An Adaptive Time-Stepping Scheme with Local Convergence Verification Using Support Vector Machines

Kenji Kawaguchi, Jun Ishikawa, and Yu Maruyama

***Abstract*—An adaptive time-stepping scheme in accordance with the local convergence of computation often involves computationally expensive procedures. As a result, many computer simulators have avoided utilizing such an adaptive scheme, while its advantages are well recognized; the scheme not only efficiently allocates computational resources, but also makes the results of the computation more reliable. In this paper, we propose a fast adaptive time-stepping scheme, ATLAS (Adaptive Time-step Learning and Adjusting Scheme), which approximates such an expensive yet beneficial scheme by using support vector machines (SVMs). We demonstrate that ATLAS performs quite favorably when compared with computations without it. ATLAS can incorporate existing solvers and other fast but unreliable adaptive schemes to meet the different criteria required in various applications.**

***Index Terms*—Adaptive time step control, machine learning, ordinary differential equations, severe accident analysis.**

I. Introduction

The numerical time-stepping method plays a vital role in the fields of computation and applied science. It approximately integrates ordinary differential equations (ODEs) and solves initial value problems. The method can be written as

$$y(t_{n+1}) = y(t_n) + h\Phi(Y_n) \tag{1}$$

where h represents the time-step size, $y \in \mathbb{R}^{l\times 1}$ is the dynamical system's state, $Y \in \mathbb{R}^{l\times(n+1)}$ contains the history of y up to the time t_{n+1}, and $\Phi:\mathbb{R}^{l\times(n+1)}\rightarrow\mathbb{R}^{d}$ determines the time stepping scheme. A straightforward choice for function Φ is the derivative of y at t_n, which leads to the explicit Euler's method:

$$\Phi(Y_n) = \frac{dy(t_n)}{dt} = f\left(y(t_n)\right) \tag{2}$$

If f in (2) is Lipschitz continuous in y and continuous in time t, by relying on Banach fixed point theorem, one can show the existence of a unique solution [1]. For such a problem, interval analysis may present the bounds of a numerical solution's error to the exact unique solution [2].

While such rigorous analyses have been conducted for the specific problems one by one [3]-[5], practical interest has been placed on how to decide time-step size h in (1) [6]-[8]. The problem of selecting h can be divided into two parts: how to set tentative h to calculate y_{n+1} and how to accept or reject h after obtaining provisional y_{n+1}. The former issue has been addressed by control theoretic approaches such as proportional-integral (PI) controller or simpler feedback controllers [7]. The objective of the approach is to maintain the local error or the changing rate under a particular tolerance. Also, time-step controlling functions have been arbitrarily defined based on analysts' experiences in order to map dynamical systems' states into particular time-step sizes [9].

On the other hand, the later problem, regarding the criteria of acceptable h, has been dealt with by utilizing constraints imposed by physical modeling properties such as Courant Friedrichs-Lewy condition [10]. Also, absolute or relative changes of systems' states were used as criteria [8], [9]. These two approaches are preferred mostly by computer simulators that require fast calculation since these add only negligible computational cost in general. However, these approaches do not address the errors of computed solutions, and thus no information is given to discuss the accuracy of the computation.

The local error method is computationally more expensive, but it verifies the numerical calculation results to some extent. The method aims to keep estimated local errors within a certain tolerance range, which renders the computed results reliable by assuming local errors are not amplified over time and the estimated errors are close to the exact errors. One way to estimate a local error is to compare the solutions computed by different orders' methods, which is done by adaptive Runge-Kutta methods and Bulirsch-Stoer method. Though, both methods depend on the assumption of high-order differentiability of y, and the latter works well only with smooth functions [11]. Comparing the solutions obtained with different time-step sizes leads more reliable local error estimators if these assumptions are without support. Indeed, as a time-step size gets close to 0, a local error should converge to small values, although confirming this involves a computationally expensive procedure. Therefore, while many computer simulators prefer computationally cheap methods to decide an acceptable h, the simple but expensive method can increase the reliability of the computation.

