benchmark problem

As described in Haware et al. (1994), the CNSI benchmark problem consists of a single volume with injected steam and water, from which heat is transmitted to a single concrete wall. The



nodalization of this problem is illustrated in Figure 7. The geometrical and thermal hydraulic parameters are listed in Table 4 and the mass flow rate profiles of injected water and steam are shown in Figure 8.

An analytical solution for this benchmark problem was derived in Haware et al. (1994). Comparisons between calculated results and the analytical solution are shown in Figures 9, 10, 11. The total pressure in the containment volume is shown in Figure 9, the partial pressure of the steam is shown in Figure 10, and the wall surface temperature is shown in Figure 11.

The calculated results for all three parameters exhibit strong agreement with the analytical solution, providing evidence of the accuracy of the ODE-solver developed for the containment analysis code presented in this study. This ODE-solver effectively ensures the conservation of mass and energy within the fluid, as well as energy within the structural wall. Furthermore, the comparisons in Figures 9, 10 illustrate that both the calculated total pressure and steam partial pressure slightly exceed the benchmark value. This discrepancy can be attributed to the adoption of thermal non-equilibrium equations

in this study, while the analytical solution is based on a thermal equilibrium model. The good agreement of the predicted wall surface temperature in Figure 11 indicates that the modeling of wall conduction is accurately represented in the analysis. In addition, Figure 9 to Figure 11 demonstrate that a larger time step (1 s) can yield the same results as a smaller time step. This observation substantiates the notion that the fully implicit method can, to a certain extent, alleviate the constraints imposed by the time step.

