Multilevel Methods to Solve the Neutron Diffusion Equation ⋆

D. Ginestar [a,c], J. Marín [a,c] and G. Verdú [b,c]

[a] *Departament de Matemàtica Aplicada.*
e-mail: {dginesta,jmarinma}@mat.upv.es

[b] *Departament d'Enginyeria Química i Nuclear.*
e-mail: gverdu@pleione.upv.es

[c] *Universitat Politècnica de València. 46071 València, Spain.*

Abstract

In this paper, we develop multilevel strategies to solve the time dependent neutron diffusion equation. These methodologies are based on the discretization of the spatial part of the neutron diffusion equation using a nodal collocation method which makes use of an expansion of the neutron flux in terms of Legendre polynomials. The different levels or grids of the multilevel strategies have been defined in terms of the different number of polynomials used for the nodal collocation method expansions instead of using different nodalization sizes. Results for two transients associated with the bidimensional TWIGL reactor are reported.

Key words: multilevel methods, neutron diffusion equation, iterative methods.

1 Introduction

A nuclear power plant generates electricity using steam. This kind of plants basically consist of a steam supply system, called the reactor core, and a turbo-generator [15]. The primary energy in a nuclear power plant is generated by nuclear reactions induced by neutrons, which take place in the core of the reactor. This primary energy appears as heat that is transferred to water. The heated water is brought to boiling at high temperature and high pressure. The steam produced is then expanded in the turbine producing mechanical work

⋆ This work was supported by the Dirección General de Enseñanza Superior e Investigación Científica under the projects PB97-0334 and 1FD97-0240.

which is used to produce electricity. The steam then condenses into water which is returned to the reactor core as feed water in a closed cycle. A typical scheme for a boiling water reactor is shown in figure reffigreactor.

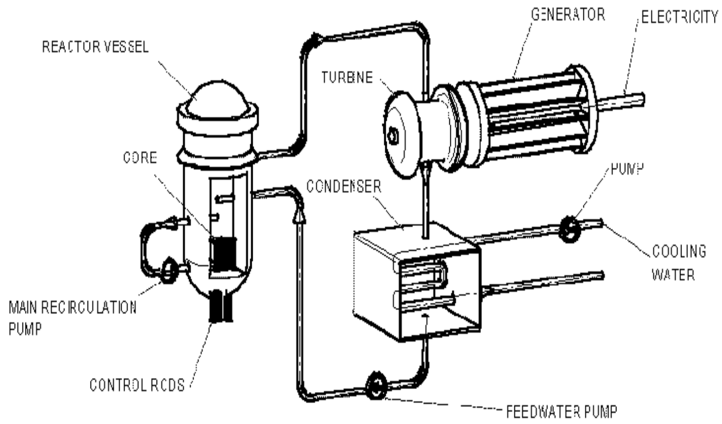

Fig. 1. BWR reactor scheme.

For safety reasons, there is a great interest in developing codes to simulate the nuclear plants behaviour as accurately as possible. These simulators consist mainly of two different blocks which solve the models used for the basic physical phenomena taking place in the plant. In this way, there is a neutronic module which simulates the neutron balance in the reactor core, and a thermal-hydraulics module, which simulates the heat transfer from the fuel to the water used as coolant, and the different condensation and evaporation processes which take place in the reactor core and in the condenser systems. The coupling between these two modules is different for each code depending on the numerical schemes used to integrate the whole set of equations constituting the model of the plant. These kind of nuclear plant simulators have a complicated input data where the geometry of each particular plant is defined.

In this paper, we focus on the neutronic module. To modelize the neutron balance inside the reactor core, generally the two energy groups approximation of the neutron diffusion equation is used [24], and it is assumed that the delayed and fission neutrons are born in the fast group and that there is no up-scattering from the thermal to the fast group. In these conditions, the neutron diffusion equation can be formulated as follows (see [24])

$$[v^{-1}]\dot{\phi} + \mathcal{L}\phi = (1-\beta)\mathcal{M}\phi + \sum_{k=1}^{K_c} \lambda_k \mathcal{C}_k \chi \,, \tag{1}$$