In this paper, we adopt the support vector machine (SVM) to approximate a reliable but computationally expensive local error method in order to increase efficiency. The proposed scheme obviates the need to calculate the local errors along with the control theoretic approach by maintaining a similar level of accuracy with a much lower computational cost. The paper focuses on the global time-stepping scheme applied to

Manuscript received April 10, 2013; revised July 10, 2013.

K. Kawaguchi, J. Ishikawa, and Y. Maruyama are with the Severe Accident Analysis Research Group, the Nuclear Risk Analysis Research Unit, Japan Atomic Energy Agency, 2-4 Shiragata-Shirane, Tokai-Mura, Ibaraki, Japan (e-mail: kawaguchi.kenji@jaea.go.jp).

DOI: 10.7763/IJET.2014.V6.675

all segments or components of a model. Thus, we do not address the solutions' projection method to maintain synchronization for locally adaptive time-steps (see [12] for a recent review of this literature). In the following, we first propose the new adaptive time-stepping scheme with SVMs, and then demonstrate the applicability of the scheme with a simple thermal-hydraulic model.

II. ATLAS: Adaptive Time-Step Learning and Adjusting Scheme

A. Overview

The suitability of various numerical methods has been studied for decades. Recently, machine learning (ML) was introduced to automatically identify appropriate methods to solve particular problems [13]. The previous study presented ways to use ML to match one numerical method to each problem. In contrast, we take advantage of ML to adaptively use various numerical methods and time-step sizes h within a problem. Due to its characteristics, we call the new proposed scheme *Adaptive Time-step Learning and Adjusting Scheme* (ATLAS).

ATLAS consists of three components: 1) local convergence verification and data generation, 2) offline learning with SVMs, 3) adaptive time-stepping with SVMs. We discuss the details of each component in the following.

B. Local Convergence Verification and Data Generation

When a large complex computer simulator has just been developed, it is unclear if the simulator maintains computational convergence property; whether or not the solutions converge as h gets close to 0. As a result, sensitivity analyses regarding h may be conducted to verify the property and gain insights on acceptable step size h (e.g. see [9]). For adaptive time-stepping schemes, the sensitivities have been studied for values of tolerance instead of h (e.g. see [14]). These sensitivity analyses are to check the global convergence property. However, it should be important to confirm the local convergence property as well. This is because if the local convergence fails, one can easily know which local part of the calculation contains problems. In addition, when local convergence is not achieved, the sensitivity analysis may mistakenly conclude the global convergence, as h and tolerance cannot be very small globally for an entire execution.

The first component of ATLAS verifies the local convergence property of computation, and while doing so, it generates data for the SVM to learn from. The procedure of this component is outlined in Fig. 1. Here, x_i and d_i are the input features and the corresponding target outputs for the SVM. A simple example of convergence criteria used in the *If statement* in Fig. 1 is $||y_r - y_N|| / ||y_N|| < TOL$ (tolerance) for $r = 0$ to $r = N - 1$. In this case, if $N = 1$ and (b) is always selected, this procedure becomes just a control theoretic approach with the local error method (plus data generation). On the other hand, if N is set to be large and the convergence criterion requires an asymptotical convergent behavior for absolute and/or relative errors by considering machine epsilon, the procedure verifies the local convergence property more strongly. As we will discuss shortly, the more reliable the local convergence verification is in this step, the better the accuracy of ATLAS becomes in the end. Thus, the first component of ATLAS is supposed to be adjusted in accordance with the targeted accuracy and speed of ATLAS.

Inputs: N, TOL
Initialization: set t, $y(t)$, i, and LS—Label of Solver
Repeat with i=i+1
 for r=0 **to** N: $\Delta t_r \leftarrow \Delta t_0 \times 0.5^r$
 for r=0 **to** N: $y_r \leftarrow$Integrates ODE with Δt_r (t to $t+\Delta t_0$)
 $x_i \leftarrow [\Delta t_0, y(t)^T, LS]^T$
  If $\{y_0, y_1, \ldots, y_N\}$ meets a convergence criteria
  $d_i \leftarrow +1$
  $y(t+\Delta t_0) \leftarrow y_0$ (or y_N)
  $t \leftarrow t+\Delta t_0$
  Set the next Δt_0 with a control theoretic approach
  $\Delta t_0 \leftarrow \Delta t_0 \times 2$
  else
  $d_i \leftarrow -1$
  Execute the following (a) **OR** (b)
   (a) Adopt a higher order solver & Change LS
   (b) $\Delta t_0 \leftarrow \Delta t_0 \times 0.5$
  end if
 Until (t <= the end calculation time**)**