4.2 Application to the CVTR test 3 problem

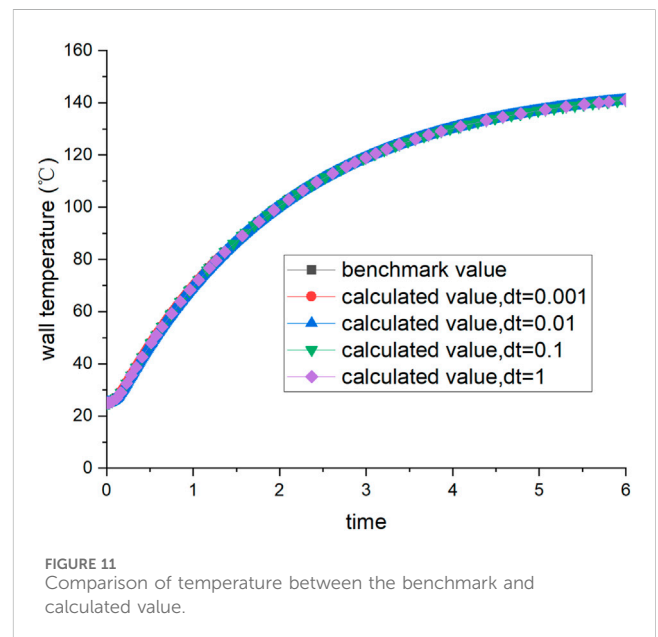
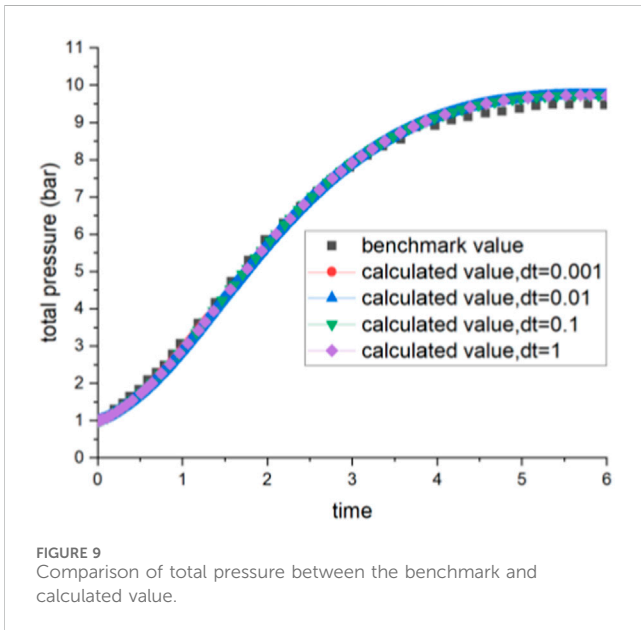
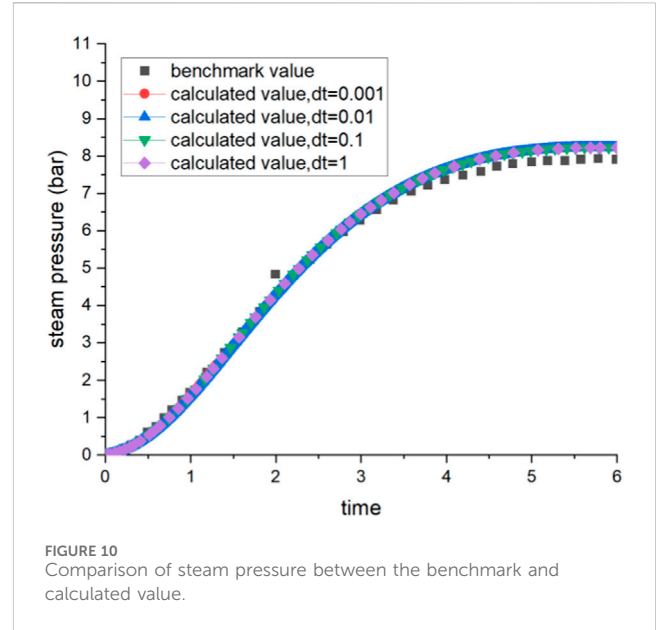
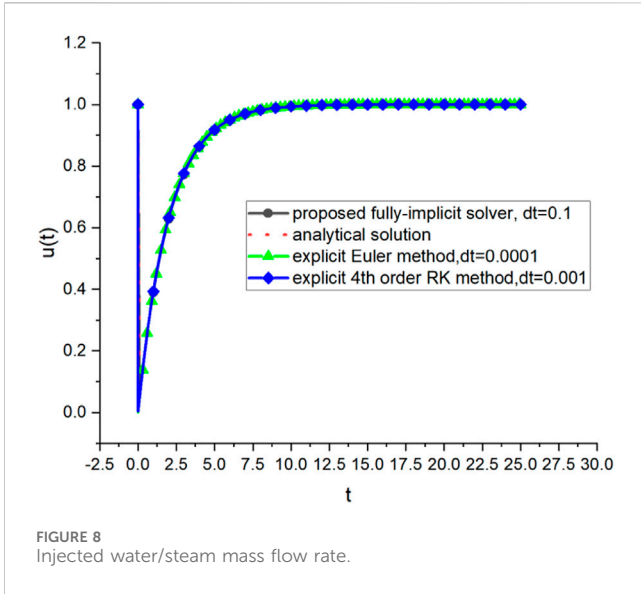
A series of design basis accident (DBA) tests were conducted on the Carolinas Virginia Tube Reactor (CVTR) containment system (Norberg et al., 1969), and the pressure and temperature responses of the containment under steam injection were simulated. The configuration of the CVTR containment system is illustrated in Figure 12. The containment consists of a hemispherical dome and a cylindrical body, with a diameter of 17.68 m inside the cylindrical tube, a total height of approximately 34.75 m, a wall thickness of approximately 0.61 m, and a concrete wall structure, and a total free volume of the containment of approximately 6425.66 m³. Depending on the height and function, the containment can be divided into four zones: the dome, the operation region, the intermediate region, and the basement region.

Test 3 of CVTR, in which the spray system is not activated, is chosen to verify the thermal hydraulic analysis ability of the containment code with the proposed ODE solver. Only two nodes (the dome region and the containment main region) are simulated in this calculation without considering the inner connection of the compartments inside the containment, and the atmosphere environment is also simulated by a node, which is shown in Figure 13.

The initial condition is also described in Figure 13, and the input parameters needed for the inner heat sink and containment wall are listed in Table 5. The mass flow rate and the enthalpy of injected steam are demonstrated in Figure 14.

TABLE 4 Geometrical and thermal hydraulic parameters.