$$\dot{\mathcal{C}}_k = \beta_k [\nu\Sigma_{f_1} \;\; \nu\Sigma_{f_2}]\phi - \lambda_k \mathcal{C}_k ; \quad k = 1, \ldots , K_c \,, \tag{2}$$

where K_c is the number of delayed neutron precursors considered,

$$\mathcal{L} = \begin{bmatrix} -\vec{\nabla}(D_1\vec{\nabla}) + \Sigma_{a1} + \Sigma_{12} & 0 \\ -\Sigma_{12} & -\vec{\nabla}(D_2\vec{\nabla}) + \Sigma_{a2} \end{bmatrix} , \; [v^{-1}] = \begin{bmatrix} \dfrac{1}{v_1} & 0 \\ 0 & \dfrac{1}{v_2} \end{bmatrix} ,$$

and

$$\mathcal{M} = \begin{bmatrix} \nu\Sigma_{f_1} & \nu\Sigma_{f_2} \\ 0 & 0 \end{bmatrix} , \; \phi = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} , \; \chi = \begin{bmatrix} 1 \\ 0 \end{bmatrix} .$$

The boundary conditions for the neutron flux are $\phi|_\Gamma = 0$, where Γ is the reactor boundary. To obtain an approximate solution to equation (1) an spatial and temporal discretization of this equation must be done. For the spatial discretization nodal methods have been widely used. To apply such a method the reactor is divided into relatively large cells or nodes, and the solution is developed in terms of an adequate base of functions in each node. Among the existing nodal methods we have chosen the nodal collocation method [9,21].

1.1 The Nodal Collocation Method

The nodal collocation method developed for the neutron diffusion equation is an adaptation of the classical collocation methods [16] for discretizing partial differential equations, which are based on the expansion of the solution as a linear combination of basis functions. Particularly, the nodal collocation method for the neutron diffusion equation assumes an expansion of the neutron flux in terms of orthonormal Legendre polynomials.

We present the main features of the nodal collocation method considering only one group of energy. The generalization of the approach to the case of two groups of energy is straightforward.

Let us consider a generic parallelepiped element e and rewrite spatial part of equation (1) for this element, in the general form

$$-D_{x,e}\frac{\partial^2 \phi_e}{\partial x^2} - D_{y,e}\frac{\partial^2 \phi_e}{\partial y^2} - D_{z,e}\frac{\partial^2 \phi_e}{\partial z^2} + \Sigma_{re}\phi_e = S_e \,. \tag{3}$$

Performing the change of variables (see figure 2)

$$u = \frac{1}{\triangle x_e}[x - \frac{1}{2}(x_{m-1/2} + x_{m+1/2})]\,, \quad v = \frac{1}{\triangle y_e}[y - \frac{1}{2}(y_{n-1/2} + y_{n+1/2})]\,,$$
$$w = \frac{1}{\triangle z_e}[z - \frac{1}{2}(z_{p-1/2} + z_{p+1/2})]\,,$$

equation (3) is expressed in the generic form

$$-\frac{\triangle y_e \triangle z_e}{\triangle x_e} D_{x,e} \frac{\partial^2 \phi_e}{\partial u^2} - \frac{\triangle x_e \triangle z_e}{\triangle y_e} D_{y,e} \frac{\partial^2 \phi_e}{\partial v^2} - \frac{\triangle x_e \triangle y_e}{\triangle z_e} D_{z,e} \frac{\partial^2 \phi_e}{\partial w^2} + \Sigma_{re} V_e \phi_e = V_e S_e \tag{4}$$

where $V_e = \triangle x_e \triangle y_e \triangle z_e$.

As a trial function in each element e we use the following truncated expansion

$$\phi_e(u,v,w) = \sum_{k_1=0}^{K} \sum_{k_2=0}^{K} \sum_{k_3=0}^{K} \phi_e^{k_1,k_2,k_3} P_{k_1}(u) P_{k_2}(v) P_{k_3}(w)\,, \tag{5}$$

where $P_k(u)$ are orthonormal Legendre polynomials in the interval $[-1/2, 1/2]$. We consider, in each direction, $K+1$ Legendre polynomials with degrees from 0 to K.