Fig. 1. Local convergence verification and data generation

Another adjustable feature is the selection of solvers as indicated by (a) in Fig. 1. A natural choice is to use a family of Runge-Kutta methods for Φ in (1) as follows:

$$\Phi(Y_n) = \sum_{i=1}^{s} b_i z_i \tag{3}$$

$$z_i = \begin{cases} f(t_n + c_i h, y_n + \sum_{j=1}^{s-1} a_{ij} z_j) & \text{for explicit methods} \\ f\left(t_n + c_i h, y_n + \sum_{j=1}^{s} a_{ij} z_j\right) & \text{for implicit methods} \end{cases}$$

where a is the Runge-Kutta matrix, b is the weight and c represents the node. Also, s specifies the number of stages and one can find s being equal to the degree of order until fourth-order methods, which contributes to the popularity of the fourth-order Runge-Kutta. The derivation of a, b and c for the fourth-order can be found in [15]. Instead, one can also use a family of multistep methods for Φ. Due to (3), we can now change the order of our methods with a unified framework. A simple criteria to shift into a higher order method is whether Δt_0 is small enough to take advantage of a higher order method such that

$$O\left(s(LS)\Delta t_0^{\,p(LS)}\right) - O\left(s(LS')\Delta t_0^{\,p(LS')}\right) > L \tag{4}$$

where LS and LS' indicate labels of the currently adopted method and the next candidate method respectively. In addition, p returns the number of the method's order, and $L > 0$ is the criteria specified by users.

C. Offline Learning with Support Vector Machines

In the second component of ATLAS, we use the input and desired output pairs (x_i, d_i) obtained above to train and test the SVM. The SVM is a theoretically elegant supervised learning algorithm, yet a user-friendly tool. In this phase of ATLAS, the SVM aims to indirectly solve the primal problem,

$$\min_{w,\xi,b} \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i \tag{5}$$

subject to $d_i(w^T\Psi(x_i)+b) \geq 1-\xi_i$, $\xi_i \geq 0$, $i=1,\ldots,m$ by finding the solution of the following dual problem [16]

$$\max_{\alpha} \quad Q(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j d_i d_j k(x_i,x_j) \tag{6}$$

subject to $0 \leq \alpha_i \leq C$, $i=1,\ldots,m$,

$$\sum_{i=1}^{m}\alpha_i d_i = 0$$

Here, w and b are the adjustable weights, and ξ is the slack variable, which permits the functional margin—the right hand side of the first constrain in (5)—less than 1 at the cost of $C\xi$. Thus, (5) finds b and small w to keep large classification margins while neglecting outliers with ξ. In other words, (5) effectively finds the boundary between those with acceptable Δt and those with non-acceptable Δt. The Ψ in (5) maps x into a higher dimension to have a nonlinear decision boundary, and concretely we choose a kernel function $k(x_i,x_j) := \Psi(x_i)^T\Psi(x_j)$ to do so as in (6). See Haykin's book [16] for the detailed derivation and a rigorous explanation of the SVM in the context of the structural risk minimization based on the VC dimension theory.

We solve (6) with sequential minimal optimization (SMO) algorithm [17]. A practical guide to use SVMs can be found in [18], and there are freely available SVM libraries, such as LIBSVM or Shark. Note that one can replace a control theoretic approach in Fig. 1 by the SVM trained in this step, and repeat *data generation* and *offline learning* phases so that the SVM can be corrected over this repetition process.

D. Adaptive Time-Stepping with Support Vector Machines

After the above two steps, we have a simulator ready to run fast with high accuracy. Concretely, at every time step, we use α learned with (6) to predict acceptable time-step sizes by calculating

$$q(x') = \sum_{i\in\Omega}\alpha_i d_i k(x_i,x') + b \tag{7}$$

where x' is a column vector comprised of a tentative $\Delta t'$, the current systems' state y, and a tentative method's label LS'. We start with a maximum possible $\Delta t'$ and some LS' and decrease $\Delta t'$ and change LS' until q becomes larger than zero. This iterative procedure may take a relatively small computational time since the summation term in (7) is conduced only for a set of support vectors Ω due to Karush-Kuhn- Tucker dual complementarity condition [19]. Therefore, with a low cost, ATLAS adaptively determines time-step sizes and methods in accordance with the specified level of the required local convergence.