Containment volume	40 m ³
Initial containment pressure	1 bar
Initial containment temperature	25°C
Initial containment relative humidity	0%
Initial wall temperature	25°C
Thermal conductivity of the wall	1.5 W/m · K
Thermal diffusivity of the wall	7.33 × 10 ⁻⁷ m ² /s
Heat transfer coefficient between the containment atmosphere and wall surface	2000 W/m ² · K
Specific heat capacity of air in the containment	718 J/kg · K
Specific enthalpy of input water/steam	1,400 kJ/kg
Containment wall surface area	90 m ²
Containment wall thickness	0.1 m



The comparison between the experimental and calculated values of pressure in the containment under test 3 conditions is shown in Figure 15. It can be seen in Figure 15 that the pressure inside the containment increases rapidly during the steam injection period and decreases slowly after the end of the steam injection, and the CASSIA code equipped with the proposed ODE solver is able to simulate this trend. The peak pressures calculated by the CASSIA code and those of the experimental values appear at basically the same time, but the peak pressure calculated by the CASSIA code is higher than the experimental value. The discrepancy can be attributed to the inadequate consideration of momentum-induced mixing phenomena in the initial pressurization process, as well as the fact that the correlation of the wall heat transfer coefficient used in the CASSIA code is conservative.

5 Conclusion

In this study, a general-purpose, fully implicit ODE solver for containment analysis code is developed.

The ODE solver uses a modified Newton iteration combined with scaling modification techniques and line search methods to improve its convergence and stability. The partial derivative components in the Jacobi matrix are calculated by the perturbation method using the finite difference approximation, which avoids the complicated derivation of partial derivatives.

This solver is employed to validate its efficacy in solving stiff ordinary differential equations (ODEs) by applying it to two representative cases. The calculated results for these two typical stiff ODE problems are subsequently compared against analytical solutions

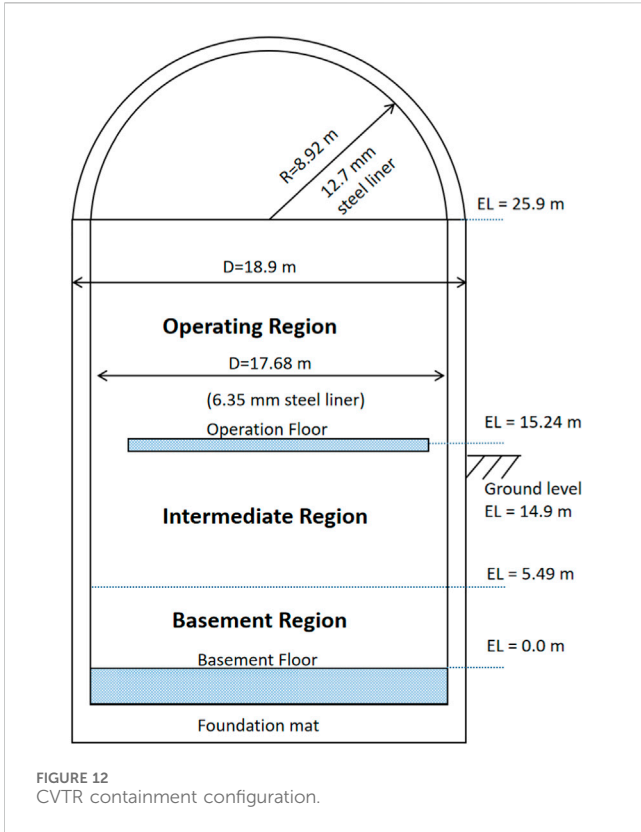


FIGURE 12 CVTR containment configuration.

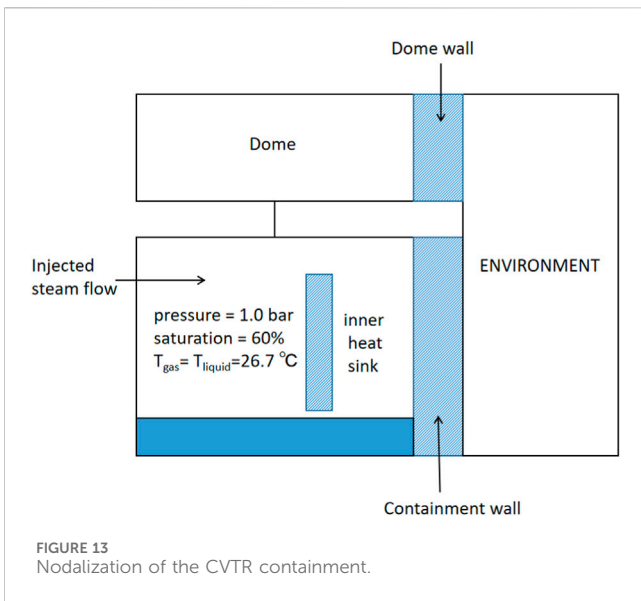


FIGURE 13 Nodalization of the CVTR containment.

and literature-based values. The comparison results not only establish the proficiency of the proposed ODE solver in addressing stiff ODE equations but also illustrate its capability to handle calculations with significantly large time steps, as exemplified by problem 2.

