

# A Generalized Adjoint Framework for Sensitivity and Global Error Estimation in Time-Dependent Nuclear Reactor Simulations <sup>1</sup>

H. F. Stripling<sup>a,\*</sup>, M. Anitescu<sup>b</sup>, M. L. Adams<sup>a</sup>

<sup>a</sup>*Nuclear Engineering Department, Texas A&M University, 3133 TAMU, College Station, TX 77843-3133*

<sup>b</sup>*Argonne National Laboratory, Mathematics and Computer Science Division, 9700 S Cass Avenue, Argonne, IL 60439*

---

## Abstract

We develop a general framework for computing the adjoint variable to nuclear engineering problems governed by a set of differential-algebraic equations (DAEs). The nuclear engineering community has a rich history of developing and applying adjoints for sensitivity calculations; many such formulations, however, are specific to a certain set of equations, variables, or solution techniques. Any change or addition to the physics model would require a reformulation of the adjoint problem and substantial difficulties in its software implementation. In this work we propose an abstract framework that allows for the modification and expansion of the governing equations, leverages the existing theory of adjoint formulation for DAEs, and results in adjoint equations that can be used to efficiently compute sensitivities for parametric uncertainty quantification. Moreover, as we justify theoretically and demonstrate numerically, the same framework can be used to estimate global time discretization error.

We first motivate the framework and show that the coupled Bateman and transport equations, which govern the time-dependent neutronic behavior of a nuclear reactor, may be formulated as a DAE system with a power constraint. We then use a variational approach to develop the parameter-dependent adjoint framework and apply existing theory to give formulations for sensitivity and global time discretization error estimates using the adjoint variable. We apply the framework to two problems: a simple pendulum problem with known solution, which serves to verify our global error estimation procedure, and a 1D model of a traveling wave reactor. We then emphasize the utility of the framework by adding heat transfer physics to the reactor problem and showing that no reformulation of the adjoint framework is required for application to the new physics. We conclude that the abstraction of the adjoint approach into a general framework will facilitate multiphysics reactor modeling in large-scale software projects.

*Keywords:* Uncertainty quantification, Error estimation, Sensitivity analysis, Adjoint method, Differential algebraic equation

---

## 1. Introduction

The growing demand for accurate, reliable, and efficient results from scientific computing has fueled an enormous amount of research and advancements in hardware, software, and algorithm technology. As the capabilities and challenges of computing environments improve and change, modelers must revisit and update existing methods, especially as multi-scale, multiphysics, and high-dimensional problems become the norm on the way to improved predictability. Further, the imperative of uncertainty quantification (UQ), verification and validation (V&V), and sensitivity analysis requires substantial ancillary (and often tailored) structure in large software projects.

---

<sup>1</sup>Preprint ANL/MCS-P1963-1011, Mathematics and Computer Science Division, Argonne National Laboratory

\*Corresponding Author, Phone:(979) 845-4161, Fax:(979) 845-6443

*Email addresses:* [h.stripling@tamu.edu](mailto:h.stripling@tamu.edu) (H. F. Stripling), [anitescu@mcs.anl.gov](mailto:anitescu@mcs.anl.gov) (M. Anitescu), [mladams@tamu.edu](mailto:mladams@tamu.edu) (M. L. Adams)

Abstractions, or generalized computational frameworks, are powerful tools for handling the complexity of a multiphysics application. An effective abstraction allows for the addition, removal, or interchange of physics while maintaining a robust yet flexible central programming model. The contribution of this work is an abstraction for computing adjoints of nuclear engineering (NE) systems governed by differential algebraic equations (DAEs). We illustrate the utility of this abstraction by leveraging existing DAE theory to compute sensitivity coefficients, estimate global numerical errors, and couple to new physics without changing the core programming model.

Our application of interest is the simulation of long-life nuclear reactor designs, specifically those that depend on a balance between the production and destruction of fissile material to achieve their target lifetime performance. An accurate model of such a design must properly account for the accumulation of error, such as discretization and model error, and the effects of uncertainty in physical inputs and numerical parameters. These considerations are especially important if the end of life behavior is sensitive to nuclide transmutation at the beginning of life. The size and complexity of these simulations require not only the use of massively parallel computer architectures but also a sound and flexible computational abstraction that simplifies the otherwise overwhelming verification tasks. Therefore, the algorithms that handle uncertainty and error estimation must be sufficiently generic yet must run efficiently at scale, motivating the flexible adjoint framework described in this paper.

The paper is organized as follows: In the remainder of the introduction, we outline the neutron-nuclide field equations, techniques for solving them, and application of their adjoint equations. We then show that the quasi-static solution technique may be formulated as a system of DAEs. In §2, we use a variational approach to develop the general adjoint system for parameter-dependent DAEs and show that the adjoints may be used for sensitivity and error estimation. In §3 we illustrate the framework on a simple pendulum problem that verifies our adjoint use for numerical error estimation. In §4 we apply our adjoint framework to a simplified traveling wave reactor simulation, and in §5 we show that this same framework is easily extended to include a new set of physics governing heat transfer. We finish with some concluding remarks about our adjoint abstraction.

### 1.1. Introduction of the burnup equations

We are interested in modeling the depletion and transmutation of nuclide densities in a neutron field, a key calculation for modeling advanced reactor designs that depend on a careful balance between fertile and fissile species. We develop model in more detail in §4.1; for this introduction it is sufficient to write the equations in operator form. Consider a column vector of  $j$  space and time dependent nuclide densities,  $N = \langle N_1(r, t), N_2(r, t), \dots, N_j(r, t) \rangle$ . The time evolution of the densities follows the Bateman equation

$$\frac{\partial}{\partial t} N(r, t) = \mathbf{M}(\psi(P, t), p) N(r, t), \quad (1)$$

where matrix  $\mathbf{M}$  accounts for nuclide production mechanisms (e.g., transmutation or fission yield) and depletion mechanisms (e.g., neutron absorption or radioactive decay),  $\psi$  is the neutron flux in space-angle-energy phase space  $P$ , and  $p$  is a vector of physical parameters (e.g., decay constants and cross-sections). The neutron flux obeys the time-dependent transport equation

$$\frac{1}{v} \frac{\partial \psi}{\partial t} = \mathbf{P}(N(r, t), p) \psi(P, t) - \mathbf{L}(N(r, t), p) \psi(P, t), \quad (2)$$

with production and loss matrices  $\mathbf{P}$  and  $\mathbf{L}$ , respectively.

Equations (1) and (2) constitute the nonlinear, coupled burnup equations and have been the subject of extensive research and software development for several decades [1, 2, 3]. Deterministic solution schemes most commonly use a variant of the quasi-static approach, which decouples  $\psi$  from  $N$  over a small time interval and makes an assumption for the shape of the flux in time. In early work, the flux was assumed to be constant over the time step [4, 5]; later work assumed a linear shape with various solution techniques (see reviews in [6] and [7]). In any case, the fully nonlinear equations are rarely solved together.

The NE literature is rich with the development and application of adjoint theory to compute sensitivity coefficients for metrics resulting from the solution to the burnup equations. Early work by Williams [8]

formalized the adjoint equations for the constant flux approximation. This led to numerous extensions and generalizations, including the adjoint equation for the constant power depletion problem [9] and other applications [10, 11]. Perhaps most notably, the nuclear power industry makes heavy use of sensitivity information for fuel cycle optimization (see review in [12]), a problem that requires the sensitivity of a small number of scalar metrics with respect to a very large number of uncertain inputs. Incidentally, this is precisely the regime in which adjoint calculations for estimating sensitivities are the most efficient [13].

## 1.2. Formulation of the differential-algebraic burnup equations

To facilitate the general framework, we propose a reformulation of Eqs. (1) and (2) into a set of differential algebraic equations. A system of DAEs differs from a system of ODEs in that the equations are not completely solvable for the derivatives of some subset of the solution vector; instead, this subset satisfies some derivative-free algebraic constraint. We write a DAE system as

$$\begin{aligned} F(\dot{x}, x, p, t) &= \begin{Bmatrix} \dot{x} - f^d(x, p, t) \\ -f^a(x, p, t) \end{Bmatrix} \triangleq \begin{Bmatrix} F^d(\dot{x}, x, p, t) \\ F^a(\dot{x}, x, p, t) \end{Bmatrix} = 0 \\ x(t_0) &= x_0(p) \\ t &\in [t_0, t_f], \end{aligned} \tag{3}$$

where  $x$  is the solution vector and superscripts  $d$  and  $a$  denote its differential and algebraic components, respectively; that is,  $x = (x^d, x^a)$ . If  $f^a(\cdot)$  is uniquely solvable for the algebraic unknowns,  $x^a$ , the DAE is said to be index-1 [14]. In this work, we will handle only index-1 DAEs. We note, however, that any DAE can be reduced by differentiation to an index-1 DAE [14]; therefore, our results can be applied, after transformation, to fairly general DAEs. We also note that this is not the most general form ( $f^d$  or  $f^a$  could depend on  $\dot{x}^d$ ); however, formulation (3) is the most common in engineering applications. Arguably, the notation  $F^a$  is a bit superfluous; nevertheless, we maintain it for a notational ease that will make subsequent computations easy to follow. The index-1 condition translates into  $F_{x^a}^a \equiv \frac{\partial F^a}{\partial x^a}$  being invertible, in  $F_{\dot{x}}^a \equiv \frac{\partial F^a}{\partial \dot{x}} \equiv 0$ , and in  $F_{\dot{x}}^d \equiv \frac{\partial F^d}{\partial \dot{x}} \equiv I$ .

The burnup equations are suitable for this formulation for the following reason: to account for reactor power history and operation, depletion codes must add an engineering degree of freedom and external constraint. These usually take the form of an external absorber and constant power or criticality constraint, respectively [7, 8, 9]. The resulting approximation is that the neutron flux instantaneously satisfies an eigenvalue equation with a normalization constraint,

$$\begin{aligned} \mathbf{L}(N(r, t), p)\psi(P, t) + \mathbf{A}(r, t)\psi(P, t) &= \frac{1}{k_{eff}}\mathbf{P}(N(r, t), p)\psi(P, t), \\ 0 &= P_0 - \left\langle E_f \psi^T \sigma_f N \right\rangle_P, \end{aligned}$$

where  $P_0$  is a target reactor power,  $E_f$  is energy release per fission,  $\sigma_f$  is the microscopic fission cross-section matrix,  $\langle \cdot \rangle_P$  indicates integration over  $P$ , and the matrix  $\mathbf{A}(r, t)$  accounts for engineering control, such as soluble boron or control rods. We note that the only physically realizable flux shape is that which corresponds to  $k_{eff}=1$ , or a critical system [4]; the remainder of our analysis will assume that  $\mathbf{A}(r, t)$  is designed to maintain criticality.

