



ELSEVIER

Available online at [www.sciencedirect.com](http://www.sciencedirect.com)

SCIENCE @ DIRECT®

---

annals of  
NUCLEAR ENERGY

---

Annals of Nuclear Energy 31 (2004) 1039–1051

[www.elsevier.com/locate/anucene](http://www.elsevier.com/locate/anucene)

Technical note

# Efficient numerical solution of the point kinetics equations in nuclear reactor dynamics

Matthew Kinard, E.J. Allen\*

*Department of Mathematics and Statistics, Texas Tech University, Lubbock, TX 79409-1042, USA*

Received 9 December 2003; accepted 23 December 2003

---

## Abstract

A numerical procedure to efficiently calculate the solution to the point kinetics equation in nuclear reactor dynamics is described and investigated. Piecewise constant approximations of the reactivity and source functions are made. The resulting system of linear differential equations is solved exactly over each time step. The method is proved to converge with order  $h^2$  where  $h$  is the time step. The procedure is tested using a variety of initial conditions, data, and reactivity functions. The computational results indicate that the method is efficient and accurate.

© 2004 Elsevier Ltd. All rights reserved.

---

## 1. Introduction

The point kinetics equations in nuclear reactor dynamics are a system of coupled linear ordinary differential equations. The time-dependent parameters in this system are the reactivity and the neutron source terms. In the proposed numerical method, these time-dependent functions are approximated by piecewise constant functions over a partition of the total time interval. The resulting system of differential equations is then solved exactly over each time step in the partition.

An important property of the point kinetics equations is the stiffness of the system. It is well known that stiffness is a severe problem in numerical solution of the point kinetics equations and results in the need for small time steps in a computational

---

\* Corresponding author. Fax: +1-806-742-1112.

E-mail address: [callen@math.ttu.edu](mailto:callen@math.ttu.edu) (E.J. Allen).

scheme. However, if the reactivity and neutron source functions are slowly varying with respect to time, then piecewise constant approximations to these functions are accurate. The piecewise constant assumption allows us to solve the point kinetics equation exactly, and the problem of stiffness disappears. One of the advantages of the numerical method explored here is its simplicity. Furthermore, the method is efficient and accurate.

There has been a great deal of research done in the field of reactor kinetics. Much of the work has focused on eliminating the aforementioned problem of stiffness. [Chao and Attard in 1985](#) investigated this problem and developed the “stiffness confinement method” that places emphasis on the physical analogies of the point kinetics equations in order to deal with the problem. Sánchez devised and implemented an A-Stable Runge-Kutta Method to avoid the problem of stiffness in 1989. Several other creative schemes have been implemented as well. [J.A.W. da Nóbrega uses a Padé approximation to the solution in 1971](#), and [Aboanber and Nahla in 2002](#) derived a technique based on the analytical inversion of polynomials to aid in the solution to these equations.

An attractive feature of the computational method proposed in the present investigation is its simplicity. This feature makes the method easy to program and understand. In addition, the computational method described here is accurate and efficient.

This paper is organized in the following manner. First, a basic formulation of the point kinetics equations is given along with a derivation of the solution. Then, the numerical scheme is described. [Section 3](#) deals with the computational results using the new numerical method. An error analysis of the method is presented [Section 4](#). [Section 5](#) summarizes the results of the present investigation and concluding remarks are made.

## 2. Numerical method

The point kinetics equations for  $m$  delayed groups are ([Hetrick, 1971](#)):

$$\frac{dn(t)}{dt} = \frac{\rho(t) - \beta}{\Lambda} n(t) + \sum_{i=1}^m \lambda_i C_i(t) + F(t). \quad (2.1)$$

$$\frac{dC_i(t)}{dt} = \frac{\beta_i}{\Lambda} n(t) - \lambda_i C_i(t) \quad i = 1, 2, \dots, m \quad (2.2)$$

where  $n(t)$  is the time-dependent neutron density,  $C_i(t)$  is the  $i$ th precursor density,  $\rho(t)$  is the time-dependent reactivity function,  $\beta_i$  is the  $i$ th delayed fraction, and  $\beta = \sum_{i=1}^m \beta_i$  is the total delayed fraction. In addition,  $\Lambda$  is the neutron generation time,  $\lambda_i$  is the  $i$ th group decay constant, and  $F(t)$  is the time-dependent neutron source function.