Using the expansion (5) in equation (4), multiplying this equation by the weight function

$$W_{l_1,l_2,l_3}(u,v,w) = P_{l_1}(u) P_{l_2}(v) P_{l_3}(w)$$

and integrating over the element volume, we obtain

$$-\triangle y_e \triangle z_e F_{e,x}^{k_1,k_2,k_3} - \triangle x_e \triangle z_e F_{e,y}^{k_1,k_2,k_3} - \triangle x_e \triangle y_e F_{e,z}^{k_1,k_2,k_3} + V_e \Sigma_{re} \phi_e^{k_1,k_2,k_3} = V_e S_e^{k_1,k_2,k_3}\,, \tag{6}$$

where

$$\begin{aligned} F_{e,x}^{k_1,k_2,k_3} &= \frac{D_{x,e}}{\triangle x_e} L_{k_1}\{\phi_{e,x}^{k_2,k_3}(u)\} \quad , \\ F_{e,y}^{k_1,k_2,k_3} &= \frac{D_{y,e}}{\triangle y_e} L_{k_2}\{\phi_{e,y}^{k_1,k_3}(v)\} \quad , \\ F_{e,z}^{k_1,k_2,k_3} &= \frac{D_{z,e}}{\triangle z_e} L_{k_3}\{\phi_{e,z}^{k_1,k_2}(w)\} \quad . \end{aligned} \tag{7}$$

The functions $\phi_{e,x}^{k_2,k_3}(u), \phi_{e,y}^{k_1,k_3}(v), \phi_{e,z}^{k_1,k_2}(w)$ are, respectively,

$$\phi_{e,x}^{k_2,k_3}(u) = \sum_{k=0}^{K} \phi_e^{k,k_2,k_3} P_k(u) \quad ,$$
$$\phi_{e,y}^{k_1,k_3}(v) = \sum_{k=0}^{K} \phi_e^{k_1,k,k_3} P_k(v) \quad ,$$
$$\phi_{e,z}^{k_1,k_2}(w) = \sum_{k=0}^{K} \phi_e^{k_1,k_2,k} P_k(w) \quad ,$$

and

$$\begin{aligned} L_k(f(u)) = & \int_{-1/2}^{1/2} du\, P_k(u) \frac{d^2}{du^2} f(u) = \\ & = \sqrt{2k+1} \left\{ (-1)^{k+1} \left[k(k+1) f(-1/2) + \frac{d}{du} f(-1/2) \right] - \right. \\ & - \left[k(k+1) f(1/2) - \frac{d}{du} f(1/2) \right] + \\ & \left. + \sum_{l=0}^{K-2} [1 + (-1)^{k+l}] \sqrt{2l+1} [k(k+1) - l(l+1)] F_l \right\} \quad , \end{aligned} \tag{8}$$

where F_l are the Legendre coefficients of the expansion of the function $f(u)$.

To impose continuity conditions, we attribute numbers e_1 to e_6 to the six neighboring elements which share surface with the element e.

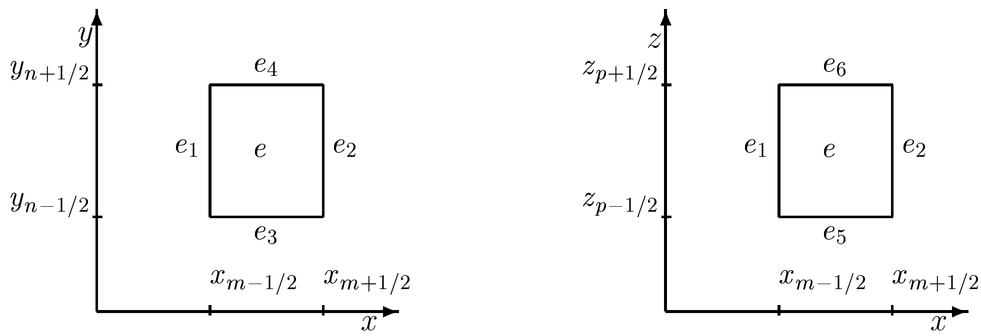

Fig. 2. Position of neighboring elements of element e.