When written with the Bateman equation, these algebraic constraints form our DAE system of interest:

$$\begin{aligned} \frac{\partial N}{\partial t} &= \mathbf{M}(\psi(P, t), p)N(r, t), \\ 0 &= \mathbf{P}(N(r, t), p)\psi(P, t) - \mathbf{L}(N(r, t), p)\psi(P, t) - \mathbf{A}(t)\psi(P, t), \\ 0 &= P_0 - \left\langle E_f \psi^T \sigma_f N \right\rangle_P. \end{aligned} \tag{4}$$

The algebraic constraints take the form of an eigenvalue problem and, with the power constraint, uniquely determine  $\psi$ ; thus, the system is index-1, and we may leverage the rich theory to develop a general adjoint framework for sensitivity and error estimation.

## 2. Adjoint for parameter-dependent DAE systems

In this section we develop the adjoint equations for computing sensitivities to inputs and for estimating the global time-discretization error.

### 2.1. Estimating first-order sensitivity to simulation inputs

Cao, Li, Petzold, and Serban develop the adjoint problem for DAEs up to index-2 in detail in [15]. Here we use a variational approach to produce the form of their result with fully general dependence on the parameters  $p$ . Consider again the forward problem given by Eq. (3), where the initial condition  $x_0(p) = \langle x_0^d(p)^T, x_0^a(p)^T \rangle$  must be consistent with  $f^a(x, p, t_0)$ . For sensitivity analysis, we are typically interested in the sensitivity of some metric,  $I(x(t), p)$ , with respect to the parameters  $p$ . The metric may have a terminal and integrated component:

$$I(x(t), p) = g(x(t_f), p) + \int_{t_0}^{t_f} \mathcal{L}(x, p, t) dt.$$

Because  $F(\cdot) = 0$ , we may introduce the Lagrange multiplier (or adjoint variable)  $\lambda$  and write an *adjointed* metric  $G(\cdot)$ :

$$G(x, p, t) = g(x(t_f), p) + \int_{t_0}^{t_f} \left\{ \mathcal{L}(x, p, t) - \lambda^T F(x, \dot{x}, p, t) \right\} dt.$$

We take its first order sensitivity with respect to  $p$ :

$$\frac{dG}{dp} = [g_p + g_x x_p]_{t=t_f} + \int_{t_0}^{t_f} \left\{ \mathcal{L}_p + \mathcal{L}_x x_p - \lambda^T [F_{\dot{x}} \dot{x}_p + F_x x_p + F_p] \right\} dt,$$

where subscripts of functions indicate Jacobians. After substituting the following result from integration by parts,

$$\int_{t_0}^{t_f} \lambda^T F_{\dot{x}} \dot{x}_p dt = [\lambda^T F_{\dot{x}} x_p]_{t_0}^{t_f} - \int_{t_0}^{t_f} (\lambda^T F_{\dot{x}})' x_p dt,$$

we write the sensitivity equation as

$$\begin{aligned} \frac{dG}{dp} = & \left[ g_p + g_x x_p - \lambda^T F_{\dot{x}} x_p \right]_{t=t_f} + \left[ \lambda^T F_{\dot{x}} x_p \right]_{t=t_0} \\ & + \int_{t_0}^{t_f} \left\{ \mathcal{L}_p - \lambda^T F_p + [\mathcal{L}_x - \lambda^T F_x + (\lambda^T F_{\dot{x}})'] x_p \right\} dt. \end{aligned} \quad (5)$$

The partial derivative  $x_p(t)$  is expensive to compute or store; thus, to eliminate this term from the integral term, we let  $\lambda(t)$  satisfy

$$(\lambda^T F_{\dot{x}})' = \lambda^T F_x - \mathcal{L}_x, \quad (6)$$

which, as we will show, is a DAE for the adjoint variable. Interestingly, this derivation holds irrespective of the index of the DAE, but ensuring that (6) has a solution requires a separate analysis for the different index cases.

For the case of index-1, we find it useful to transpose and expand Eq. (6) in terms of its differential and algebraic components. Noting the block structure of  $F_{\dot{x}}$ , we have

$$\begin{bmatrix} \lambda^d \\ 0 \end{bmatrix}' = \begin{bmatrix} (F_{x^d}^d)^T & (F_{x^d}^a)^T \\ (F_{x^a}^d)^T & (F_{x^a}^a)^T \end{bmatrix} \begin{bmatrix} \lambda^d \\ \lambda^a \end{bmatrix} - \begin{bmatrix} (\mathcal{L}_{x^d})^T \\ (\mathcal{L}_{x^a})^T \end{bmatrix}.$$

The algebraic constraint gives a linear expression for  $\lambda^a$ ,

$$\lambda^a = [(F_{x^a}^a)^T]^{-1}[(\mathcal{L}_{x^a})^T - (F_{x^a}^d)^T \lambda^d], \quad (7)$$

where we have used the fact that for index-1 DAEs,  $F_{x^a}^a$  is nonsingular. The term  $[\lambda^T F_{dx} x_p]_{t=t_0} = (\lambda^d)^T x_p^d \Big|_{t=t_0}$  in Eq. (5) requires a knowledge of the dependence of the differential variable initial condition on the parameters, which is typically given explicitly. The first term in Eq. (5) gives a terminal condition for Eq. (6). We cannot directly eliminate  $x_p$  by setting  $\lambda^T F_{\dot{x}} = g_x$ , since this is inconsistent with both Eq. (7) and the algebraic constraint in Eq. (3). Instead we write  $x_p^a$  in terms of  $x_p^d$  by computing the total derivative of  $F^a$  with  $p$ :

$$\frac{dF^a}{dp} = 0 = F_p^a + F_x^a x_p = F_p^a + F_{x^d}^a x_p^d + F_{x^a}^a x_p^a \quad \Rightarrow \quad x_p^a = -[F_{x^a}^a]^{-1} [F_{x^d}^a x_p^d + F_p^a],$$

where we have used again the index-1 property that  $F_{x^a}^a$  is invertible. Then by expanding  $g_x x_p$  and  $\lambda^T F_{\dot{x}} x_p$

$$\begin{aligned} g_x x_p &= g_{x^d} x_p^d + g_{x^a} x_p^a = \left[ g_{x^d} - g_{x^a} [F_{x^a}^a]^{-1} F_{x^d}^a \right] x_p^d - g_{x^a} [F_{x^a}^a]^{-1} F_p^a \\ \lambda^T F_{\dot{x}} x_p &= (\lambda^d)^T x_p^d, \end{aligned}$$

we equate terms multiplying  $x_p^d$  and arrive at the terminal condition

$$\lambda^d(t_f) = [(g_{x^d})^T - (F_{x^a}^a)^T [(F_{x^a}^a)^T]^{-1} (g_{x^a})^T]_{t=t_f}, \quad (8)$$

which leaves the following sensitivity equation:

$$\frac{dG}{dp} = \left[ (\lambda^d)^T x_p^d \right]_{t=t_0} + \left[ g_p - g_{x^a} [F_{x^a}^a]^{-1} F_p^a \right]_{t=t_f} + \int_{t_0}^{t_f} \left\{ \mathcal{L}_p - \lambda^T F_p \right\} dt. \quad (9)$$

Together, Eqs. (6) and (8) constitute the system of differential algebraic equations governing the adjoint of system (3), and Eq. (9) gives the sensitivity of metric  $I(\cdot)$  with respect to the parameters  $p$ .

## 2.2. Using the adjoint to estimate global time-discretization error

In this section we extend the adjoint-based method developed by Cao and Petzold [16] for estimating the error in a computed metric  $g(\cdot)$  of the final time solution  $x(t_f)$ . The terminal-metric assumption is not restrictive; one can always add a dummy variable to the differential equations to transform a distributed metric into a final one (by adding the equation  $\dot{z} = \mathcal{L}(x, p, t)$ , and defining  $I \equiv g(x(t_f), p) + z(t_f)$ ). For ODEs, Cao and Petzold showed that integrating the adjoint against local truncation error estimates provides an estimate for the global discretization error. For DAEs, we begin by writing the true system governing the unknowns:

$$F(\dot{x}, x, p, t) = 0, \quad x(t_0, p) = x_0(p). \quad (10)$$

Time discretization schemes approximate the temporal derivative in Eq. (10), resulting in an error in the calculation of  $x(t)$ . The system that is actually solved is

$$F(\dot{\tilde{x}}, \tilde{x}, p, t) = r_1(t) \quad \tilde{x}(t_0, p) = x_0(p) + r_2, \quad (11)$$

where  $r_1(t)$  is some systematic perturbation (for example, truncation error), and  $r_2$  is an error in the initial condition. We now define the global error  $e(t) \equiv x(t) - \tilde{x}(t)$  and subtract (11) from (10) to write an exact DAE system that governs  $e(t)$ :

$$F(\dot{x}, x, p, t) - F(\dot{\tilde{x}}, \tilde{x}, p, t) = -r_1(t), \quad e(t_0, p) = -r_2.$$

If we write a first-order expansion of  $F(\dot{x}, x, p, t)$  about  $\tilde{x}$ , we find the following  $\mathcal{O}(\|e^2\|)$ -accurate DAE system for the error evolution:

$$F_{\dot{x}}\dot{e} + F_x e = -r_1(t), \quad e(t_0, p) = -r_2.$$

Again, as in the derivation of the adjoint equation, the preceding statement makes no assumption about the index of the differential equation. Note, however, that well-posedness of this equation might depend on the index. In this work, we are interested only in index-1 formulations. In this case, it will be useful to write this system expanded in terms of its differential and algebraic components. Recalling the block structure of  $F_{\dot{x}}$ , we have

$$\begin{aligned} \dot{e}^d &= -F_{x^d}^d e^d - F_{x^a}^d e^a - r_1^d(t) \\ 0 &= -F_{x^d}^a e^d - F_{x^a}^a e^a - r_1^a(t). \end{aligned} \tag{12}$$

Again, because  $F_{x^a}^a$  is nonsingular, we can write the explicit constraint

$$e^a = [F_{x^a}^a]^{-1} [-F_{x^d}^a e^d - r_1^a]. \tag{13}$$

We also linearize  $g(x(t_f), p)$  about  $\tilde{x}$  to write an  $\mathcal{O}(\|e^2\|)$ -accurate expression for the total error in the scalar terminal metric:

$$\begin{aligned} g(x(t_f), p) - g(\tilde{x}(t_f), p) &\approx \Delta g \triangleq g_{x^d}(\tilde{x}(t_f), p)[x^d(t_f) - \tilde{x}^d(t_f)] + g_{x^a}(\tilde{x}(t_f), p)[x^a(t_f) - \tilde{x}^a(t_f)] \\ &= g_{x^d} e^d(t_f) + g_{x^a} e^a(t_f) \\ &= g_{x^d} e^d(t_f) + g_{x^a} [F_{x^a}^a]^{-1} [-F_{x^d}^a e^d(t_f) - r_1^a(t_f)] \\ &= [g_{x^d} - g_{x^a} [F_{x^a}^a]^{-1} F_{x^d}^a] e^d(t_f) - g_{x^a} [F_{x^a}^a]^{-1} r_1^a(t_f). \end{aligned}$$

To simplify further analysis, we rewrite this estimate in terms of a linear form:

$$\Delta g(x(t_f), p) = l_1^T e^d(t_f) - l_2^T; l_1 \triangleq [g_{x^d} - g_{x^a} [F_{x^a}^a]^{-1} F_{x^d}^a]^T; l_2 \triangleq [-g_{x^a} [F_{x^a}^a]^{-1} r_1^a(t_f)]^T.$$

In the preceding expression, all the  $g$  and  $F$  terms are evaluated at  $(x(t_f), p, t)$ . As a matter of style in the rest of the section we will drop the arguments of  $g$  and  $F$  since the values at which they are evaluated will be clear from the context.