A useful way to consider this problem is by putting the problem in matrix form. In the present investigation, the point kinetics equations is considered in the form:

$$\begin{aligned}\frac{d\vec{x}}{dt} &= A\vec{x} + B(t)\vec{x} + \vec{F}(t) \\ \vec{x}(0) &= \vec{x}_0\end{aligned}\quad (2.3)$$

where we define  $A$  as the  $(m+1) \times (m+1)$  matrix:

$$A = \begin{pmatrix} \frac{-\beta}{\Lambda} & \lambda_1 & \lambda_2 & \cdots & \lambda_m \\ \frac{\beta_1}{\Lambda} & -\lambda_1 & 0 & \cdots & 0 \\ \frac{\beta_2}{\Lambda} & 0 & -\lambda_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\beta_m}{\Lambda} & 0 & 0 & \cdots & -\lambda_m \end{pmatrix},$$

$B(t)$  is the  $(m+1) \times (m+1)$  matrix:

$$B(t) = \begin{pmatrix} \frac{\rho(t)}{\Lambda} & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix},$$

and,  $\vec{F}(t)$  is defined as:

$$\vec{F}(t) = \begin{pmatrix} F(t) \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

It is often the case that the reactivity and source functions are slowly varying with respect to time, in comparison with the solution of Eq. (2.3). The numerical method

approximates these two functions by piecewise constant functions over a partition in time. Specifically, let

$$\rho(t) \approx \rho\left(\frac{t_i + t_{i+1}}{2}\right) = \rho_i \text{ for } t_i \leq t \leq t_{i+1}$$

and

$$\vec{F}(t) \approx \vec{F}\left(\frac{t_i + t_{i+1}}{2}\right) = \vec{F}_i \text{ for } t_i \leq t \leq t_{i+1}$$

yielding the following initial-value problem:

$$\frac{d\hat{\vec{x}}}{dt} = A\hat{\vec{x}} + B_i\hat{\vec{x}} + \vec{F}_i \quad (2.4)$$

$$\hat{\vec{x}}(t_i) = \vec{x}_i$$

where  $\hat{\vec{x}}(t) \approx \vec{x}(t)$

Notice that Eq. (2.4) can now be solved exactly. Multiplying both sides by the integrating factor  $e^{-(A+B_i)t}$  yields:

$$\frac{d}{dt} \left( e^{-(A+B_i)t} \hat{\vec{x}} \right) = e^{-(A+B_i)t} \vec{F}_i. \quad (2.5)$$

Integrating both sides of the equation over the  $i$ th time step from  $t_i$  to  $t_{i+1}$  gives

$$e^{-(A+B_i)t_{i+1}} \hat{\vec{x}}_{i+1} - e^{-(A+B_i)t_i} \hat{\vec{x}}_i = \int_{t_i}^{t_{i+1}} e^{-(A+B_i)u} \vec{F}_i du. \quad (2.6)$$

Subtracting and evaluating the integral yields an equation of the form:

$$\hat{\vec{x}}_{i+1} = e^{(A+B_i)h_i} \hat{\vec{x}}_i + (e^{(A+B_i)h_i} - I)(A + B_i)^{-1} \vec{F}_i \text{ where } h_i = t_{i+1} - t_i \quad (2.7)$$

The numerical scheme proposed in the present investigation is based on this equation. We now consider an efficient way to compute this equation.