If, for example, we consider the interface between the element e and element e_1, the continuity conditions for the flux and the current over the interface are

$$\phi_{e_1}(\frac{1}{2}, v, w) = \phi_e(-\frac{1}{2}, v, w) \quad ,$$
$$\frac{D_{x,e_1}}{\triangle x_{e_1}} \frac{\partial}{\partial u} \phi_{e_1}(\frac{1}{2}, v, w) = \frac{D_{x,e}}{\triangle x_e} \frac{\partial}{\partial u} \phi_e(-\frac{1}{2}, v, w) \quad .$$

Multiplying by the weight function $W = P_{k_2}(v) P_{k_3}(w)$ and integrating over the common surface, these expressions can be expressed in the form

$$
\begin{aligned}
\phi_{e_1,x}^{k_2,k_3}(\frac{1}{2}) &= \phi_{e,x}^{k_2,k_3}(-\frac{1}{2}) \quad , \\
\frac{D_{x,e_1}}{\triangle x_{e_1}}\frac{d}{du}\phi_{e_1,x}^{k_2,k_3}(\frac{1}{2}) &= \frac{D_{x,e}}{\triangle x_e}\frac{d}{du}\phi_{e,x}^{k_2,k_3}(-\frac{1}{2}) \quad .
\end{aligned} \tag{9}
$$

Similar conditions can be obtained for all other interfaces of the element e. Notice that these kind of conditions are weaker than the continuity conditions for fluxes and currents over the interfaces.

Making use of the result (8) we can express $F_{e,\alpha}^{k,k_2,k_3}$, $\alpha = x, y, z$ of equation (6) in the form [9,21]

$$
\begin{aligned}
F_{e,x}^{k,k_2,k_3} &= \sum_{l=0}^{K-1}(A_{e,x}^{k,l;K}\phi_{e1}^{l,k_2,k_3} - B_{e,x}^{k,l;K}\phi_{e}^{l,k_2,k_3} + C_{e,x}^{k,l;K}\phi_{e2}^{l,k_2,k_3})\,, \\
F_{e,y}^{k_1,k,k_3} &= \sum_{l=0}^{K-1}(A_{e,y}^{k,l;K}\phi_{e3}^{k_1,l,k_3} - B_{e,y}^{k,l;K}\phi_{e}^{k_1,l,k_3} + C_{e,y}^{k,l;K}\phi_{e4}^{k_1,l,k_3})\,, \\
F_{e,z}^{k_1,k_2,k} &= \sum_{l=0}^{K-1}(A_{e,z}^{k,l;K}\phi_{e5}^{k_1,k_2,l} - B_{e,z}^{k,l;K}\phi_{e}^{k_1,k_2,l} + C_{e,z}^{k,l;K}\phi_{e6}^{k_1,k_2,l})\,,
\end{aligned} \tag{10}
$$

with the coefficients

$$
\begin{aligned}
A_{e,\alpha}^{k,l;K} &= \frac{(-1)^k}{2K(K+1)}\sqrt{2k+1}\sqrt{2l+1}[K(K+1)-k(k+1)][K(K+1)-l(l+1)]W_{e,\alpha}^{-} \quad , \\
C_{e,\alpha}^{k,l;K} &= \frac{(-1)^l}{2K(K+1)}\sqrt{2k+1}\sqrt{2l+1}[K(K+1)-k(k+1)][K(K+1)-l(l+1)]W_{e,\alpha}^{+} \quad , \\
B_{e,\alpha}^{k,l;K} &= \frac{\sqrt{2k+1}\sqrt{2l+1}}{K(K+1)}\left\{\frac{D_{\alpha,e}}{\triangle\alpha_e}[1+(-1)^{k+l}][K(K+1)-k(k+1)]l(l+1)\right. \\
&\left. + \frac{1}{2}[K(K+1)-k(k+1)][K(K+1)-l(l+1)][(-1)^{k+l}W_{e,\alpha}^{-}+W_{e,\alpha}^{+}]\right\} \quad , \\
&\text{if } l < k \quad ,
\end{aligned}
$$