To compute an error estimate of the terminal metric using the preceding equation, we need an estimate for the global error  $e^d$  at  $t = t_f$ . To that end, we proceed with an analysis analogous to that of Cao and Petzold, as follows. We first perform an index reduction by substituting the algebraic constraint (13) into the differential components of system (12). The result is an ODE for the error in the differential variables,  $e^d(t)$ :

$$\begin{aligned} \dot{e}^d &= -F_{x^d}^d e^d - F_{x^a}^d [F_{x^a}^a]^{-1} [-F_{x^d}^a e^d - r_1^a] - r_1^d \\ &= [F_{x^d}^d [F_{x^a}^a]^{-1} F_{x^d}^a - F_{x^d}^d] e^d + F_{x^d}^d [F_{x^a}^a]^{-1} r_1^a - r_1^d, \quad e^d(t_0, p) = r_2^d. \end{aligned} \tag{14}$$

This ODE is an analog to that developed by Cao and Petzold with the exception of additional terms that account for the algebraic constraint on  $e^a(t)$ . At first glance, integrating this equation would seem to require another forward integration along the approximated trajectory  $\tilde{x}(t)$ . By a careful analysis, however, one can obtain an estimate for this error using adjoint differentiation concepts.

To that end, define the resolvent matrix of the error equation,  $\Phi(t)$ , that satisfies

$$\dot{\Phi} = [F_{x^d}^d [F_{x^a}^a]^{-1} F_{x^d}^a - F_{x^d}^d] \Phi, \quad \Phi(0) = I.$$

Then  $e(t_f) = \int_{t_0}^{t_f} \Phi(t_f)\Phi^{-1}(s) \left[ F_{x^a}^d [F_{x^a}^a]^{-1} r_1^a(s) - r_1^d(s) \right] ds + \Phi(t_f)r_2^d$ , and

$$l_1^T e(t_f) = \int_{t_0}^{t_f} l_1^T \Phi(t_f)\Phi^{-1}(s) \left[ F_{x^a}^d [F_{x^a}^a]^{-1} r_1^a(s) - r_1^d(s) \right] ds + l_1^T \Phi(t_f)r_2^d.$$

Now, consider the adjoint equation given by

$$\dot{\lambda} = - \left[ F_{x^a}^d [F_{x^a}^a]^{-1} F_{x^a}^a - F_{x^a}^d \right]^T \lambda, \quad \lambda(t_f) = l_1, \quad (15)$$

the solution to which satisfies  $\lambda^T(s) = l_1^T \Phi(t_f)\Phi^{-1}(s)$  and  $\lambda^T(0) = l_1^T \Phi(t_f)$ . Thus

$$l_1^T e(t_f) = \int_{t_0}^{t_f} \lambda^T(s) \left[ F_{x^a}^d [F_{x^a}^a]^{-1} r_1^a(s) - r_1^d(s) \right] ds + \lambda^T(0)r_2^d.$$

Referring back to our linearized expression for  $\Delta g$ , we now can write our global error estimate in terms of the adjoint that satisfies Eq. (15):

$$\Delta g(x(t_f), p) = \int_{t_0}^{t_f} \lambda^T(s) \left[ F_{x^a}^d [F_{x^a}^a]^{-1} r_1^a(s) - r_1^d(s) \right] ds + \lambda^T(0)r_2^d - g_{x^a} [F_{x^a}^a]^{-1} r_1^a. \quad (16)$$

We show that this adjoint, which leads to the error estimate in  $g(\cdot)$  is the same variable that solves Eqs. (6) and (8) to give the sensitivity of  $g$  with respect to  $p$ . First, we note that the terminal condition  $l_1 = [(g_{x^d})^T - (F_{x^d}^a)^T [(F_{x^a}^a)^T]^{-1} (g_{x^a})^T]_{t=t_f}$  is identical to that given by Eq. (8) in the previous section.

To show that the dynamical equations in (6) and (15) are equivalent, recall that  $\lambda^a = -[(F_{x^a}^a)^T]^{-1} (F_{x^a}^d)^T \lambda^d$ . Substituting into the dynamical equation in system (15), which we now denote as the adjoint for the differential variables, we have

$$\begin{aligned} \dot{\lambda}^d &= - \left[ F_{x^a}^d [F_{x^a}^a]^{-1} F_{x^a}^a - F_{x^a}^d \right]^T \lambda^d = (F_{x^d}^d)^T \lambda^d - (F_{x^d}^a)^T [(F_{x^a}^a)^T]^{-1} (F_{x^a}^d)^T \lambda^d \\ &= (F_{x^d}^d)^T \lambda^d + (F_{x^d}^a)^T \lambda^a. \end{aligned}$$

This result, written along with the algebraic equation for  $\lambda^a$ , is identical to the dynamical equation developed in the previous section. Thus, we find that the same adjoint variable may be used to compute sensitivities of *and* estimate global time discretization errors in a terminal metric of the forward solution.

### 2.3. Truncation estimates for semi-explicit, index-1 DAEs

The previous section provided a method for estimating the global error in some metric  $g(x(t_f))$  resulting from the local error  $r_1(t)$ , which is a consequence of the numerical treatment of the problem. For example, if  $r_1(t)$  is an estimate of the truncation error incurred at each time-step, the theory gives an estimate for the global time discretization error in  $g(\cdot)$ .

A closer inspection of Eq. (11) gives a clearer definition of  $r_1^d(t)$  and  $r_1^a(t)$ :

$$\begin{pmatrix} r_1^d(t) \\ r_1^a(t) \end{pmatrix} = \begin{pmatrix} \dot{\tilde{x}} - f^d(\tilde{x}, p, t) \\ -f^a(\tilde{x}, p, t) \end{pmatrix}.$$

From this we see  $r_1^d(t)$  is the error in estimating the time-derivative of the differential variables, and  $r_1^a(t)$  is the error in solving the algebraic constraints. Along a given differentiable trajectory  $\tilde{x}(t)$ , both residual terms can be computed by evaluating both the right-hand sides of the preceding equation at times  $t$  required by a quadrature approximation to the integral (16). We can then use Hermite interpolation to obtain a differentiable trajectory out of the integration knots  $t_n$ , values  $x_n$ , and derivatives  $f(x_n, p, t)$  [16]. This approach adds a substantial computational burden, however, as it can substantially increase the number of evaluations of  $f$ , the main computational expense of differential integrators.

While one often has no recourse other than to use such an approach, we propose a more lightweight approach for the case of Runge-Kutta methods, based on the concept of embedded Runge-Kutta integration [17]. We demonstrate it on the index-1 semi-explicit case investigated in this work.

The idea of embedded methods is to compute two estimates for the solution at each time step, one of  $\mathcal{O}(p)$  and the other  $\mathcal{O}(p+1)$ . Their difference is an  $\mathcal{O}(p+1)$  estimate of the local truncation error and may be used to guide time-step size for dynamic error control. Here, we also propose to use it for residual estimation.

We use an  $s$ -stage, half-explicit Runge-Kutta method for time step  $t \in [t_{n-1}, t_n]$ :

$$\begin{aligned} x_i^d &= x_{n-1} + h \sum_{j=1}^{i-1} a_{ij} f^d(x_j, p, t_j) \\ 0 &= f^a(x_i, p, t_i) \quad i = 1 \dots s \\ x_n^d &= x_{n-1}^d + h \sum_{i=1}^s b_i f^d(x_i, p, t_i) \\ \hat{x}_n^d &= x_{n-1}^d + h \sum_{i=1}^s \hat{b}_i f^d(x_i, p, t_i) \\ 0 &= f^a(x_n, p, t_n), \end{aligned}$$

where the coefficients  $a_{ij}$ ,  $b_i$ , and  $\hat{b}_i$  define the stage weights, and  $x$  and  $\hat{x}$  are the  $\mathcal{O}(p)$  and order  $\mathcal{O}(p+1)$  solution estimate, respectively. The appeal of the embedded method is that the costly evaluations of  $f$  at each stage are carried out only once, and the different order methods are obtained by using different weights. Note that the half-explicit Runge-Kutta schemes solve for the differential stage vectors explicitly and require a (possibly nonlinear) solution of the algebraic constraint at each stage. The order of accuracy of the scheme for index-1 DAEs is the same as for ODEs [18]. Moreover, the use of half-explicit methods has an additional advantage in our case. We can interpret the  $x^a$  variables as being solved uniquely for any evaluation of  $f$  from the algebraic constraints given the  $x^d$ . We can therefore use a time-filling algorithm (which produces a differentiable solution for all times  $t$  at which to evaluate the residuals) that satisfies the constraints exactly. Thus, we can set  $r_1^a(t) = 0$ .

We note that the difference  $\hat{x} - x$  is an estimate of the truncation in the *solution*, not that in its derivative as required by  $r_1^d(t)$ . The conversion from the former to the latter will depend on  $p$  and the expansion that defines  $a_{ij}$  and  $b_i$ , but in general the error estimate in the derivative will be  $\mathcal{O}(p)$ .

Nevertheless, from Runge-Kutta theory [17], we have that the local residual error for the differential variables satisfies

$$x_{n+1}^d - x^{d,n}(t_n + h) = L_x h^{p+1} + \mathcal{O}(h^{p+2}); \quad \hat{x}_{n+1}^d - x^{d,n}(t_n + h) = \hat{L}_x h^{p+2} + \mathcal{O}(h^{p+3}).$$

Here  $x_{n+1}^d$  is the approximation produced by the order  $p$  method at step  $n+1$ ,  $\hat{x}_{n+1}^d$  is the approximation produced by the order  $p+1$  method at step  $n+1$ , and  $h$  is the time step, whereas  $x^{d,n}$  is the *exact* solution of the DAE when started at  $x_n^d$ . Note that  $L_x$  and  $\hat{L}_x$  are vectors of the same dimension as  $x$ .