Let  $A + B_i = X_i D_i X_i^{-1}$  where  $X_i$  are the eigenvectors of  $A + B_i$  and  $D_i = \text{diag}[\omega_1^{(i)}, \dots, \omega_{m+1}^{(i)}]$ . The values  $\omega_1^{(i)}, \omega_2^{(i)}, \dots, \omega_{m+1}^{(i)}$  are the associated eigenvalues of  $A + B_i$  (Note: a discussion of the efficient calculation of the eigenvalues and eigenvectors is given later). Replacing  $A + B_i$  with the associated decomposition, Eq. (2.7) becomes:

$$\hat{\vec{x}}_{i+1} = X_i e^{D_i h_i} X_i^{-1} \hat{\vec{x}}_i + (X_i e^{D_i h_i} X_i^{-1} - I) X_i D_i^{-1} X_i^{-1} \vec{F}_i \quad (2.8)$$

or equivalently,

$$\hat{\vec{x}}_{i+1} = X_i e^{D_i h_i} X_i^{-1} \left[ \hat{\vec{x}}_i + X_i D_i^{-1} X_i^{-1} \vec{F}_i \right] - X_i D_i^{-1} X_i^{-1} \vec{F}_i \quad (2.9)$$

Notice that after  $X_i$  and  $D_i$  are determined, (2.9) can be computed at each time step by using only a series of matrix-vector multiplications and vector additions. Hence (2.9) can be computed rapidly once  $X_i$  and  $D_i$  are calculated.

Thus, in order for the method to be efficient, the computation of their eigenvectors and their associated eigenvalues must be efficient as well. It is well known that the  $(m+1)$  eigenvalues of the point-kinetics matrix are the roots of the inhour equation (Hetrick, 1971):

$$\rho_i = \beta + \Lambda\omega - \sum_{j=1}^m \frac{\beta_j \lambda_j}{\omega + \lambda_j} \quad (2.10)$$

which can be equivalently expressed as a polynomial  $P_i(\omega)$  of degree  $m+1$ :

$$P_i(\omega) = (\rho_i - \omega\Lambda) \prod_{l=1}^m (\lambda_l + \omega) - \omega \sum_{k=1}^m \beta_k \prod_{\substack{l=1 \\ l \neq k}}^m (\lambda_l + \omega) = 0 \quad (2.11)$$

Note that it is well-known that the roots of  $P_i(\omega)$  are real (Hetrick, 1971). A modified Newton's method is used to calculate the roots of the polynomial. After a given root,  $\omega_i$ , is calculated using Newton's method, the polynomial is deflated using Horner's method. (All of the algorithms are provided as MATLAB codes in Kinard, 2003.) These calculations yield the set of eigenvalues  $\{\omega_1^{(i)}, \omega_2^{(i)}, \dots, \omega_{m+1}^{(i)}\}$ , each calculated for the  $i$ th time step. This procedure is very rapid and calculating each eigenvalue only requires three or four iterations of Newton's method to yield accurate results.

The task of calculating the eigenvectors is quite simple. Hetrick points out that the eigenvectors of the point-kinetics matrix have the form

$$X_i = U_i \text{ where } \bar{u}_k^{(i)} = \left[ 1, \frac{\mu_1}{\lambda_1 + \omega_k^{(i)}}, \frac{\mu_2}{\lambda_2 + \omega_k^{(i)}}, \dots, \frac{\mu_m}{\lambda_m + \omega_k^{(i)}} \right]^T$$

and where each  $\bar{u}_k^{(i)}$  is the  $k$ th column of  $U_i$  for  $k=1, 2, \dots, m+1$ . Note that the  $\omega_k^{(i)}$  are the appropriate eigenvalues, and  $\mu_l = \frac{\beta_l}{\Lambda}$  for  $l=1, 2, \dots, m$ . In addition, the inverse of the matrix of eigenvectors can be written as:

$$X_i^{-1} = V_i^T \text{ where } \bar{v}_k^{(i)} = v_k^{(i)} \left[ 1, \frac{\lambda_1}{\lambda_1 + \omega_k^{(i)}}, \frac{\lambda_2}{\lambda_2 + \omega_k^{(i)}}, \dots, \frac{\lambda_m}{\lambda_m + \omega_k^{(i)}} \right]^T$$

is the  $k$ th column of  $V_i$  and  $v_k^{(i)} = \left[ 1 + \sum_{j=1}^m \frac{\mu_j \lambda_j}{(\lambda_j + \omega_k^{(i)})^2} \right]^{-1}$ .