Furthermore, the solver is also employed to assess its competence in conducting thermal-hydraulic analysis within a containment, specifically applied to the CNSI single-volume containment benchmark problem and CVTR test 3 problem. In the CNSI benchmark problem, three essential parameters, namely, total pressure, steam partial pressure, and wall temperature, are calculated. The obtained results for all three

TABLE 5 The characteristic parameters of the heat structures in the CVTR containment.

No.	Heat structure	Material	Area (m ²)	Width(m)
1	Containment wall	Steel	1437.21	0.00635
		Air		0.0016
		Concrete		0.61
2	Dome	Steel	498.89	0.0127
		Air		0.0016
		Concrete		0.5334
3	Inner heat slab	Concrete	1065.23	0.89
		Steel	447.24	0.036

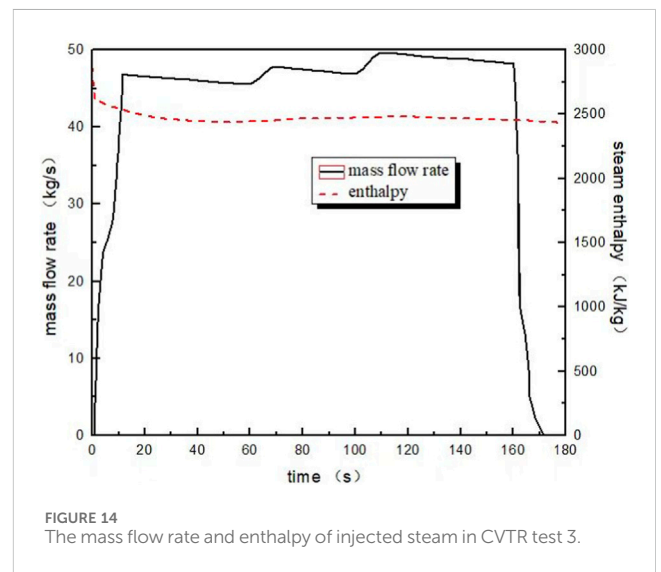


FIGURE 14 The mass flow rate and enthalpy of injected steam in CVTR test 3.

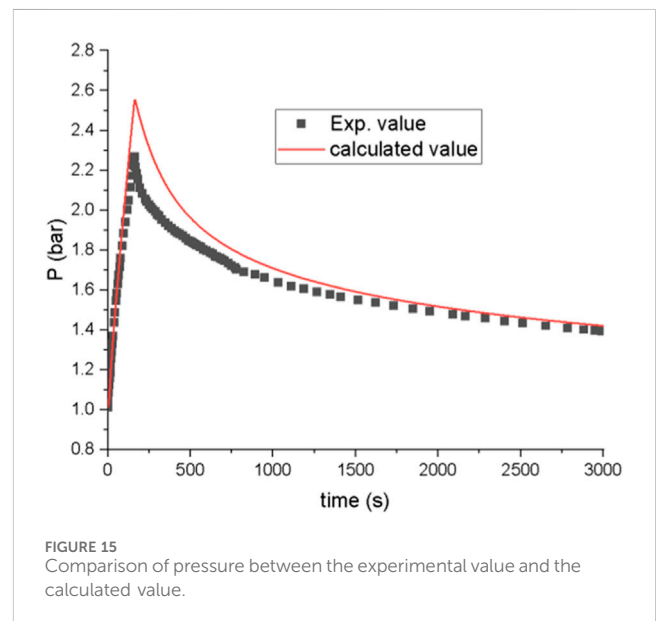


FIGURE 15 Comparison of pressure between the experimental value and the calculated value.

parameters exhibit a close agreement with the benchmark solution, thereby verifying the capability of the CASSIA code, which incorporates the developed ODE solver, to accurately calculate mass and energy conservation as well as wall heat conduction. In the CVTR test 3 problem, the pressure within the containment is compared to the experimental value. The observed trend and the timing of the peak pressure align closely with the experimental data, providing a preliminary assessment of the CASSIA code's capability to conduct thermal hydraulic analysis within a containment.