Since the preceding relationship is true for any  $h$ , it follows that this relationship also holds for derivatives with time, while dropping an order in  $h$ :

$$\dot{x}_{n+1}^d - \dot{x}^{d,n}(t_n + h) = L_x h^p + \mathcal{O}(h^{p+1}); \quad \dot{\hat{x}}_{n+1}^d - \dot{x}^{d,n}(t_n + h) = \hat{L}_x h^{p+1} + \mathcal{O}(h^{p+2});$$

Subtracting the two and solving for  $L_x$  we obtain

$$L_x = \frac{1}{h^p} \left[ \dot{x}_{n+1}^d - \dot{\hat{x}}_{n+1}^d \right] + \mathcal{O}(h).$$

We can obtain the following expression of the residual on the interval  $[t_n, t_{n+1}]$ :

$$r_1^d(t) = \frac{(t - t_n)^p}{h^p} \left[ \dot{x}_{n+1}^d - \dot{\hat{x}}_{n+1}^d \right] + (t - t_n)^p \mathcal{O}(h) \approx \frac{(t - t_n)^p}{h^p} \left[ \dot{x}_{n+1}^d - \dot{\hat{x}}_{n+1}^d \right] \quad (17)$$

This residual can now be used in conjunction with the integral (16) to estimate the effect of the numerical error on a given scalar terminal estimator function, such as  $g(x(t_f), p)$ .

In Sec. 3 we give details of the implementation for an order 2/3 Runge-Kutta scheme. We note, however, that more work is needed to obtain residual estimates for other schemes based on this approach when we do not have a semi-explicit behavior; in that case  $r_1^a(t)$  will not be 0.

We also note an interesting advantage of our approach. One immediate competitor to our estimator is the “halving” procedure where the same simulation is run with 1/2 of the time step and a Richardson extrapolation estimator is produced. Such an approach, however, would have substantial difficulties with an adaptive time-stepping scheme that varies the time step. Both our approach and the Hermite interpolation approach proposed in [16] have no difficulty dealing with varying time step sizes; indeed all one has to do is evaluate the integral (16).

#### 2.4. Our adjoint-based uncertainty framework

Based on our analysis, we define the following framework for uncertainty analysis of DAEs.

1. We integrate the DAE (3) forward in time. For the index-1 case we use a semi-explicit Runge-Kutta approach defined in §2.3.
2. We produce the estimates of the *local* truncation error using the techniques described in §2.3.
3. We integrate the adjoint equation (6) backwards to produce the adjoint variable.
4. We compute the gradients of the function of interest using the adjoint variable, as described in (9).
5. We use these gradients in a gradient-enhanced uncertainty propagation framework, as described in [19]. While we do not report results for this part in this work, we have defined in past work a process for taking the gradient information in a hybrid approach that accelerates uncertainty quantification but is applicable to nonlinear models (unlike the use of gradients in conjunction with linear models). Such a process uses the gradient information produced by this work.
6. We use the same adjoint variable with the estimate of the local truncation error described above to produce an estimate of the numerical error for a metric of interest by evaluating (16).

We note that this framework can be used to obtain estimates of *global* error even with adaptive methods (which use *local* error estimates to adapt the time step), which cannot be done by classical extrapolation methods. Moreover, the framework can easily support the addition of more physics; all that is needed is the expansion of the definition of the mapping  $F$  in (3). The framework is thus well suited for modern multiphysics simulation environments. After the application of this framework we obtain estimates of parametric sensitivity as propagated through our model, as well as estimates of numerical error.

### 3. Demonstration: The Simple Pendulum

In this section we demonstrate the adjoint framework for sensitivity and global error estimation on a verification problem with a known solution. The equations that govern a simple pendulum of length  $L$  pivoting about the point  $(x = 0, y = 0)$  are

$$F(\dot{z}, z, p, t) = \left\{ \begin{array}{l} \dot{x} - u \\ \dot{u} - \Lambda x \\ x^2 + y^2 - L^2 \\ ux + vy \\ u^2 + v^2 - \gamma y + L^2 \Lambda \end{array} \right\} = 0, \quad \begin{array}{l} x(t_0) = x_0(p) \\ u(t_0) = u_0(p) \end{array},$$

where  $u$  and  $v$  are velocities in the  $x$  and  $y$  directions, respectively,  $\Lambda$  is a Lagrange multiplier resulting from an index reduction, and  $\gamma$  is the local gravitational constant. The unknown vector  $z = \langle z^d, z^a \rangle^T$  and parameters vector  $p$  are

$$z^d = \begin{bmatrix} x \\ u \end{bmatrix}, \quad z^a = \begin{bmatrix} y \\ v \\ \lambda \end{bmatrix}, \quad p = \begin{bmatrix} L \\ \gamma \end{bmatrix}$$

Note that this DAE system is semi-explicit index-1, since the algebraic variables can be solved for explicitly given values of the differential variables. Thus, the modeler need only specify  $z^d(t_0)$  to initialize the system. The solution for the  $x$ -coordinate as a function of time is

$$\begin{aligned} x(t) &= L \sin(\theta(t)) \\ \theta(t) &= \theta_0 \cos\left(\frac{2\pi t}{T}\right) \\ T &= 2\pi \sqrt{\frac{L}{\gamma}} (1 + C_1 \theta_0^2 + C_2 \theta_0^4 + C_3 \theta_0^6 + C_4 \theta_0^8 + \mathcal{O}(\theta_0^8)) \\ C_1 &= \frac{1}{16}, \quad C_2 = \frac{11}{3072}, \quad C_3 = \frac{173}{737280}, \quad C_4 = \frac{22931}{1321205760} \\ \theta_0 &= \sin^{-1}\left(\frac{x_0}{L}\right), \end{aligned}$$

where  $\theta$  is the deflection of the pendulum, in radians, and  $T$  is the period of oscillation. This result may be considered analytic for  $\theta_0^8 \leq \epsilon_{mach}$ , the machine precision. Analytic results for the velocities,  $y$  coordinate, and derivatives of the unknowns with respect to  $p$  follow from the expression for  $x(t)$ .

We solve these equations using an embedded, half-explicit Runge-Kutta 2-3 scheme. The order-2 scheme is a midpoint rule, and the order-3 scheme is Simpson's rule. The extended Butcher tableau is

$$\begin{array}{c|ccc} 0 & 0 & & \\ 1/2 & 1/2 & & \\ 1 & -1 & 2 & \\ \hline \text{order 2} & 0 & 1 & \\ \text{order 3} & 1/6 & 2/3 & 1/6 \end{array} .$$

As described in §2.3, this scheme provides an estimate for the local truncation error,  $r_1^d(t)$ , at each time step.

We define a terminal metric as the  $x$ -coordinate of the pendulum at  $t = t_f$ :

$$g(x(t_f)) = x(t_f).$$

Then, by Eqs. (6) and (8), our adjoint system is

$$\begin{aligned} (F_x^T \lambda)' &= F_x^T \lambda \\ \lambda^d(t_f) &= \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \end{aligned}$$

where

$$F_{\dot{x}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad F_x = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 \\ -\Lambda & 0 & 0 & 0 & x \\ 2x & 0 & 2y & 0 & 0 \\ u & x & v & y & 0 \\ 0 & 2u & -\gamma & 2v & L^2 \end{bmatrix}.$$

From Eq. (9), the sensitivity of our metric with respect to  $p = [L, \gamma]^T$  is

$$\frac{dg}{dp} = - \int_{t_0}^{t_f} \lambda^T F_p dt,$$

where

$$F_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -2L & 0 \\ 0 & 0 \\ 2\Lambda L & -y \end{bmatrix}.$$

We compute this integral by appending an unknown  $g_p$  to the  $\lambda$  vector that satisfies the following ODE:

$$\begin{aligned} \dot{g}_p &= \lambda^T F_p \\ \dot{g}_p(t_f) &= 0. \end{aligned}$$

This integrates the sensitivity equation to the same order of accuracy as the adjoint solve and prevents the need for a quadrature rule. After we have solved for the adjoint, we can compute our estimate of the global time discretization error using Eq. (16) appropriate for this problem:

$$\Delta g = - \int_{t_0}^{t_f} \lambda^T(s) r_1(s) ds.$$

We cannot use the DAE solver to compute this integral, as above, because the integrand is a discontinuous function in time ( $r_1 = 0$  at the beginning of each time step and grows as  $(\Delta t/h)^2$  during the time step). The integral is estimated by a sum over the time steps

$$\Delta g \approx \sum_{i=2}^N \int_0^{h_i} \left[ \frac{\lambda(t_{i-1}) + \lambda(t_i)}{2} \right]^T r_1(t_i) \left( \frac{s}{h_i} \right)^2 ds = \sum_{i=2}^N h_i \left[ \frac{\lambda(t_{i-1}) + \lambda(t_i)}{6} \right]^T r_1(t_i).$$

In this expression we allow for the time step to vary, to point out that it is relatively easy to allow for adaptive time step.

As a numerical example, we solve the pendulum equations with

$$z^d(t_0) = \begin{bmatrix} 1.0 \\ 0.0 \end{bmatrix}, \quad p = \begin{bmatrix} 100.0 \\ 9.8 \end{bmatrix}, \quad t \in [0, 120s].$$

Our solution metric is  $g = x(t_f = 120s)$ . The error in this metric,  $\Delta g$ , is the difference between the known solution and the numerical solution. We also have known expressions for  $\frac{dg}{dL}$ , and  $\frac{dg}{d\gamma}$ , which we can compare with the sensitivities estimated via the adjoint solve. We also can compare our estimate for  $\Delta g$  resulting from the adjoint solve and local truncation estimates to its known value. Figure 1 gives each of these estimates as a function of the solver time step.

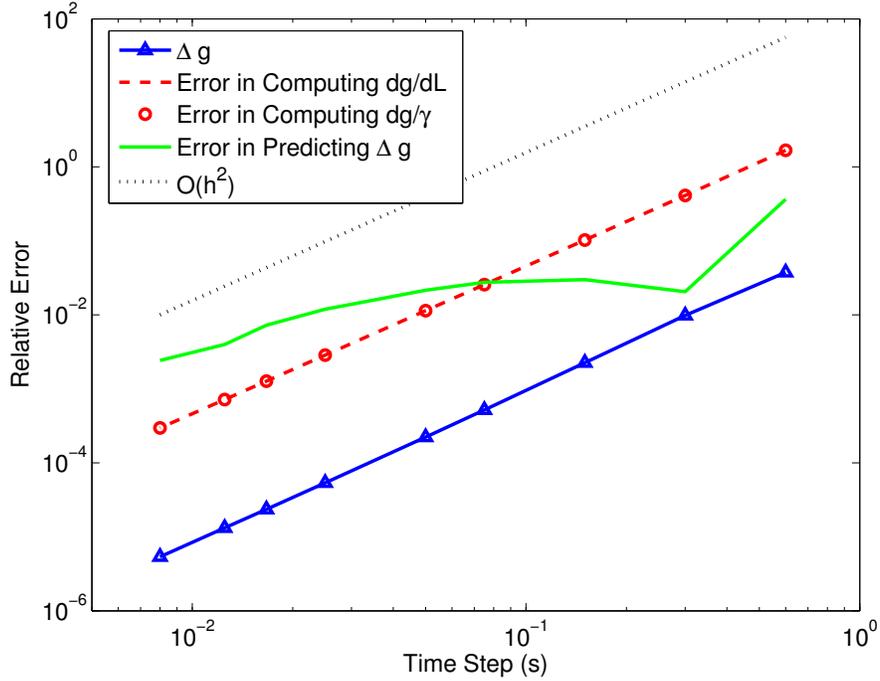


Figure 1: Relative error in computed sensitivities and in the estimate of the global time discretization error

As expected from the second-order scheme, the numerical error  $\Delta g$  decreases as  $\mathcal{O}(h^2)$ . Similarly, the sensitivities resulting from the second-order adjoint solve decrease as  $\mathcal{O}(h^2)$ . The adjoint-based estimate of  $\Delta g$  is a higher-order error estimate, and we do not expect second-order convergence. We do expect, of course, that the predictions become more accurate in the limit as  $h$  decreases, which we see here. We also note that an estimate of numerical error within a relative error of 1% is highly encouraging.