The eigenvalues and eigenvectors are computed at each time step  $i$  if the reactivity and source functions change with time. Thus, the matrices,  $X_i$  and  $D_i$  are efficiently calculated at each step  $i$  and Eq. (2.9) is used to calculate  $\tilde{x}_{i+1}$ . We now consider computational results obtained from this method.

### 3. Analysis and computational results

We consider a variety of computational examples. All of the examples were computed using the MATLAB code given in Kinard, (2003). Built-in MATLAB functions were not used so that other investigators can easily translate the MATLAB code into another programming language such as FORTRAN or C++ . Not using the built-in functions also gives us complete control over the method. For simplicity, throughout the remainder of the paper, the proposed method will be referred to as the piecewise constant approximation method, or the PCA method.

All of the computational examples have the same form for the initial condition. The initial condition assumes a source-free equilibrium where  $\tilde{x}(0)$  equals:

$$\begin{pmatrix} 1 \\ \frac{\beta_1}{\lambda_1 \Lambda} \\ \frac{\beta_2}{\lambda_2 \Lambda} \\ \vdots \\ \frac{\beta_m}{\lambda_m \Lambda} \end{pmatrix}$$

The first example models a step-reactivity insertion. In this case,  $\rho(t) = \rho_0$  for  $t \geq 0$ . As the reactivity and source functions are constant, the PCA method gives the exact solution as can be seen by referring to (2.4) and (2.3). For step-reactivity insertions, the following parameters were used:  $\Lambda = 0.00002$ ,  $\beta = 0.007$ ,  $\beta_i = (0.000266, 0.001491, 0.001316, 0.002849, 0.00896, 0.000182)$ , and  $\lambda_i = (0.0127, 0.0317, 0.155, 0.311, 1.4, 3.87)$  with  $m=6$  delayed groups. We consider three step insertions, one prompt subcritical  $\rho=0.003$ , one prompt critical  $\rho=0.007$ , and one prompt supercritical  $\rho=0.008$ . The results of the PCA method are presented in Tables 1–3 and are compared with those of the SCM method proposed by Chao and Attard in 1985, and with the exact values.

For step-reactivity insertions, notice that the roots of the inhour equations need only be calculated once, at time  $t=0$ . There is no need to re-calculate the roots at each time stop as  $X_i$  and  $D_i$  do not change. The roots of the inhour equation for the three step reactivity examples are given in Table 4. Also, while the SCM method yields an accurate solution, the PCA method yields exact results for any time interval in the case of a step-reactivity insertion.

Table 1

A comparison using a prompt subcritical step reactivity,  $\rho=0.003$ 

Time (s)	SCM	PCA	Exact
$t=1$	2.2254	2.2098	2.2098
$t=10$	8.0324	8.0192	8.0192
$t=20$	$2.8351 \times 10^1$	$2.8297 \times 10^1$	$2.8297 \times 10^1$

Table 2

A comparison using prompt critical step reactivity,  $\rho=0.007$ 

Time (s)	SCM	PCA	Exact
$t=0.01$	4.5001	4.5088	4.5088
$t=0.5$	$5.3530 \times 10^3$	$5.3459 \times 10^3$	$5.3459 \times 10^3$
$t=2$	$2.0627 \times 10^{11}$	$2.0591 \times 10^{11}$	$2.0591 \times 10^{11}$