It is worth noting that the Newton method adopted in the proposed ODE solver involves the computation of the Jacobian matrix, which can be computationally expensive, especially for large systems. Additionally, if the Jacobian needs to be updated in each iteration, it adds to the computational cost. Considering that the equations of the containment code are large and sparse, it may be beneficial to take advantage of the sparsity structure of the Jacobian matrix. Sparse matrix techniques can significantly reduce the computational cost of both storing and computing the Jacobian. This would be the future improvement of the proposed ODE solver.

Data availability statement

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

Author contributions

JH: methodology, software, and writing—original draft. JL: supervision and writing—review and editing. YM: software, validation, and writing—review and editing.

References

- Allelein, H. J., Arndt, S., Klein-Hessling, W., Schwarz, S., Spengler, C., and Weber, G. (2008). COCOSYS: status of development and validation of the German containment code system. *Nucl. Eng. Des.* 238 (4), 872–889. doi:10.1016/j.nucengdes.2007.08.006
- Bogacki, P., and Shampine, L. F. (1989). A 3(2) pair of Runge–Kutta formulas. *Appl. Math. Lett.* 2, 321–325. doi:10.1016/0893-9659(89)90079-7
- Brosche, D. (1972). ZOCO V, a computer code for the calculation of time- and space-dependent pressure distributions in reactor containments. *Nucl. Eng. Des.* 23 (3), 239–271. doi:10.1016/0029-5493(72)90149-5
- Butcher, J. C. (2008). *Numerical methods for ordinary differential equations*. Chichester, UK: Wiley.
- Cai, Q., Fan, K., and Shan, J. (2022). Development and preliminary validation of fully implicit scheme single-phase liquid metal fast reactor code nusol-lmr. *SSRN Electron. J.*, doi:10.2139/ssrn.4077541
- Chen, Y., Wu, Y. W., Wang, M. J., Zhang, Y., Tan, B., Zhang, D., et al. (2018). Development of a multi-compartment containment code for advanced PWR plant. *Nucl. Eng. Des.* 334, 75–89. doi:10.1016/j.nucengdes.2018.05.001
- Cooper, J. R., and Dooley, R. B. (2007). “Revised release on the IAPWS industrial formulation 1997 for the thermodynamic properties of water and steam,” in 18th International Conference on the Properties of Water and Steam, Boulder, Colorado, August, 2007.
- Florez, W. F., Gonzalez, J. W., Hill, A. F., López, J. D., and López, G. J. (2017). Numerical methods coupled with Richardson extrapolation for computation of transient power systems. *Ing. Cienc.* 13 (26), 65–89. doi:10.17230/ingciencia.13.26.3
- Gauntt, R., Cole, R., Erickson, C., Gido, R., Gasser, R., Rodriguez, S., et al. (2005). *MELCOR computer code manuals*. Albuquerque, New Mexico: Sandia National Laboratories, 6119. NUREG/CR.
- George, T. L., Wiles, L., Claybrook, S., Wheeler, C., and McElroy, J. (2001). *Gothic containment analysis package technical manual*. Richland, WA, USA: Numerical Applications Inc.
- Hairer, E., and Wanner, G. (1989). Stiff differential equations solved by Radau methods. *J. Comput. Appl. Math.* 111, 93–111. doi:10.1016/s0377-0427(99)00134-x
- Haware, S. K., Ghosh, A. K., Raj, V. V., and Kakodkar, A. (1994). “Analysis of CSNI benchmark test on containment using the code CONTRAN,” in Proceedings of the third international conference on containment design and operation v2, Canada, 580.
- Hoffman, J. D., and Frankel, S. (2001). “Fixed-point iteration,” in *Numerical methods for engineers and scientists* (New York, NY, USA: CRC Press), 141–145.
- Merga, F. E., and Chemed, H. M. (2021). Modified crank–nicolson scheme with Richardson extrapolation for one-dimensional heat equation. *Iran. J. Sci. Technol. Transaction A, Sci.* 45 (54), 1725–1734. doi:10.1007/s40995-021-01141-0
- Moran, M., and Shapiro, H. (1992). *Fundamentals of engineering thermodynamics*. SERBIULA (sistema librum 2.0). Available at: [http://ftp.demec.ufpr.br/disciplinas/TM104/Termodinamica/Moran-Thermo-Book/Michael%20J.%20Moran,%20Howard%20N.%20Shapiro,%20Daisie%20D.%20Boettner,%20Margaret%20B.%20Bailey%20-%20Fundamentals%20of%20Engineering%20Thermodynamics%20\(2014,%20Wiley\).pdf](http://ftp.demec.ufpr.br/disciplinas/TM104/Termodinamica/Moran-Thermo-Book/Michael%20J.%20Moran,%20Howard%20N.%20Shapiro,%20Daisie%20D.%20Boettner,%20Margaret%20B.%20Bailey%20-%20Fundamentals%20of%20Engineering%20Thermodynamics%20(2014,%20Wiley).pdf)
- Murata, K., Williams, D., Griffith, R., Gido, R., Tadios, E., Davis, F., et al. (1997). *Code manual for CONTAIN 2.0: a computer code for nuclear reactor containment analysis*, Albuquerque, NM, United States: Nuclear Regulatory Commission, Washington, DC (United States). Div. of Systems Technology; Sandia National Labs., Albuquerque, NM (United States); Tills (Jack) and Associates, Inc.
- Norberg, J. A., Bingham, G. E., and Schmitt, R. C. (1969). *Simulated design basis accident tests of the Carolinas Virginia tube reactor containment: preliminary results*. Denton, Texas: UNT Digital Library. doi:10.2172/4753505