#### 4. A Breeding/Depletion Problem

In this section we develop a relatively simple instance of Eqs. (1) and (2), apply our adjoint framework, and present sensitivity and error estimation results.

##### 4.1. DAE formulation for the burnup equations

Our reactor geometry is a one-dimensional cylinder with radius  $R$ , only one neutron energy group, and axial coordinate  $z \in [0, L]$ . We model the concentration of only three nuclides: a fissile nuclide ( $^{239}\text{Pu}$ ,  $N_9$ ), fertile nuclide ( $^{238}\text{U}$ ,  $N_8$ ), and representative fission product ( $N_0$ ). The following set of simplified Bateman equations governs the nuclide densities

$$\begin{aligned} \frac{\partial N_9(z, p, t)}{\partial t} &= \phi(z, p, t) [\sigma_{a,8} N_8(z, p, t) - \sigma_{a,9} N_9(z, p, t)] \\ \frac{\partial N_8(z, p, t)}{\partial t} &= \phi(z, p, t) [\Gamma \sigma_{a,9} N_9(z, p, t) - \sigma_{a,8} N_8(z, p, t)] \\ \frac{\partial N_0(z, p, t)}{\partial t} &= \phi(z, p, t) [2\sigma_{f,9} N_9(z, p, t) - \sigma_{a,0} N_0(z, p, t)], \end{aligned}$$

where  $\phi \equiv \langle \psi \rangle_{4\pi}$  is the scalar neutron flux,  $\sigma_a$  and  $\sigma_f$  are the absorption and fission microscopic cross-sections, respectively,  $p$  is a yet-undefined vector of the physical and numerical parameters required for the

problem, and  $\Gamma$  is a small positive number to represent the probability of generating a fissile nuclide after a nonfission absorption in  $N_9$ .

To model the neutron flux, we integrate Eq. (2) over all angles and energies to obtain the one-group continuity equation

$$\frac{1}{v} \frac{\partial \phi(z, p, t)}{\partial t} = \left[ \nu \Sigma_f(z, p, t) - \Sigma_a(z, pt) - D(z, p, t) B_g^2 \right] \phi(z, p, t) - \frac{d}{dz} J(z, p, t),$$

where  $J$  is the net neutron current,  $\Sigma_f$ , and  $\Sigma_a$  are the macroscopic cross-sections, that depend on  $N_9$ ,  $N_8$ , and  $N_0$ ,  $D$  is the diffusion coefficient, with a modified definition of  $D = \alpha_D / (3\Sigma_t)$ ,  $DB_g^2$  is a buckling term to account for radial leakage,  $v$  is the neutron speed, and  $\nu$  is the number of neutrons produced per fission..

Our spatial discretization of the Bateman and continuity equation follows a standard cell-centered finite-difference scheme, described as follows. We generate  $n$  equally spaced cells (of width  $\Delta z$ ) in the axial dimension and integrate over each cell. The result in cell  $i$  is

$$\begin{aligned} \frac{\partial N_{9,i}(p, t)}{\partial t} &= \phi_i(p, t) [\sigma_{a,8} N_{8,i}(p, t) - \sigma_{a,9} N_{9,i}(p, t)] \\ \frac{\partial N_{8,i}(p, t)}{\partial t} &= \phi_i(p, t) [\Gamma \sigma_a N_{9,i}(p, t) - \sigma_{a,8} N_{8,i}(p, t)] \\ \frac{\partial N_{0,i}(p, t)}{\partial t} &= \phi_i(p, t) [2\sigma_{f,9} N_{9,i}(p, t) - \sigma_{a,0} N_{0,i}(p, t)] \\ \frac{\Delta z}{v} \frac{\partial \phi_i(p, t)}{\partial t} &= \left[ \nu \Sigma_{f,i}(p, t) - \Sigma_{a,i}(p, t) - D_i(p, t) B_g^2 \right] \Delta z \phi_i(p, t) - J_{i+1/2}(p, t) + J_{i-1/2}(p, t) \end{aligned}$$

where subscript  $i$  indicates a cell average. Our definition of the cell-edge current is such that  $J$  is conserved across the cell boundary. The result of applying Fick's law,  $J = -D\nabla\phi$ , and equating half-cell currents on either side of the interior boundary at  $z_{i+1/2}$  is

$$J_{i+1/2}(p, t) = -\frac{2}{\Delta z} \frac{D_i D_{i+1}}{D_i + D_{i+1}} (\phi_{i+1} - \phi_i) \equiv -\frac{D_{i+1/2}}{\Delta z} (\phi_{i+1} - \phi_i).$$

We use zero re-entrant conditions at the axial boundaries, namely,

$$\begin{aligned} J_{1/2}(p, t) &= -\frac{2D_1}{\Delta z + 4D_1} \phi_1, \\ J_{n+1/2}(p, t) &= \frac{2D_n}{\Delta z + 4D_n} \phi_n. \end{aligned}$$

The nonlinear, spatially discretized burnup equations can each be written in matrix-vector format as follows:

$$\frac{\partial N(p, t)}{\partial t} = \phi(p, t)^T \mathbf{K}(p) N(p, t), \quad N \in \mathbb{R}^{3n} \quad (18)$$

$$\frac{\Delta z}{v} \frac{\partial \phi(p, t)}{\partial t} = \mathbf{M}(N(t), p) \phi(p, t), \quad \phi \in \mathbb{R}^n. \quad (19)$$

As mentioned in the introduction, an external absorber is usually imposed on a reactor in order to control power or reactivity. Our representation for this control variable is the addition of a blanket external absorber with cross-section  $\Sigma_a^{\text{ext}}(t)$  such that the flux shape satisfies the steady state version of Eq. (19). The result is an algebraic constraint on  $\phi$  and  $\Sigma_a^{\text{ext}}$ , which takes the form of an eigenproblem:

$$\mathbf{M}(N(t), p) \phi_i(p, t) - \Sigma_a^{\text{ext}} \phi(p, t) = 0. \quad (20)$$

Finally, a constant power constraint  $(\phi^T N_9) = P_0$  is applied to close the system. In summary, our DAE

model for the burnup equations is

$$\begin{aligned}
\frac{\partial N(p, t)}{\partial t} &= \phi(p, t)^T \mathbf{K}(p) N(p, t) \\
\mathbf{M}(N(t), p) \phi(p, t) - \Sigma_a^{\text{ext}} \phi(p, t) &= 0 \\
(\phi^T N_9) - P_0 &= 0 \\
N_{8,9,0}(t_0) &= N_{t_0}(p).
\end{aligned} \tag{21}$$

In order to simulate the dynamics of a traveling wave reactor, the initial concentration of the material is such that the power density is concentrated on one axial end of the reactor (the “starter” end). The region immediately next to the starter end has a higher concentration of fertile material, and the idea is that leaking neutrons from the starter end will be absorbed and transmuted to fissile nuclides. Over time, a successful rate of continuous fuel breeding will cause the reaction to travel axially to the other end of the reactor.

#### 4.2. Solving the forward problem in MATLAB

To solve system (21), we leverage the sophistication and convenience of the time integrators built into MATLAB, two of which can solve DAEs [20]. The solvers employ dynamic error control to a user-specified tolerance and output a data structure that can be used to evaluate the solution at any  $t \in [t_0, t_f]$ , which becomes necessary during the subsequent adjoint solve.

Table 1 specifies the geometry and parameters for the forward problem. We choose cross-sections representative of the nuclide’s resonance integrals and amplify the diffusion coefficient to simulate the presence of a moderator and facilitate the breeding process.

Table 1: Geometry and parameters for reactor problem

Parameter(s)	Value(s)
$L, R$	400cm, 150cm
$\sigma_{f,9}, \sigma_{a,9}, \sigma_{t,9}$	1000b, 1018b, 1026b
$\sigma_{a,8}, \sigma_{t,8}$	500b, 600b
$\sigma_{a,0}, \sigma_{t,0}$	20b, 50b
$\Gamma$	$0.1 \left(1 - \frac{\sigma_{f,9}}{\sigma_{a,9}}\right)$
$\nu$	2.2
$\alpha_D$	500
$P_0/V_{reactor}$	100W

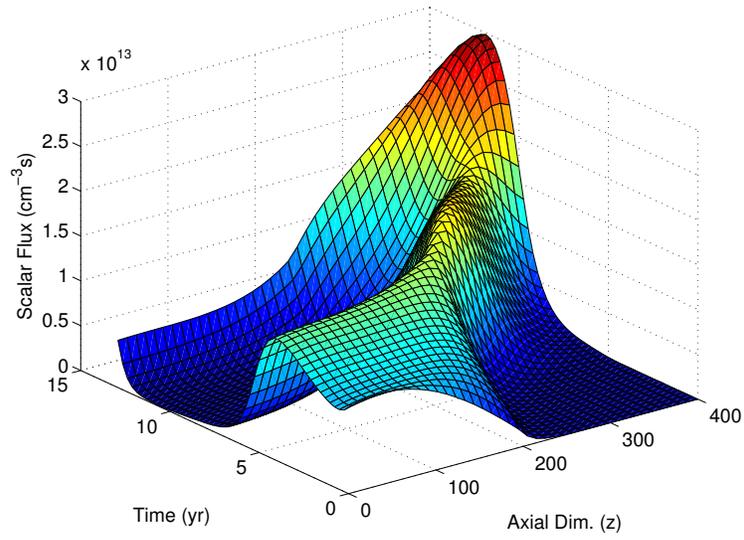
Table 2 specifies the initial conditions for the forward problem. As previously mentioned, the initial fuel loading of the reactor is such that the power density will initially be concentrated at one axial end of the reactor. We do not specify the initial flux distribution explicitly, since it follows from the algebraic constraint in system (21).