Table 3

A comparison using a prompt supercritical step reactivity,  $\rho=0.008$ 

Time (s)	SCM	PCA	Exact
$t=0.01$	6.2046	6.0229	6.0229
$t=0.5$	$1.4089 \times 10^3$	$1.4104 \times 10^3$	$1.4104 \times 10^3$
$t=2$	$6.1574 \times 10^{23}$	$6.1634 \times 10^{23}$	$6.1634 \times 10^{23}$

Table 4

Solutions to the inhour equation for several step reactivities

$\rho=0.003$	$\rho=0.007$	$\rho=0.008$
-0.13513	-0.01312	-0.01307
-0.49175	-0.03926	-0.03827
0.12353	-0.14053	-0.13723
-0.16620	-0.76952	-0.65721
-1.14757	-3.17734	-2.54004
-3.72268	11.6442	-5.15004
-200.7647	-13.2449	52.80352

Consider now the case of a ramp reactivity. Ramp reactivities usually take the form  $\rho(t)=\rho_0t$  for some constant  $\rho_0=\frac{\rho}{\beta}$  which is a given reactivity expressed in dollars (Hetrick, 1971). We will use the same parameters that we used in the step-reactivity example, and compare our results to those of Chao and Attard. The ramp reactivity that we consider is a “moderately fast” ramp of \$.01/sec. The computational results for this case are presented in Table 5 along with the exact values obtained from Sanchéz in 1989. While the ramp reactivity results are not as accurate as those for the step reactivity example, it will be shown that the error in the PCA method is proportional to  $h^2$ .

Finally, to provide an interesting example of the accuracy and flexibility of the PCA method, we will examine the sinusoidal reactivity case considered in Hetrick. The reactivity function in this case has the form:

$$\rho(t)=\rho_0\sin\frac{\pi t}{T}$$

where  $T$  is a half-period. For this reactivity function, we consider only one delayed neutron group, instead of the six as in the previous examples. The parameters for this example are:  $\rho_0=0.005333$  (or equivalently, 0.68 dollars).  $\beta=0.0079$ ,  $\lambda=0.077$ ,  $\Lambda=10^{-3}$  and  $T=50$ . The initial conditions for the computation were selected to be at equilibrium as previously described. The computation was run for a period of 100 seconds, and Fig. 1 plots the neutron density with respect to time. Fig. 1 agrees with the figure provided in Hetrick. In Fig. 2, the precursor density for this problem is plotted along with the corresponding neutron density. These results were obtained for a time step  $h=1$  sec. No significant changes in the figures occur if a smaller time step is used.

4. Error analysis

Consider the matrix formulation of the point kinetics equations

$$\frac{d\vec{x}}{dt}=A\vec{x}+B(t)\vec{x}+\vec{F}(t)$$
$$\vec{x}(0)=\vec{x}_0.$$

(4.1)

Table 5  
A comparison using a moderately fast ramp reactivity

$t$ (s)	$\theta$ Weighting $h=0.01$	$\theta$ Weighting $h=0.1$	PCA $h=0.01$	PCA $h=0.1$	Exact
2	1.3382	1.3383	1.3382	1.3305	1.3379
4	2.2286	2.2290	2.2278	2.2117	2.2283
6	5.5830	5.5885	5.5802	5.5230	5.5815
8	$4.2811\times10^1$	$4.3215\times10^1$	$4.2772\times10^1$	$4.2049\times10^1$	$4.2781\times10^1$
9	$4.8816\times10^2$	$5.0636\times10^2$	$4.8735\times10^2$	$4.7639\times10^2$	$4.8745\times10^2$