and

$$
\begin{aligned}
B_{e,\alpha}^{k,l;K} &= \frac{\sqrt{2k+1}\sqrt{2l+1}}{K(K+1)}\left\{\frac{D_{\alpha,e}}{\triangle\alpha_e}k(k+1)[1+(-1)^{k+l}][K(K+1)-l(l+1)] + \right. \\
&\left. + \frac{1}{2}[K(K+1)-k(k+1)][K(K+1)-l(l+1)][(-1)^{k+l}W_{e,\alpha}^{-}+W_{e,\alpha}^{+}]\right\} \quad , \\
&\text{if } l \geq k \quad ,
\end{aligned}
$$

where the centered finite difference coupling factors are defined as:

$$
\begin{aligned}
W_{e,x}^{-} &= W_{e_1,x}^{+} = 2D_{x,e}D_{x,e_1}(\triangle x_e D_{x,e_1} + \triangle x_{e_1} D_{x,e})^{-1} \quad , \\
W_{e,x}^{+} &= W_{e_2,x}^{-} = 2D_{x,e}D_{x,e_2}(\triangle x_e D_{x,e_2} + \triangle x_{e_2} D_{x,e})^{-1} \quad , \\
W_{e,y}^{-} &= W_{e_3,y}^{+} = 2D_{y,e}D_{y,e_3}(\triangle y_e D_{y,e_3} + \triangle y_{e_3} D_{y,e})^{-1} \quad , \\
W_{e,y}^{+} &= W_{e_4,y}^{-} = 2D_{y,e}D_{y,e_4}(\triangle y_e D_{y,e_4} + \triangle y_{e_4} D_{y,e})^{-1} \quad , \\
W_{e,z}^{-} &= W_{e_5,y}^{+} = 2D_{z,e}D_{z,e_5}(\triangle z_e D_{z,e_5} + \triangle z_{e_5} D_{z,e})^{-1} \quad , \\
W_{e,z}^{+} &= W_{e_6,z}^{-} = 2D_{z,e}D_{z,e_6}(\triangle z_e D_{z,e_6} + \triangle z_{e_6} D_{z,e})^{-1} \quad .
\end{aligned}
$$

These coefficients for the reactor boundary elements take the form $W_{e,\alpha}^{-} = \dfrac{2D_{\alpha,e}}{\triangle \alpha_e}$, if the left surface of element e is a zero flux boundary, and $W_{e,\alpha}^{+} = \dfrac{2D_{\alpha,e}}{\triangle \alpha_e}$, if the right surface of node e is a zero flux boundary. Similarly, $W_{e,\alpha}^{-} = 0$, if the left surface is a zero current boundary, and $W_{e,\alpha}^{+} = 0$, if the right surface is a zero current boundary.

In this nodal collocation method, it is possible to introduce consistently a serendipity approximation [9]. This approximation is applied directly to the nodal collocation method replacing equation (10) by

$$
\begin{aligned}
F_{e,x}^{k,k_2,k_3} &= \sum_{l=0}^{K-1-k_2-k_3} (A_{e,x}^{k,l;K-k_2-k_3}\phi_{e1}^{l,k_2,k_3} - B_{e,x}^{k,l;K-k_2-k_3}\phi_{e}^{l,k_2,k_3} + C_{e,x}^{k,l;K-k_2-k_3}\phi_{e2}^{l,k_2,k_3}) \, , \\
F_{e,y}^{k_1,k,k_3} &= \sum_{l=0}^{K-1-k1-k_3} (A_{e,y}^{k,l;K-k_1-k_3}\phi_{e3}^{k_1,l,k_3} - B_{e,y}^{k,l;K-k_1-k_3}\phi_{e}^{k_1,l,k_3} + C_{e,y}^{k,l;K-k_1-k_3}\phi_{e4}^{k_1,l,k_3}) \, , \\
F_{e,z}^{k_1,k_2,k} &= \sum_{l=0}^{K-1-k_1-k_2} (A_{e,z}^{k,l;K-k_1-k_2}\phi_{e5}^{k_1,k_2,l} - B_{e,z}^{k,l;K-k_1-k_2}\phi_{e}^{k_1,k_2,l} + C_{e,z}^{k,l;K-k_1-k_2}\phi_{e6}^{k_1,k_2,l}) \, ,
\end{aligned}
\tag{11}
$$

with $k_1 = 0, \ldots, K-1$; $k_2 = 0, \ldots K-1-k_1$; $k_3 = 0, \ldots K-1-k_1-k_2$.