Table 2: Initial conditions for reactor problem

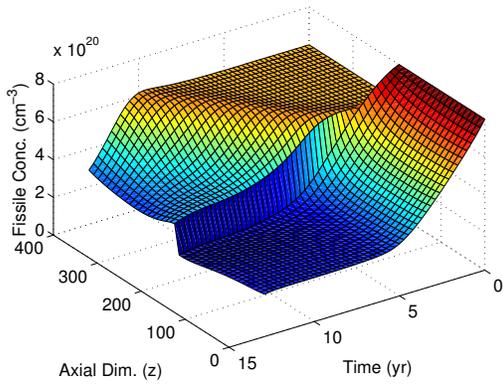
Unknown	Initial Value
Fissile concentration $N_9(t_0)$	$8.0 \times 10^{20} \text{ cm}^{-3}$ $0 \leq z \leq \frac{L}{2}$ $6.0 \times 10^{20} \text{ cm}^{-3}$ $\frac{L}{2} \leq z \leq L$
Fertile concentration $N_8(t_0)$	$6.0 \times 10^{20} \text{ cm}^{-3}$ $0 \leq z \leq \frac{L}{2}$ $1.5 \times 10^{21} \text{ cm}^{-3}$ $\frac{L}{2} \leq z \leq L$
Fission product concentration $N_0(t_0)$	$0.0 \text{ cm}^{-3}$ $0 \leq z \leq L$

We run an example problem with 40 spatial cells using the *ode23t* (a Runge-Kutta 2/3-like scheme) solver and a relative time-integration tolerance of  $10^{-7}$ . The problem runs until the external absorber concentration approaches zero, indicating the system can no longer sustain the reaction. In this case, the reactor lifetime is just under 13 years. Figure 2 shows the trajectory of the unknowns.

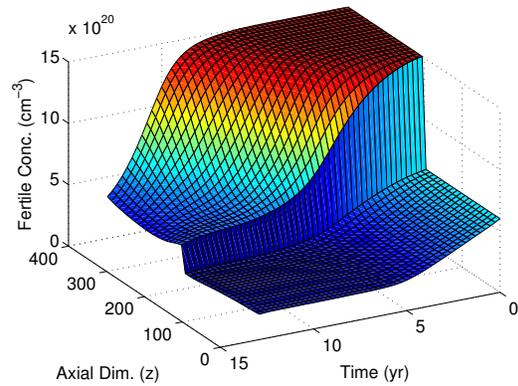
For the first five years, the reaction stays relatively stationary, and the system is dominated by leakage both out of the problem and into the breeding region. The result is a relatively sharp decline in the external absorber concentration as reactivity is lost due to fuel depletion. After some time, sufficient fissile material is generated in the cells next to the starter region, and the reaction begins to move axially through the reactor. Once away from the boundary, neutron loss due to leakage abates, requiring an increase in the external absorber concentration to maintain criticality. Near the end of the reactor lifetime, the fuel has depleted substantially, driving the flux magnitude higher (to satisfy the power constraint), and the system eventually is no longer able to sustain criticality.



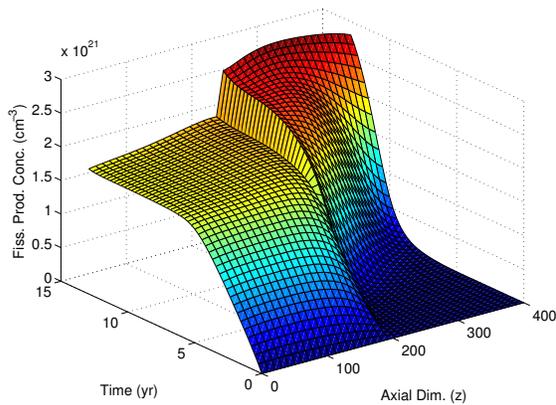
(a) Neutron flux



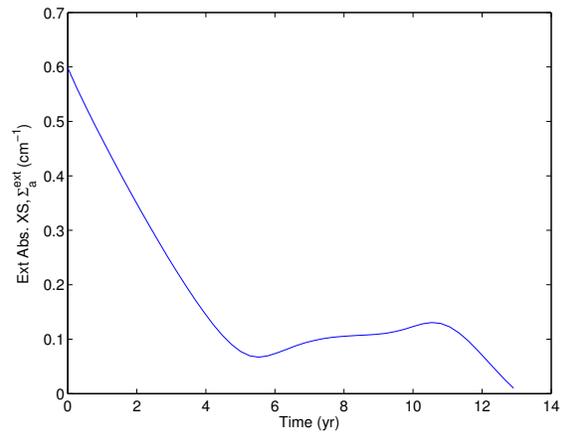
(b) Fissile ( $^{239}\text{Pu}$ ) inventory



(c) Fertile ( $^{238}\text{U}$ ) inventory



(d) Fission product inventory



(e) External absorber macroscopic cross-section required for criticality

Figure 2: Representative solution for the breed/burn problem (note the differing viewing angle for each surface plot)

Reactor analysts are typically interested in a set of figures-of-merit (FOM) and the sensitivity of the FOMs to the design inputs or physical parameters. We define two such metrics for this reactor: (1) the inventory of fertile material at  $t_f$  and (2) the ratio of the neutrons leaked from the system to neutrons used for the production of fissile material. The equations for these quantities in discrete form are

$$I_1 = \sum_{i=1}^n \pi R^2 \Delta z N_{8,i}(t_f) \quad (22)$$

$$I_2 = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} \frac{\pi R^2 (-J_{1/2}(t) + J_{n+1/2}(t)) + \Delta z \sum_{i=1}^n D_i(t) B_g^2 \phi_i(t)}{\pi R^2 \Delta z \sigma_{a,8} \sum_{i=1}^n \phi_i(t) N_8(t)} dt. \quad (23)$$

We will compute the sensitivity of these metrics with respect to the following vector of parameters:

$$p \equiv \langle \sigma_{f,9}, \sigma_{a,9}, \sigma_{t,9}, \sigma_{a,8}, \sigma_{t,8}, \sigma_{a,0}, \sigma_{t,0}, \Gamma, \nu, \alpha_D \rangle^T.$$

#### 4.3. Solving the adjoint problem in MATLAB

The adjoint satisfies a DAE system with a terminal condition, which we rewrite here along with the resulting sensitivity equation:

$$\begin{aligned} (\lambda^T F_x)' &= \lambda^T F_x - \mathcal{L}_x \\ \lambda^d(t_f) &= [(g_{x^d})^T - (F_{x^d}^a)^T [(F_{x^a}^a)^T]^{-1} (g_{x^a})^T]_{t=t_f} \\ \frac{dG}{dp} &= \left[ (\lambda^d)^T x_p^d \right]_{t=t_0} + \left[ g_p - g_{x^a} [F_{x^a}^a]^{-1} F_p^a \right]_{t=t_f} + \int_{t_0}^{t_f} \left\{ \mathcal{L}_p - \lambda^T F_p \right\} dt. \end{aligned}$$

We again allow the built-in solvers in MATLAB to handle the time integration, this time backwards using the *ode23t* solver with relative integration tolerance of  $10^{-8}$ . The terminal condition, dynamical equation, and sensitivity equation each require differentiation of the system  $F(\cdot)$  and/or FOM  $I$  with respect to the unknowns  $x$  or parameters  $p$ . For efficiency, we use hand-coded Jacobians that have been verified against automatic differentiation results from the INTLAB package [21]. To compute these derivatives, we need the solution to the forward problem at any  $t \in [t_0, t_f]$ ; the data structure output from the forward model allows for interpolation in  $t$  at the same order of accuracy as the initial solve.

We integrate the sensitivity equation simultaneously with the dynamical equation by appending a decoupled unknown vector to the  $\lambda$  vector. These unknowns satisfy

$$\begin{aligned} \dot{g}_p &= -\left\{ \mathcal{L}_p - \lambda^T F_p \right\} \\ g_p(t_f) &= \left[ g_p - g_{x^a} [F_{x^a}^a]^{-1} F_p^a \right]_{t=t_f}. \end{aligned}$$

This technique gives an integration tolerance consistent with the solve for  $\lambda$  and avoids the need to store and compute a quadrature rule. We perform an analogous integration during the forward solve for the integral term in  $I_2$ .

#### 4.4. First-order sensitivity results

We compute first-order sensitivities of FOM  $I_1$  and  $I_2$  to the defined parameter vector. We then compare these results with estimates obtained from divided differences, whereby we perturb one entry in the parameter vector, recompute the forward problem, and make a linear estimate for the gradient in that direction. The problem has 100 spatial cells and is run to seven years of reactor lifetime. Tables 3 and 4 summarize the results for  $I_1$  and  $I_2$ , respectively.

Table 3: Estimate for  $\frac{dI_1}{dp}$  computed via adjoints, showing close agreement with the result from divided differences

Parameter	Divided Diff.	Adjoint	Rel. Error
$\sigma_{f,9}$	1.019e+49	1.020e+49	3.042e-04
$\sigma_{a,9}$	-9.057e+48	-9.058e+48	5.409e-05
$\sigma_{t,9}$	3.273e+47	3.271e+47	7.877e-04
$\sigma_{a,8}$	-2.158e+47	-2.161e+47	1.349e-03
$\sigma_{t,8}$	3.497e+47	3.493e+47	1.189e-03
$\sigma_{a,0}$	-1.046e+49	-1.046e+49	1.487e-04
$\sigma_{t,0}$	2.031e+48	2.032e+48	8.914e-04
$\Gamma$	1.576e+28	1.576e+28	1.889e-05
$\nu$	-2.534e+27	-2.534e+27	2.071e-04
$\alpha_D$	-1.294e+24	-1.294e+24	5.691e-05

Table 4: Estimate for  $\frac{dI_2}{dp}$  computed via adjoints, showing close agreement with the result from divided differences

Parameter	Divided Diff.	Adjoint	Rel. Error
$\sigma_{f,9}$	-3.455e+18	-3.458e+18	8.901e-04
$\sigma_{a,9}$	-1.899e+17	-1.916e+17	9.043e-03
$\sigma_{t,9}$	1.720e+17	1.720e+17	3.520e-04
$\sigma_{a,8}$	1.439e+18	1.437e+18	1.138e-03
$\sigma_{t,8}$	6.209e+16	6.207e+16	2.977e-04
$\sigma_{a,0}$	-9.391e+18	-9.391e+18	1.144e-06
$\sigma_{t,0}$	1.472e+18	1.472e+18	1.123e-04
$\Gamma$	-1.324e-03	-1.322e-03	1.225e-03
$\nu$	-2.965e-03	-2.967e-03	7.239e-04
$\alpha_D$	-5.744e-07	-5.747e-07	5.062e-04

For both metrics and each of the 10 parameters, the divided difference and adjoint estimate agree to within at least 1%. The advantage of the adjoint approach is that it takes only two system solves (one forward and one backward) to compute all dimensions of the gradient, whereas the divided difference approach takes one forward solve *per* dimension (10 in this case). Recent work [19] related to gradient-enhanced response surface modeling shows that the modest cost of computing adjoint-based sensitivities results in increasingly accurate uncertainty estimates, especially in the case(s) of codes with sparsely sampled and high-dimensional input spaces.

#### 4.5. Global time-discretization error estimates for the reactor problem

We now give predictions of the global time discretization error in our solution vector. We return to our Runge-Kutta 2-3 scheme described in the previous section and verify that this solver is producing a solution that is within reasonable agreement with the MATLAB time-integrator. We also verify that the adjoint-based sensitivity estimates are in agreement with divided difference estimates and with the MATLAB sensitivity estimates.