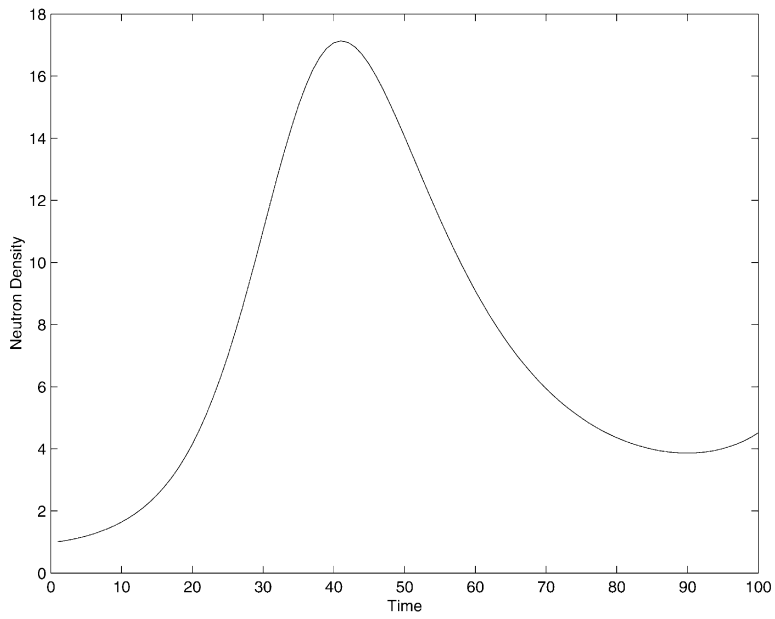


Fig. 1. Neutron densities based on a sinusoidal reactivity.

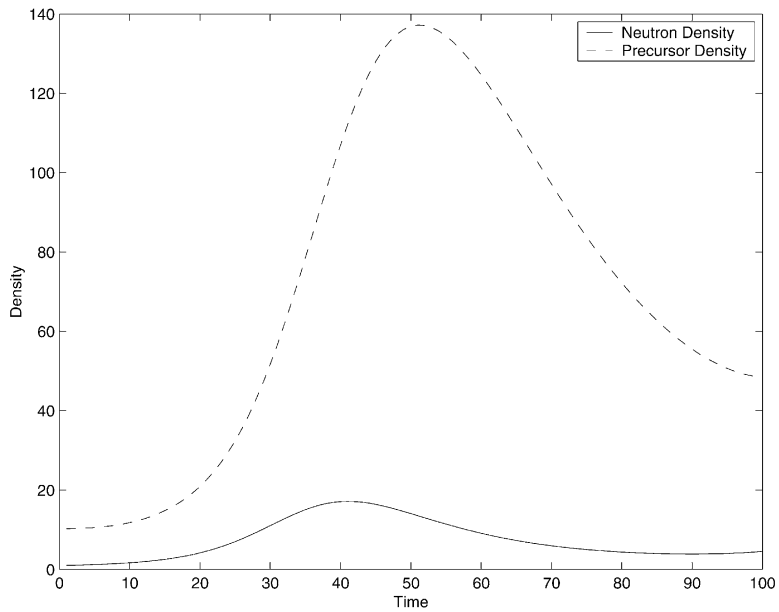


Fig. 2. Precursor density together with neutron density.

In addition, consider the matrix form of the PCA scheme:

$$\frac{d\hat{\vec{x}}}{dt} = A\hat{\vec{x}} + B_i\hat{\vec{x}} + \vec{F}_i \quad (4.2)$$

$$\hat{\vec{x}}(t_i) = \hat{\vec{x}}_i$$

for  $t_0=0$ ,  $t_1=h$ ,  $t_2=2h$ , ... and  $h=t_{i+1}-t_i$ . Subtracting Eqs. (4.1) and (4.2) yields:

$$\frac{d\vec{\varepsilon}}{dt} = A\vec{\varepsilon} + (B(t) - B_i)\vec{x} + B_i\vec{\varepsilon} + \vec{F}(t) - \vec{F}_i \quad (4.3)$$

$$\vec{\varepsilon}(t_i) = \vec{\varepsilon}_i$$

where  $\vec{\varepsilon}(t) = \vec{x}(t) - \hat{\vec{x}}(t)$ ,  $\vec{\varepsilon}(0) = \vec{\varepsilon}(0) = \vec{0}$  and  $\vec{x}(t)$  is the exact solution.