III. Experiment

A. Problem Description

We demonstrate the advantages of ATLAS on a problem from the literature where existing adaptive schemes are likely inadequate. Specifically, for the demonstration, we developed a thermal-hydraulic model pictured in Fig. 2, which is a simplified version of *severe accident* analysis models [20]. As illustrated by the simulation codes in the field of severe accident analysis (e.g. MAAP or THALES2 [21]), this type of problem needs very fast computation, and thus adaptive schemes with local error methods are usually not accepted [9], [20]. As a result, the simulation codes have required the extensive use of well experienced analysts' insights in order to properly determine time-step sizes (these simulators ask users to define time-step adjusting functions [9]). Even with such an exhaustive procedure, trial-and-error is inevitable in some cases, and the justification of the time-step sizes being specified in that way is almost solely subjective.

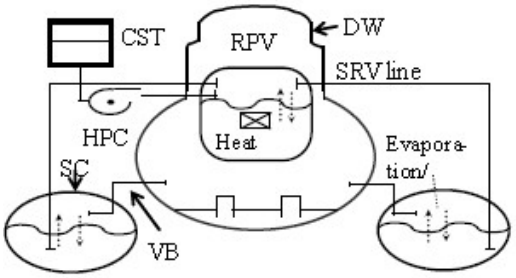


Fig. 2. A thermal-hydraulic model of a powe plant

The meanings of the abbreviations in Fig. 2 are: reactor pressure vessel (RPV), suppression chamber (SC), drywell (DW), condensate storage tank (CST), high pressure coolant injection (HPCI), safety relief valve (SRV), and vacuum breaker (VB). The overall calculation flow is outlined in the following. First, the mass flow through a path or junction (e.g. SRV line) is determined by the orifice flow model

$$\frac{dM(t_n)}{dt} = cA\sqrt{2\rho(t_n)\tilde{P}(t_n)} \tag{8}$$

where c is the orifice constant, A is the flow area, ρ denotes the fluid density, and $\tilde{P}$ defines the pressure difference at the ends of the path. In accordance with the flow computed by (8) and the enthalpy of the flow, the heat H is updated. Then, we separate liquid/vapor phases with certain velocities. Also, heat transfer among the heat source in RPV, water/steam and non-condensation gases occurs. Note that we use the steam table for the water/steam and the ideal gas law for the non-condensation gases in the entire calculation. As a result of these changes, the pressures are updated to maintain the geometric volumes $V_{geometric}$ such that

$$V_{geometric} = V(P,H)_{non-condensation} + \sum_{water,steam} Mv\left(P,\frac{H}{M}\right)$$

Here, $V_{non\text{-}condensation}$ is the sum of the non-condensation gases' volumes, and v is the specific volume for water/steam as the function of P, H and mass M.

The plant data and initial conditions were determined by reference to *the construction permit application forms* of BWR plants in Japan. We increased the heat sources' temperature (^oC) by making it equal to time (s) until 400 s, from where the temperature was fixed to be 400 ^oC. The end calculation time was set to be 1200 s. We let the HPCI inject water into RPV with the amount 378.88 kg/s, when the water level was less than 9.4 m. For the data generation phase of ATLAS, we instead used the injection amount 188.44 kg/s (50% of the base case above). Thus, this experiment demonstrates the efficiency of ATLAS when it is prepared with a different trajectory but in a similar scenario. This is particularly important, since researchers and analysts frequently encounter the need to run simulations in a short amount of time by changing parameter values like this case (i.e. in sensitivity analysis, uncertainty analysis, and accident management support systems [22]). Preparing ATLAS for more general scenarios would require a well-organized sampling of the relevant scenarios' data, which we do not discuss in this paper, leaving it as future work.