We redefine our quantity of interest as the fissile nuclide concentration in the first spatial cell after five years of reactor life. Our goal is to estimate the error in this quantity due to time discretization. Figure 3 gives our estimate of this error as a function of the time-step for a 40-cell problem. We note that our error prediction decreases as the square of the time step, which we expect of the true error. We also note that the predicted error relative to the unknown is less than 1% for all time steps considered.

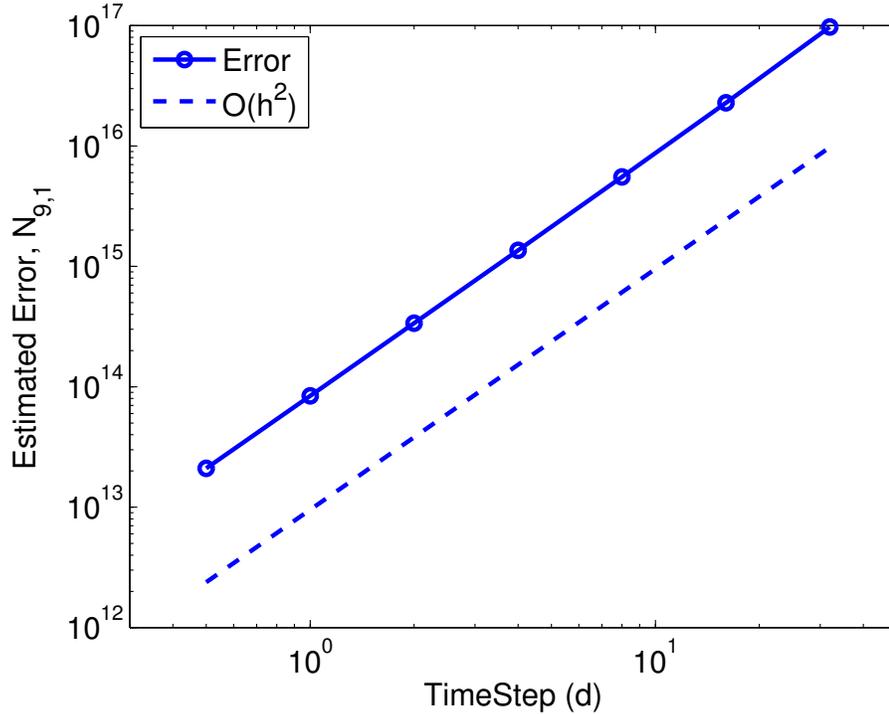


Figure 3: Predicted time-discretization error in the predicted fissile concentration in cell 1 as a function of the time step

## 5. Addition of Heat Transfer Physics

We emphasize that the power of this abstraction is its flexibility. To illustrate this, we add a heat conduction equation to the system to model the temperature distribution in the reactor. This in effect modifies our  $F(\cdot)$  function but does not change the algebra leading to the equations governing the adjoint variable or sensitivity estimate. We first describe our heat transfer model and then show the result of its implementation in our reactor problem.

### 5.1. Heat transfer model

The general 1D heat conduction equation is

$$c_p(z, t)\rho(z, t)\frac{\partial T(z, t)}{\partial t} = -\frac{\partial}{\partial z}\{\Psi(z, t)\} + Q(z, t) - L_r(z, t),$$

where the properties density ( $\rho$ ) and specific heat ( $c_p$ ) are defined functions of the nuclide densities,  $\Psi(\cdot)$  is the heat flux,  $Q(z, t)$  is a heating source term, in this case due to fission, and  $L_r(z, t)$  is a term to account

for heat loss due to radial convection. We again integrate over each cell  $i$  (all cells have width  $\Delta z$ ) to obtain an exact cell balance equation:

$$\Delta z c_{p,i}(t) \rho_i(t) \frac{\partial T_i(t)}{\partial t} = -[\Psi_{i+1/2}(t) - \Psi_{i-1/2}(t)] + \Delta z Q_i(t) - \Delta z L_{r,i}(t).$$

Now we invoke Fourier's law to write  $\Psi_i(t) = -k_i(t) \frac{\partial T_i(t)}{\partial z}$ , where  $k$ , the thermal conductivity, is also a function of the local nuclide densities. To enforce continuity of heat flux at interior cell-edges, we temporarily add an unknown temperature at the cell boundary, write a balance equation for  $\Psi_{i+1/2}$  (suppressing time arguments)

$$\Psi_{i+1/2} = -k_i \frac{T_{i+1/2} - T_i}{\Delta z/2} = -k_{i+1} \frac{T_{i+1} - T_{i+1/2}}{\Delta z/2},$$

solve for  $T_{i+1/2}$

$$T_{i+1/2} = \frac{k_i T_i + k_{i+1} T_{i+1}}{k_i + k_{i+1}},$$

substitute back into the equation for  $\Psi_{i+1/2}$ , and manipulate to obtain

$$\Psi_{i+1/2} = -\frac{2}{\Delta z} \frac{k_i k_{i+1}}{k_i + k_{i+1}} [T_{i+1} - T_i].$$

The fission heating term couples the neutronic and heat transfer equations and is written as

$$Q_i(t) = \phi_i N_{9,i} \sigma_{f,9} E_f$$

where  $E_f$  is energy release per fission. We next make a crude approximation for the radial convection term

$$\Delta z L_{r,i}(t) = h_r [T_i(t) - T_{\infty,r}],$$

where  $h_r$  and  $T_{\infty,r}$  are the radial heat transfer coefficient and radial ambient temperature, respectively. The resulting difference equation for an interior cell is

$$\begin{aligned} \frac{\partial T_i(t)}{\partial t} = \frac{1}{c_{p,i} \rho_i} \left\{ -\frac{2}{\Delta z^2} \frac{k_i k_{i+1}}{k_i + k_{i+1}} [T_i - T_{i+1}] - \frac{2}{\Delta z^2} \frac{k_i k_{i-1}}{k_i + k_{i-1}} [T_i - T_{i-1}] \right. \\ \left. + Q_i - \frac{h_r}{\Delta z} [T_i - T_{\infty,r}] \right\}. \end{aligned} \quad (24)$$

We model convective heat transfer at the problem boundaries. In cell 1, the boundary flux is

$$\Psi_{1/2} = -k_1 \frac{T_1 - T_{1/2}}{\Delta z/2} = -h_1 [T_{1/2} - T_{\infty,1}].$$

Solving for and eliminating  $T_{1/2}$ , we write the boundary flux as

$$\Psi_{1/2} = -\frac{2h_1 k_1}{h\Delta z + 2k_1} [T_1 - T_{\infty,1}].$$

An analogous result is found for  $\Psi_{N+1/2}$ , and the left and right boundary cell difference equations are

$$\begin{aligned} \frac{\partial T_1(t)}{\partial t} = \frac{1}{c_{p,1} \rho_1} \left\{ -\frac{2}{\Delta z^2} \frac{k_1 k_2}{k_1 + k_2} [T_1 - T_2] - \frac{2}{\Delta z} \frac{h_1 k_1}{h_1 \Delta z + 2k_1} [T_1 - T_{\infty,1}] \right. \\ \left. + Q_1 - \frac{h_r}{\Delta z} [T_1 - T_{\infty,r}] \right\}, \end{aligned} \quad (25)$$

$$\frac{\partial T_N(t)}{\partial t} = \frac{1}{c_{p,N}\rho_N} \left\{ -\frac{2}{\Delta z} \frac{h_N k_N}{h_N \Delta z + 2k_N} [T_N - T_{\infty,N}] - \frac{2}{\Delta z^2} \frac{k_N k_{N-1}}{k_N + k_{N-1}} [T_N - T_{N-1}] + Q_N - \frac{h_r}{\Delta z} [T_N - T_{\infty,r}] \right\}. \quad (26)$$

Finally we will assume a perfect mixture model for the material properties within a cell:

$$c_{p,i} = \frac{\sum_{j=8,9,0} c_{p,j} N_{j,i}}{\sum_{j=8,9,0} N_{j,i}}$$

$$\rho_{p,i} = \frac{\sum_{j=8,9,0} \rho_j N_{j,i}}{\sum_{j=8,9,0} N_{j,i}}$$

$$k_{p,i} = \frac{\sum_{j=8,9,0} k_j N_{j,i}}{\sum_{j=8,9,0} N_{j,i}}.$$

### 5.2. Forward problem and sensitivity solutions with the addition of heat transfer physics

As previously mentioned, the addition of new physics redefines our  $F(\cdot)$  function. In this case, we add a differential unknown (temperature) to each cell; these unknowns satisfy Eqs. (24)–(26) developed in the previous section. They are coupled to the neutronics equations via the fission heating term, and their initial conditions are solved for using a steady-state heat conduction equation consistent with the neutronics initial conditions.

Table 5 gives numerical values for the material properties and heat transfer parameters used to define the problem. Figure 4 shows the temperature distribution for the same 13 year simulation using 40 cells given in Fig. 2. As expected, the result of the forward problem shows that the temperature distribution moves with the neutron flux distribution through the reactor.

Table 5: Material properties and heat transfer parameters for the conduction model

Parameter(s)	Value(s)
$\rho_9, c_{p,9}, k_9$	19.82 g/cm <sup>3</sup> , 0.1485 J/(g-K), 0.0674 W/(cm-K)
$\rho_8, c_{p,8}, k_8$	19.10 g/cm <sup>3</sup> , 0.1162 J/(g-K), 0.2750 W/(cm-K)
$\rho_0, c_{p,0}, k_0$	1.930 g/cm <sup>3</sup> , 0.2352 J/(g-K), 0.3590 W/(cm-K)
$h_1, T_{\infty,1}$	0.40 W/(cm <sup>2</sup> -K), 295.0K
$h_N, T_{\infty,N}$	0.40 W/(cm <sup>2</sup> -K), 295.0K
$h_R, T_{\infty,R}$	3.00 W/(cm <sup>2</sup> -K), 295.0K