With an appropriate ordering for the indices k_1, k_2, k_3, and establishing an order for the nodes of the reactor core, we define a vector ψ constituted by the unknowns $\phi_e^{k_1,k_2,k_3}$ of the nodal collocation method for all the nodes of the reactor discretization. In this way the initial problem defined by (1) and (2) is approximated by the following system of ordinary differential equations

$$
[v^{-1}]\dot{\psi} + L\psi = (1-\beta)M\psi + \sum_{k=1}^{K_c} \lambda_k X C_k \, , \tag{12}
$$

$$
\dot{C}_k = \beta_k [M_{11} \;\; M_{12}]\psi - \lambda_k C_k; \quad k = 1, \ldots, K_c \, , \tag{13}
$$

where L, M, and $[v^{-1}]$ are matrices with the following block structure

$$L = \begin{bmatrix} L_{11} & 0 \\ -L_{21} & L_{22} \end{bmatrix}, \quad M = \begin{bmatrix} M_{11} & M_{12} \\ 0 & 0 \end{bmatrix}, \quad [v^{-1}] = \begin{bmatrix} v_1^{-1} & 0 \\ 0 & v_2^{-1} \end{bmatrix}, \quad X = \begin{bmatrix} I \\ 0 \end{bmatrix}. \tag{14}$$

1.2 Temporal discretization

For the time dependent part, the set of ordinary differential equations (12) and (13) are integrated over a series of time steps $[t_n, t_{n+1}]$. To simplify the calculations, equation (13) is integrated under the assumption that the term $[M_{11}M_{12}]\psi$ varies linearly from t_n to t_{n+1}, obtaining the solution C_k at t_{n+1} expressed as [22]

$$C_k^{n+1} = C_k^n e^{-\lambda_k h} + \beta_k (a_k [M_{11}M_{12}]^n \psi^n + b_k [M_{11}M_{12}]^{n+1} \psi^{n+1}) \,, \tag{15}$$

where $h = t_{n+1} - t_n$, and the coefficients a_k and b_k are given by

$$a_k = \frac{(1+\lambda_k h)(1-e^{-\lambda_k h})}{\lambda_k^2 h} - \frac{1}{\lambda_k}, \quad b_k = \frac{\lambda_k h - 1 + e^{-\lambda_k h}}{\lambda_k^2 h}.$$

To integrate equation (12), we must take into account that it constitutes a system of stiff differential equations, mainly due to the elements of the diagonal matrix $[v^{-1}]$. This requires the use of an implicit backward difference formula (BDF) for its the integration (see [22]), resulting in a system of linear algebraic equations to solve at each time step. A stable one-step BDF to integrate (12) is given by

$$\frac{[v^{-1}]}{h} \left(\psi^{n+1} - \psi^n\right) + L^{n+1}\psi^{n+1} = (1-\beta) M^{n+1} \psi^{n+1} + \sum_{k=1}^{K_c} \lambda_k X C_k^{n+1}. \tag{16}$$

Taking into account equation (15) and the structure of matrices L and M, we rewrite (16) as the system of linear equations

$$\begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} \psi_1^{n+1} \\ \psi_2^{n+1} \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix} \begin{bmatrix} \psi_1^n \\ \psi_2^n \end{bmatrix} + \sum_{k=1}^{K_c} \lambda_k e^{-\lambda_k h} \begin{bmatrix} C_k^n \\ 0 \end{bmatrix} \,, \tag{17}$$

where

$$
\begin{aligned}
T_{11} &= \frac{I}{h}\upsilon_1^{-1} + L_{11}^{n+1} - (1-\beta)M_{11}^{n+1} - \sum_{k=1}^{K_c} \lambda_k \beta_k b_k M_{11}^{n+1} \ , \quad T_{21} = -L_{21}^{n+1} \ , \\
T_{12} &= -(1-\beta)M_{12}^{n+1} - \sum_{k=1}^{K_c} \lambda_k \beta_k b_k M_{12}^{n+1} \ , \quad T_{22} = \frac{I}{h}\upsilon_2^{-1} + L_{22}^{n+1} \ , \\
R_{11} &= \frac{I}{h}\upsilon_1^{-1} + \sum_{k=1}^{K_c} \lambda_k \beta_k a_k M_{11}^{n} \ , \quad R_{12} = \sum_{k=1}^{K_c} \lambda_k \beta_k a_k M_{12}^{n} \ , \quad R_{22} = \frac{I}{h}\upsilon_2^{-1} \ .
\end{aligned}
\tag{18}
$$