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Acknowledgments

The authors would like to express their appreciation for the financial support of CNPRI (China Nuclear Power Technology Research Institute Co., Ltd.).

Conflict of interest

Authors JH, JL, and YM were employed by China Nuclear Power Technology Research Institute Co., Ltd.

The authors declare that this study received funding from China Nuclear Power Technology Research Institute Co., Ltd. The funder had the following involvement in the study: the decision to submit it for publication.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

- O'Dell, C R, and Wong, K (2010). COCOSYS – New Modelling of Safety Relevant Phenomena and Components. *Helv* 111, 846.
- Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical recipes: the art of scientific computing*. Cambridge, United Kingdom: Cambridge University Press.
- Süli, E., and Mayers, D. (2003). *An introduction to numerical analysis*. Cambridge, United Kingdom: Cambridge University Press.
- Tagami, T. (1965). *Interim report on safety assessment and facilities establishment project in Japan for period ending*. NUREG-75/087. National Reactor Testing Station.
- Thohura, S., and Rahman, A. (2013). Numerical approach for solving stiff differential equations: a comparative study. *Glob. J. Sci. Front. Res.*,
- Uchida, H., Oyama, A., and Togo, Y. (1964). *Evaluation of post-incident cooling systems of light-water power reactors*. A/CONF.28/P/436. Tokyo: Tokyo Univ.,
- Wille, D. R. (1998). Experiments in stepsize control for Adams linear multistep methods. *Adv. Comput. Math.* 8 (4), 335–344. doi:10.1023/a:1018960717197
- Yuan, Y. (2000). "A review of trust region algorithms for optimization," in *Iciam 99: proceedings of the fourth international congress on industrial and applied mathematics, edinburgh* (USA: Oxford University Press).

Nomenclature

Acronyms

<i>ODE</i>	Ordinary differential equation
<i>IVP</i>	Initial value problem
<i>LOCA</i>	Loss of coolant accidents
<i>MSLB</i>	Main steam line breaks

English symbols

<i>M</i>	Mass (kg)
<i>C_p</i>	Specific heat (J/(kg·K))
<i>G</i>	Mass flow rate (kg/s)
<i>h</i>	Specific enthalpy (kJ/kg); elevation (m)
\dot{Q}	Heat exchange rate (kJ/s)
<i>k</i>	Thermal conductivity (kW/(m·K))
<i>P</i>	Pressure (MPa)
<i>T</i>	Temperature (K)
<i>A</i>	Cross-section area (m ²)
<i>l</i>	Junction length (m)
<i>V</i>	Volume (m ³)
\vec{F}	Nonlinear equation set vector
\vec{y}	Unknow variables vector

Greek symbols

$[\rho]$	Density (kg/m ³)
ϵ	Perturbation parameter
ξ	Frictional factor
Δt	Time step
Δx	Space step

Subscript

<i>steam</i>	Steam component
<i>water</i>	Water component
<i>i</i>	Non-condensable gas component; the i-th scalar cell
<i>in</i>	Inlet
<i>out</i>	Outlet
<i>g</i>	Vapor phase
<i>f</i>	Liquid water phase
<i>i+1</i>	The (i+1)-th scalar cell

Superscript

<i>n</i>	The n-th time layer
<i>n+1</i>	The (n+1)-th time layer
<i>k</i>	The k-th iteration step
<i>k+1</i>	The (k+1)-th iteration step