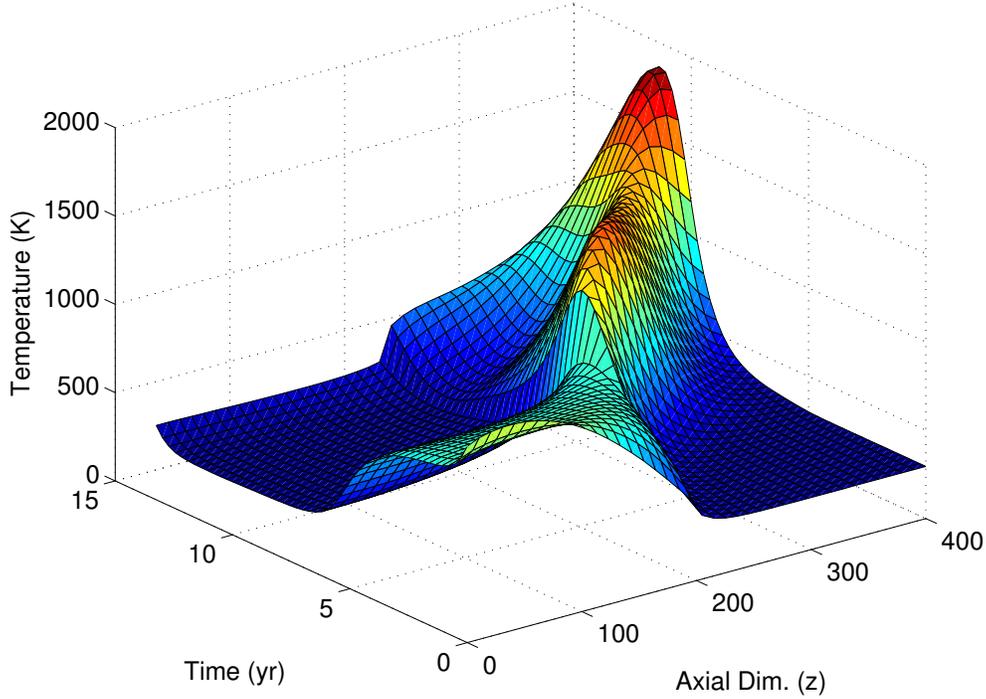


Figure 4: Temperature distribution in simplified traveling wave reactor model

The procedure for solving for the adjoint variables does not change with the addition of the heat transfer physics. In practice, the automatic differentiation (AD) packages would handle the modified  $F(\cdot)$  and  $I(t_f)$  functions (in our case, we hand-code derivatives for efficiency but verify against AD packages). As an example, we append the heat transfer parameters to the  $p$  vector,

$$p \equiv \langle \sigma_{f,9}, \sigma_{a,9}, \sigma_{t,9}, \sigma_{a,8}, \sigma_{t,8}, \sigma_{a,0}, \sigma_{t,0}, \Gamma, \nu, \alpha_D, h_1, h_N, h_R, T_{\infty,1}, T_{\infty,N}, T_{\infty,R} \rangle^T,$$

and define a heat transfer QOI, the average temperature in the reactor at  $t = t_f$ :

$$I_3 = \frac{1}{n} \sum_{i=1}^n T_i(t_f).$$

We again solve the system using 100 spatial cells and seven years of simulation time. The sensitivity of  $I_3$  with respect to our new set of parameters is computed via the adjoint method and the divided differences method. A comparison of the results is given in Table 6. Again, we see strong agreement between the adjoint and divided differences results, providing verification evidence for our adjoint solution framework. More important, however, we emphasize that the work required to add new physics was related only to model modification, *not* a reformulation of the adjoint equations or the solution technique for either the forward or reverse integration.

Table 6: Adjoint-based and divided-difference estimates of  $\frac{dI_3}{dp}$ , showing close agreement

Parameter	Divided Diff.	Adjoint	Rel. Error
$\sigma_{f,9}$	9.784e+18	9.776e+18	8.567e-04
$\sigma_{a,9}$	6.373e+19	6.377e+19	5.585e-04
$\sigma_{t,9}$	-9.109e+18	-9.100e+18	9.806e-04
$\sigma_{a,8}$	-1.796e+20	-1.797e+20	3.624e-04
$\sigma_{t,8}$	-6.505e+18	-6.494e+18	1.728e-03
$\sigma_{a,0}$	2.658e+20	2.671e+20	4.745e-03
$\sigma_{t,0}$	-7.388e+19	-7.356e+19	4.388e-03
$\Gamma$	-7.751e-02	-7.387e-02	4.695e-02
$\nu$	8.528e-02	8.523e-02	5.835e-04
$\alpha_D$	2.793e-05	2.794e-05	1.400e-04
$h_1$	-3.135e-02	-3.133e-02	3.878e-04
$h_N$	-2.338e-04	-2.135e-04	8.686e-02
$h_R$	-4.440e+01	-4.441e+01	3.161e-04
$T_{\infty,1}$	3.623e-04	3.624e-04	5.695e-05
$T_{\infty,N}$	2.755e-04	2.755e-04	7.318e-05
$T_{\infty,R}$	9.994e-01	9.994e-01	1.100e-08

## 6. Discussion and Conclusions

We have presented a flexible adjoint framework for DAEs in nuclear engineering applications. We are concerned primarily with index-1 systems, such as those in burnup equations where the neutron flux and thermal behavior evolve on much faster time scales and can thus be considered as algebraic constraints on the time scale of interest.

The adjoint calculations can be used for uncertainty propagation in the context of gradient-enhanced universal kriging or regression, as recently introduced [19, 22]. Their use appears in the context of computing gradients of metrics of interest with respect to uncertain parameters that have a probability structure. As a result, statistics of interest can be computed with far fewer samples [19, 22], facilitating efficient design optimization. Moreover, we derive theoretically a method that uses the same adjoint variable as the gradient to produce estimates of the time discretization error. This approach needs an estimate of the residual error, which we provide for the case of embedded Runge-Kutta integrators. As a result, we are now in a position to present estimates of both uncertainty propagation and approximation errors.

We demonstrate our findings on a simple pendulum example, a DAE for which we know the solution, and we show that the discretization error estimate is within 1% of the actual error for the whole tested range of time steps, which we find encouraging. In addition, we verify the adjoint use for derivative calculations on multiple instances of a traveling-wave reactor. Moreover, the high level of abstraction in our framework allows for increased flexibility in modeling, a fact that we demonstrate by adding physics to the traveling-wave reactor and the relatively immediate extension of our calculations to that case.

This work would benefit from extension in several directions to accommodate other challenges in nuclear engineering applications. A topic of interest for us in the near future is the extension of this framework for DAEs of higher index and for schemes that do not use a full nonlinear iteration on the constraint, as the semi-explicit schemes we have investigated here do. Moreover, a topic of practical interest is the derivation of inexpensive schemes for estimation of the residual error for other DAE indices. In particular, our framework allows for residual error in the algebraic constraints that may result in more efficient algorithms whose numerical error can now be estimated by our framework.

## Acknowledgements

H. F. Stripling is supported by the Department of Energy Computational Science Graduate Fellowship program under grant number DE-FD02-97ER25308. M. Anitescu is supported by the U.S. Department of Energy under contract no. DE-AC02-06CH11357.

- [1] M. D. DeHart, TRITON: A two-dimensional transport and depletion module for characterization of spent nuclear fuel, Oak Ridge National Laboratory, ORNL/TM-2005/39, Revision 5.1 Vol. I, Book 3, Sect. T1.
- [2] J. S. Hendricks, G. W. McKinney, M. L. Fensin, MCNPX, version 26e, Los Alamos Report, LA-UR-07-6632, November 2007.
- [3] D. I. Poston, H. R. Trellue, Development of a Fully Automated Monte Carlo Burnup Code MONTEBURNS, Los Alamos Report, LA-UR-99-42, January 1999.
- [4] G. Bell, S. Glasstone, Nuclear Reactor Theory, Van Nostrand Rienhold, New York, 1970.
- [5] A. Henry, Nuclear Reactor Analysis, MIT Press, Cambridge, 1975.
- [6] S. Dulla, E. H. Mund, P. Ravetto, The quasi-static method revisited, Progress in Nuclear Energy 50 (8) (2008) 908–920, in Honour of Prof. Bruno Montagnini. doi:10.1016/j.pnucene.2008.04.009.  
URL <http://www.sciencedirect.com/science/article/pii/S014919700800084X>
- [7] W. S. Yang, T. J. Downar, A linear time-dependent flux approximation for nuclide depletion based on a perturbation method, Annals of Nuclear Energy 17 (1) (1990) 37–42. doi:DOI: 10.1016/0306-4549(90)90063-J.  
URL <http://www.sciencedirect.com/science/article/pii/030645499090063J>
- [8] M. L. Williams, Development of depletion perturbation-theory for coupled neutron-nuclide fields, Nuclear Science and Engineering 70 (1) (1979) 20–36.
- [9] W. S. Yang, T. J. Downar, Generalized perturbation-theory for constant power core depletion, Nuclear Science and Engineering 99 (4) (1988) 353–366.
- [10] F. C. da Silva, Z. D. Thom, Depletion calculations with static generalized perturbation theory, Annals of Nuclear Energy 15 (8) (1988) 431–434. doi:DOI: 10.1016/0306-4549(88)90039-4.  
URL <http://www.sciencedirect.com/science/article/pii/0306454988900394>
- [11] T. J. Downar, Depletion perturbation theory for burnup dependent microscopic cross sections, Annals of Nuclear Energy 19 (1) (1992) 27–37. doi:DOI: 10.1016/0306-4549(92)90051-C.  
URL <http://www.sciencedirect.com/science/article/pii/030645499290051C>
- [12] R. van Geemert, J. E. Hoogenboom, Development of parallelized higher-order generalized depletion perturbation theory for application in equilibrium cycle optimization, Annals of Nuclear Energy 28 (14) (2001) 1377–1411. doi:DOI: 10.1016/S0306-4549(00)00135-3.  
URL <http://www.sciencedirect.com/science/article/pii/S0306454900001353>
- [13] A. Griewank, A. Walther, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2008.
- [14] K. Brenan, S. Campbell, S. Campbell, L. Petzold, Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, Vol. 14, Society for Industrial Mathematics (SIAM), Philadelphia, 1996.
- [15] Y. Cao, S. Li, L. Petzold, R. Serban, Adjoint sensitivity analysis for differential-algebraic equations: algorithms and software, Journal of Computational and Applied Mathematics 149 (1) (2002) 171–191. doi:DOI: 10.1016/S0377-0427(02)00528-9.  
URL <http://www.sciencedirect.com/science/article/pii/S0377042702005289>
- [16] Y. Cao, L. Petzold, A posteriori error estimation and global error control for ordinary differential equations by the adjoint method, SIAM Journal on Scientific Computing 26 (2) (2004) 359–374.
- [17] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, Springer-Verlag, Berlin, 1996.
- [18] U. Ascher, L. Petzold, Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, Society for Industrial and Applied Mathematics, Philadelphia, 1998, Ch. 10.
- [19] B. Lockwood, M. Anitescu, Gradient-enhanced universal kriging for uncertainty propagation, Nuclear Science and Engineering To appear. Also, Preprint P1808-1110, Mathematics and Computer Science Division, Argonne National Laboratory.
- [20] L. F. Shampine, M. W. Reichelt, J. A. Kierzenka, Solving index-1 daes in MATLAB and SIMULINK, SIAM Review 41 (3) (1999) 538–552.  
URL <http://www.jstor.org.lib-ezproxy.tamu.edu:2048/stable/2653267>
- [21] S. Rump, INTLAB - INTerval LABoratory, in: T. Csendes (Ed.), Developments in Reliable Computing, Kluwer Academic Publishers, Dordrecht, 1999, pp. 77–104, <http://www.ti3.tu-harburg.de/rump/>.
- [22] O. Roderick, M. Anitescu, P. Fischer, Stochastic finite element approaches using derivative information for uncertainty quantification, Nuclear Science and Engineering 164 (2) (2010) 122–139.

The submitted manuscript has been created by the University of Chicago as Operator of Argonne National Laboratory ("Argonne") under Contract No. DE-AC02-06CH11357 with the U.S. Department of Energy. The U.S. Government retains for itself, and others acting on its behalf, a paid-up, nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.