B. Experimental Results

We first report the results in the preparation phases of ATLAS. For the case with 188.44 kg/s of HPCI, we conducted *Local Convergence Verification and Data Generation* outlined in Fig. 1 by using the infinity norm of relative errors with TOL = 0.001 and N = 5 as the convergence criteria. Also, we used 1st and 4th order explicit Runge-Kutta methods (we call them "Euler" and "RK4" hereinafter) to be selected with (4), following the convention in the popular computer codes, MAAP and MELCOR. With this setting, the local convergence was confirmed, and 130,208 learning data sets were obtained. In the phase of *offline learning with the SVM*, we divided the data into training data (60%), cross-validation data (20%), and test data (20%). With the training and cross-validation data sets, we conducted grid-search [18] to find appropriate values of the parameters C in (5) and σ in the Gaussian redial basis function (RBF) kernel. As a result, C and σ were set to be 1000 and 1.0 respectively. For the test data sets, the overall accuracy turned out to be 93.11%, but the accuracy to classify positive data sets (i.e. d = 1) was 41.68% (too conservative). Therefore, as stated in the previous section, we used this trained SVM to conduct *Data Generation* phase again so that we could get more positive data sets around where this tentative SVM conservatively outputs too small Δt. Then, we obtained 216,722 data sets in total. With the same parameter values and 30% of 216,722 data sets, we obtained the overall test accuracy of 98.79%: correctly classifying positive pairs with 98.68% and negative sets with 99.84%.

Having ATLAS prepared above, we measured the accuracy and the speed of ATLAS in comparison with those of Euler and RK4. Fig. 3 shows the CPU time versus the number of significant correct digits (denoted by 'scd'), which is a conventional measure of computational accuracy and is defined to be -log10(||relative error||$_\infty$) [23]. The reference solution for scd was computed by the procedure in Fig. 1 with $TOL = 1\times10^{-4}$, $N = 3$ and the minimum $\Delta t = 2^{-9}$. It took 73000 s to compute the reference solution. To vary ATLAS's speed and precision, we scaled Δt determined by ATLAS by using $(\Delta t \times 2^{-u})^{u'}$ as time-step sizes with nine sets of factors u, $u' \geq 0$: u = {0, 0, -0.2, -0.2, -0.4, -0.4, -0.6, -0.6, -0.8} and u' = {1.0, 1.4, 1.7, 1.8, 2.0, 2.2, 2.5, 2.6, 3.0}. As it can be seen in Fig. 3, ATLAS outperformed the original Euler and RK4 without the adaptive scheme. The chief reason why mere Euler and RK4 could not keep up with ATLAS in terms of scd was that the heat contained in non-condensation gases in RPV was miscalculated at several points where smaller time-step sizes were required to obtain the reference solution. Because of this, the increases of their scd seem to stop around 1.4, while their overall accuracies (parameters other than the heat) continued to very slowly improve.