To simplify the notation and calculations, let  $\vec{g}_i(t) = (B(t) - B_i)\vec{x}(t) + \vec{F}(t) - \vec{F}_i$ . This change yields the system:

$$\frac{d\vec{\varepsilon}}{dt} = (A + B_i)\vec{\varepsilon} + \vec{g}_i(t) \quad (4.4)$$

$$\vec{\varepsilon}(t_i) = \vec{\varepsilon}_i$$

Suppose that  $B_i = B(t_{i+\frac{1}{2}})$ ,  $F_i = \vec{F}(t_{i+\frac{1}{2}})$  where  $t_{i+\frac{1}{2}} = \frac{t_i + t_{i+1}}{2}$ . Then, by our piecewise constant approximation, we see that  $\vec{g}_i(t_{i+\frac{1}{2}}) = \vec{0}$ .

Now consider Eq. (4.4). Multiplying by the appropriate integrating factor yields:

$$\frac{d}{dt} [e^{-(A+B_i)t} \vec{\varepsilon}] = e^{-(A+B_i)t} \vec{g}_i(t).$$

Integrating from  $t_i$  to  $t_{i+1}$  gives us:

$$e^{-(A+B_i)t_{i+1}} \vec{\varepsilon}_{i+1} - e^{-(A+B_i)t_i} \vec{\varepsilon}_i = \int_{t_i}^{t_{i+1}} e^{-(A+B_i)t} \vec{g}_i(t) dt$$

By adding  $e^{-(A+B_i)t_i} \vec{\varepsilon}_i$  to both sides and dividing through by  $e^{-(A+B_i)t_{i+1}}$ , we obtain the form:

$$\vec{\varepsilon}_{i+1} = e^{-(A+B_i)h} \vec{\varepsilon}_i + \int_{t_i}^{t_{i+1}} e^{-(A+B_i)(t_{i+1}-t)} \vec{g}_i(t) dt$$

Now let  $\vec{y}_i(t) = e^{-(A+B_i)(t_{i+1}-t)} \vec{g}_i(t)$  be the vector of length  $m+1$  and let  $y_{i,j}(t)$  be the  $j$ th component of  $\vec{y}_i(t)$  for  $j=1, 2, \dots, m+1$ . Notice that  $y_{i,j}(t_{i+\frac{1}{2}}) = 0$  for  $j=1, 2, \dots, m+1$  as  $\vec{g}_i(t_{i+\frac{1}{2}}) = \vec{0}$ . Now consider using the midpoint rule to approximate the integral. We obtain that.

$$\int_{t_i}^{t_{i+1}} y_{i,j}(t) dt = h y_{i,j}(t_{i+\frac{1}{2}}) + \frac{h^3}{24} \frac{d^2}{dt^2} [y_{i,j}(\xi_{i,j})] = \frac{h^3}{24} \frac{d^2}{dt^2} [y_{i,j}(\xi_{i,j})]$$

where  $\xi_{ij} \in [t_i, t_{i+1}]$ . This implies that

$$(\vec{\varepsilon}_{i+1})_j = (e^{-(A+B_i)h} \vec{\varepsilon}_i)_j + \frac{h^3}{24} \frac{d^2}{dt^2} [y_{ij}(\varepsilon_{ij})] \text{ for } j = 1, 2, \dots, m+1. \quad (4.5)$$

Let

$$\tau = \max_{1 \leq j \leq m+1} \max_{0 \leq t, t_i \leq t_f} \frac{1}{24} \left| \frac{d^2}{dt^2} y_{ij}(t) \right|.$$

Then,

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq \|e^{-(A+B_i)h} \vec{\varepsilon}_i\|_{\infty} + h^3 \tau$$

and we can rewrite Eq. (4.5) in the following form:

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq \|e^{-(A+B_i)h}\|_{\infty} \|\vec{\varepsilon}_i\|_{\infty} + h^3 \tau.$$

Then, because  $\|e^{At}\| \leq e^{\|A\|t}$  for  $t \leq 0$  we obtain

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq e^{h\|A+B_i\|_{\infty}} \|\vec{\varepsilon}_i\|_{\infty} + h^3 \tau$$

Now if we let  $k = \max_{0 \leq t \leq t_f} \|A + B(t)\|$  we obtain:

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq e^{kh} \|\vec{\varepsilon}_i\|_{\infty} + h^3 \tau. \quad (4.6)$$

$$\|\vec{\varepsilon}_0\| = 0$$

We can equivalently obtain:

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq h^3 \tau \left( \frac{e^{kh(i+1)} - 1}{e^{kh} - 1} \right)$$

giving us:

$$\|\vec{\varepsilon}_{i+1}\|_{\infty} \leq \frac{h^2 \tau}{k} (e^{kt_f} - 1).$$

Thus, the error in the PCA method is proportional to  $h^2$  as previously stated.

We can most clearly test the error estimate by employing the PCA method to solve a simple problem. Consider, the PCA method applied to the following IVP:

$$\frac{d\vec{y}}{dt} = A(t)\vec{y}$$

$$\vec{y}(0) = [1, 1]^T$$

Table 6  
Order  $h^2$  computational results

h	PCA Approximation	Error
0.1	4.52525	0.000166
0.05	4.52749	0.000042
0.025	4.52788	0.000010

where

$$A(t) = \begin{bmatrix} t-1 & 1 \\ 1 & -1 \end{bmatrix}.$$

This IVP corresponds to the one delayed group, ramp input reactivity with the following parameters:  $\beta = 1$ ,  $\Lambda = 1$  and  $\lambda_1 = 1$ . The reactivity function is  $\rho(t) = t$ . The computational results obtained using the PCA method are given in Table 6.

Clearly, as  $h$  decreases by a factor of two, the error in the PCA approximation decreases by a factor of four. This simple example supports the results of the formal error analysis.

## 5. Conclusions and future work

The PCA method is clearly an effective numerical method for solving the point kinetics equations. The method is efficient, simple, and accurate. The PCA method yields exact results in step reactivity insertions and the method has accuracy of order  $h^2$  for problems with time-varying reactivities.

In possible future work, the reactivity and source functions could perhaps be approximated by piecewise linear functions. The implementation of such a method may be more complicated, but exact results could be obtained in the ramp reactivity case along with the step reactivity case. In addition, provided that the error can be shown to have the correct form, perhaps a Richardson extrapolation procedure can be applied to the PCA method to increase the accuracy to order  $h^3$ .

## Acknowledgements

Partial support for this research was provided by the National Science Foundation Grant DMS-0201105 and the Texas Advanced Research Program Grant ARP 0212-44-1582.

## References

- Aboanber, A.E., Nahla, A.A., 2002. Generalization of the analytical inversion method for the solution of the point kinetics equations. *Journal of Physics A: Mathematical and General* 35, 3245–3263.

- Chao, Y., Attard, A., 1985. A resolution to the stiffness problem of reactor kinetics. *Nuclear Science and Engineering* 90, 40–46.
- Da Nóbrega, J.A.W., 1971. A new solution of the point kinetics equations. *Nuclear Science and Engineering* 46, 366–375.
- Hetrick, D.L., 1971. *Dynamics of Nuclear Reactors*. The University of Chicago Press, Chicago.
- Kinard, M. 2003. *An Efficient Numerical Solution to the Point Kinetics Equation in Nuclear Reactor Dynamics*. Texas Tech University.
- Sánchez, J., 1989. On the numerical solution of the point kinetics equations by generalized Runge-Kutta methods. *Nuclear Science and Engineering* 103, 94–99.