We have to note that matrices L^{n+1} and M^{n+1} depend on the nuclear cross section values at the instant t_{n+1}, and these values are dependent on the control rod positions and the thermal-hydraulic feedback which also depends on the neutronic flux. Depending on the numerical discretization used, these dependencies can lead to a system of nonlinear equations for the neutronic flux coefficients which should be solved using an iterative method, integrating the neutron diffusion equation and the thermal-hydraulic equations altogether.

As a first approach to study the performance of the numerical methods developed to integrate the neutron diffusion equation, we will not consider the thermal-hydraulic coupling with should be carried out with the thermal-hydraulic module of the reactor simulator, and we will suppose that the time evolution of the nuclear cross sections is known. Hence, for each time step it will be necessary to solve a large and sparse system of linear equations with the following block structure:

$$
\begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} . \tag{19}
$$

This system will also be denoted by

$$
T\psi = E. \tag{20}
$$

1.3 Solution of the linear systems

Since the matrix of coefficients in (19) is non-symmetric, to solve this system of equations it is necessary to use a suitable iterative method for this kind of matrices. We have employed two different methods. The first one corresponds to that used in [22], where the system (19) is solved with a block stationary iterative method with the following structure

$$
\begin{aligned}
T_{11}\psi_1^{\ell+1} &= E_1 - T_{12}(\omega\psi_2^{\ell} + (1-\omega)\psi_2^{\ell-1}) \ , \\
T_{22}\psi_2^{\ell+1} &= E_2 - T_{21}(\omega\psi_1^{\ell+1} + (1-\omega)\psi_1^{\ell}) \ ,
\end{aligned}
\tag{21}
$$

where ω is a relaxation factor, typically chosen as $w = 0.5$. In [6] this iterative method is accelerated with the variational technique given by

$$
\psi^{n+1} = \psi^n + \alpha r^n + \beta d^n, \tag{22}
$$

where $r^n = E - T\psi^n$ and $d^n = \psi^n - \psi^{n-1}$ are the residual and the previous change in ψ at step n respectively. The parameters α and β are chosen such that $\|r^n\|_2^2$ is minimized. The second method used is the well known BiCGSTAB method [20], [3], preconditioned with the incomplete LU preconditioner ILU0 [17], applied to the diagonal blocks T_{11} and T_{22}. This method is based on the biconjugate gradient (BCG) algorithm. The BCG was first proposed by Lanczos [12] in 1952 and then modified by Fletcher [5] in 1974, and it is based on the Lanczos algorithm for solving nonsymmetric linear systems. The main disadvantage of the method is that each step of the BCG algorithm requires a matrix-by-vector product with both T and T^T. Among the existing methods to avoid the product with T^T, the BiCGSTAB algorithm is widely used since it presents smooth convergence on the residual norm compared with other methods as the Conjugate Gradient Squared (CGS) [19].

2 Multilevel Algorithms

To analyze transients where large neutronic flux gradients appear, it is necessary to use an accurate spatial discretization. The accuracy of the nodal collocation method is improved increasing the number of Legendre polynomials used for the discretization. When a large number of polynomials is used, multilevel techniques to integrate the neutron diffusion equation can be applied. These multilevel methods are based on the calculation of successive approximations for the solution computed on coarser grids. Normally, coarser grids are associated with larger nodalizations sizes, but we will associate here the different grids or levels of the algorithm with the number of polynomials, K, used for the nodal collocation method, as it will be explained below.

We will review multilevel methodology basics and we shall describe two multilevel algorithms for the nodal collocation method. A complete exposition of the multilevel concepts can be found for instance in [2], [7] and [11], [14].