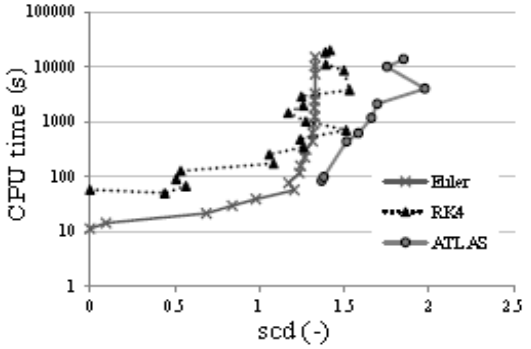


Fig. 3. Cost versus precision with infinity

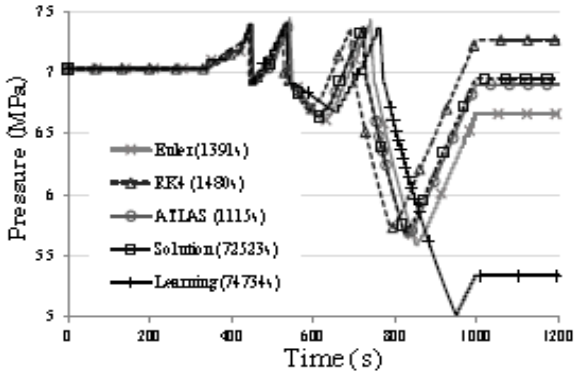


Fig. 4. RPV pressure

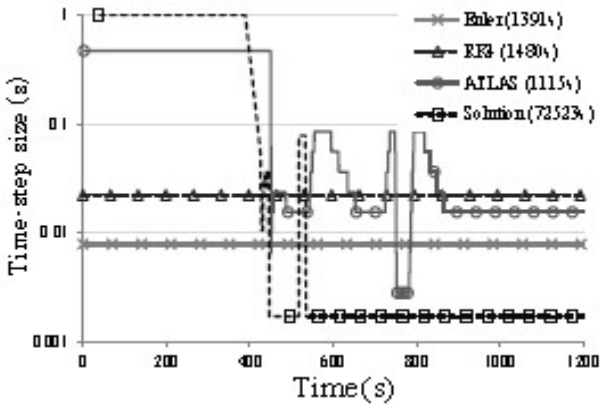


Fig. 5. Time-step sizes used by different schemes

To get a close look at the efficiency of ATLAS, we present the behaviors of RPV pressures (Fig. 4) and time-step sizes (Fig. 5), both of which are from computations at the y-axis ≈ 1200 s in Fig. 3. The calculation time for each scheme is listed in parentheses in the legend. "Solution" and "Learning" in the legend indicate the reference solution and the solution of the scenario used for ATLAS' preparation phases. The result of ATLAS was closest to the reference solution with the lowest cost, even though ATLAS was prepared in a different scenario (Fig. 4). Fig. 5 shows that ATLAS effectively adjusted time-step sizes. Here, ATLAS almost always used RK4 when time-step sizes were less than 0.1. The minimum time-step size used by ATLAS was less than that by the original RK4, but ATLAS used less time for computation due to the proper adjustments of the time-step sizes (Fig. 5). Note that the time-step sizes for the reference solution were restricted by the minimum size $\Delta t = 2^{-9}$ to maintain a feasible computation time, and it used RK4 most of the time.

IV. Conclusion

In this paper, we introduced a novel time-stepping scheme, called ATLAS, with which one can solve the initial value problem with low computational cost and high accuracy. Since ATLAS requires preparation with selected scenarios, an instant application due to this paper's result would be uncertainty analyses and database-driven simulations such as in [22]. However, an underlying idea of ATLAS is that the local errors essentially depend only on local system's states for a simulator, and thereby applying ATLAS to scenarios other than ones used in the preparation is possible. Thus, future work may present the methods of sampling various scenarios' data sets to make ATLAS robust in the state space. Also, future work involves the examination of ATLAS combined with other traditional control theoretic approaches. The combination would be able to take advantages of both types of adaptive schemes.

References

[1] D. M. Marquis, E. Guillaume, A. Camillo, T. Rogaume, and F. Richard, "Existence and uniqueness of solutions of a differential equation system modeling the thermal decomposition of polymer materials," *Combustion and Flame*, vol. 160, pp. 818-829, 2013.

[2] R. E. Moore, *Interval Analysis*, Prentice-Hall, New York, 1996.

[3] J. G. Heywood, W. Nagata and W. Xie, "A numerically based existence theorem for the Navier-Stokes equations," *J. math. Fluid mech*. vol. 1, pp. 5-23, 1999.

[4] W. Tucker, "A rigorous ODE solver and Smales's 14th problem," *Found. Comput. Math.*, vol. 2, pp. 53-117, 2002.

[5] Y. Watanabe, M. Plum, and M.T. Nakao, "A computer-assisted instability proof for the Orr-Sommerfeld problem with Poiseuille flow," *Z. angew. Math. Mech.*, vol. 89, pp. 5-18, 2009.

[6] L. F. Shampine and A. Witt, "A simple step size selection algorithm for ODE codes," *J. Comput. Appl. Math.*, vol. 58, pp. 345-354, 1995.

[7] G. Söderlind, "Time-step selection algorithms: Adaptivity, control and signal processing," *Appl. Numer. Math.*, vol. 56, pp. 488–502, 2006.

[8] S. E. Minkoff and N. M. Kridler, "A comparison of adaptive time stepping methods for coupled flow and deformation modeling," *Appl. Math. Model*, vol. 30, pp. 993-1009, 2006.

[9] R. O. Gauntt, *et al.*, *MELCOR computer code manuals*, version 1.8.5. NUREG/CR-6119, 2001.

[10] G. L. Mesina, "Experimental RELAP5-3D time step improvements," RELAP5 International Users Seminar, presented at the RELAP5 International Users Seminar, Idaho Falls, 2008.

[11] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in Fortran 77: The Art of Scientific Computing*, 2nd ed. Cambridge University Press: New York, 1992, ch. 16.

[12] M. J. Gander and L. Halpern, "Techniques for locally adaptive Timestepping developed over the last two decades," in *Proc. 20th Inter. Conf. Domain Decomposition Methods in Science and Engineering*, 2013, pp. 375-382.

[13] V. Eijkhout and E. Fuentes, "Multi-stage learning of linear algebra algorithms," in *Proc. 7th inter. Conf. Machine Learning and Application*, 2008, pp. 402-407.

[14] G. Söderlind and L. Wang, "Adaptive time-stepping and computational stability," *J. Comput. Appl. Math.*, vol. 185, pp. 225–243, 2006

[15] H. Musa, I. Saidu, and M. Y. Waziri, "A simplified derivation and analysis of fourth order Runge Kutta method," *International Journal of Computer Applications*, vol. 9, no. 8, pp. 51-55, 2010.

[16] S. Haykin, *Neural networks and learning machines*, 3rd ed. New Jersey: PrenticeHall, 2008, ch. 6.

[17] J. Platt, *Sequential minimal optimization: A fast algorithm for training support vector machines*, in Advances in Kernel Methods - Support Vector Learning, MIT Press, 1999, pp. 185-208.

[18] C. W. Hsu, C. C. Chang and C. J. Lin, "A practical guide to support vector classification," Technical report, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, 2003.

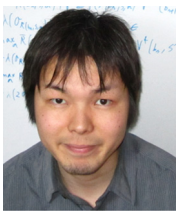
[19] R. Fletcher, *Practical Methods of Optimization*, John Wiley and Sons, Inc., 2nd ed. 1987, ch. 9.

[20] J. V. Dorsselaere, J. S. Lamy, A. Schumm and J. Birchley, "Integral codes for severe accident analyses," in *Nuclear Safety in Light Water Reactors: Severe Accident Phenomenology*, 1st ed. Academic Press, 2012, pp. 625-656.

[21] J. Ishikawa, K. Muramatsu and T. Sakamoto, "Systematic source term analysis for level 3 PSA of a BWR with Mark-II containment with THALES-2 code," in *Proc. 10th Inter. Conf. Nuclear Engineering*, ICONE-10-22080, 2002.

[22] S. Park and K. Ahn, "SAMEX: A severe accident management support expert," *Annals of Nuclear Energy*, vol. 37, pp. 1067-1075, 2010.

[23] N. S. Nedialkov and J. D. Pryce, "Solving differential-algebraic equations by taylor series (I): Computing taylor coefficients," *BIT Numerical Mathematics*, vol. 45, pp. 561-591, 2005.



Kenji Kawaguchi is a research engineer at Japan Atomic Energy Agency on secondment from Computer Simulation and Analysis of Japan. He has written a number of papers regarding computer science and nuclear engineering, some of which are published in refereed journals and conferences He is currently developing severe accident models and also analyzing the phenomenon in the Fukushima nuclear accident.

Jun Ishikawa joined Japan Atomic Energy Research Institute (currently Japan Atomic Energy Agency) in 1996. Since then, he has been engaged in research activities regarding the development of probabilistic risk assessment methods of light water reactors and nuclear fuel facilities. His current main research interests are accident progression and source term evaluations of the Fukushima Daiichi Nuclear Power plants accident.

Yu Maruyama received his Ph.D. in Engineering from the University of Tsukuba, Japan. He joined Japan Atomic Energy Research Institute (currently Japan Atomic Energy Agency) in 1986. He is currently the group leader of Severe Accident Analysis Research Group, Nuclear Safety Research Center, Japan Atomic Energy Agency. His current research interest includes safety assessment of nuclear power plants.