Let us consider a system of linear equations like (20)

$$T\psi = E, \tag{23}$$

where T is an $n \times n$ matrix. We can think T as arising from the discretization of a partial differential equation on a given domain Ω. For simplicity, we assume that Ω is discretized on a uniform grid Ω^h, where h represents the mesh size. To solve (23) using an iterative technique, one typically calculates successive approximations to the exact solution, ψ, of the form

$$\psi^{new} = G(T, \psi^{old}, E), \tag{24}$$

where G is some expression involving the old assignment of ψ. One natural way to improve this process is to obtain better initial guesses (closer to the solution) at a computational cost as low as possible. To accomplish with this aim, a nested sequence of grids Ω^k, $k = 1, \ldots, K-1$ can be used to generate improved initial guesses to the solution in the finest grid Ω^K. We will use the following notation. Ω^k, $k = 1, \ldots, K$ denotes the grid at coarse level k, such that $\Omega^1 \subset \Omega^2 \subset \cdots \subset \Omega^K$. At each coarse level a system of linear equations of the form

$$T^k \psi^k = E^k, \tag{25}$$

is solved, where T^k can be seen as the Ω^k version of $T^K = T$. To use a vector ψ^k as an initial guess for the problem associated with the finer grid $\Omega^{k'}$, $k' > k$, we define the linear operator called *interpolation* or *prolongation* operator $I_k^{k'} : \Omega^k \to \Omega^{k'}$. Thus, $\psi^{k'} = I_k^{k'} \psi^k$. Similarly, to represent a vector $\psi^{k'}$ on a coarser grid Ω^k, $k < k'$, we use the *restriction* operator $I_{k'}^k : \Omega^{k'} \to \Omega^k$.

As we have already mentioned, to develop the multilevel algorithm for the nodal collocation method we will use different coarse levels, or different grids, depending on the number of Legendre polynomials used for the discretization. At each level the number of cells or nodes in which the reactor has been discretized remains constant and what changes is the number of polynomials used for the nodal collocation method expansions. If we want to solve the neutron diffusion equation using K Legendre polynomials we perform for each time step, t_{n+1}, a nested iteration taking as initial guess the Legendre coefficients vector, $\psi^K(t_n)$, i.e., the solution in the previous time step. The nested iteration can be structured as follows.

Algorithm 1: *The nested iteration.*
On the coarsest grid
  Restrict $\psi_0^{k_{start}}(t_{n+1}) = I_K^{k_{start}} \psi^K(t_n)$.
  Solve $T^{k_{start}} \psi^{k_{start}}(t_{n+1}) = E^{k_{start}}$ *using as initial guess* $\psi_0^{k_{start}}(t_{n+1})$.
For $k = k_{start}, K-1$ **do**
  Interpolate $\psi_0^{k+1}(t_{n+1}) = I_k^{k+1} \psi^k(t_{n+1})$.
  Solve $T^{k+1} \psi^{k+1}(t_{n+1}) = E^{k+1}$ *using as initial guess* $\psi_0^{k+1}(t_{n+1})$.

End do.

In order to complete the algorithm, we still have to define the restriction and interpolation operators. Let us suppose that we want to determine the neutronic flux using ℓ Legendre polynomials. For each node e of the reactor, if we use the nodal collocation method with the serendipity approximation, we have the flux expansion given by

$$\phi_e^\ell(u,v,w) = \sum_{i=0}^{\ell-1} \sum_{j=0}^{\ell-1-i} \sum_{k=0}^{\ell-1-i-j} \phi_e^{\ell;i,j,k} P_i(u) P_j(v) P_k(w)\,, \tag{26}$$

and the vector ψ^ℓ is formed by the coefficients $\phi_e^{\ell;i,j,k}$ for all the nodes of the reactor. The interpolation of a vector ψ^ℓ from grid Ω^ℓ to grid $\Omega^{\ell'}$, i.e., $\psi^{\ell'} = I_\ell^{\ell'} \psi^\ell$ with $\ell' > \ell$, is defined for each node as

$$\phi_e^{\ell';i,j,k}(t_{n+1}) = \begin{cases} \phi_e^{\ell;i,j,k}(t_{n+1}) \text{ if } & i+j+k < \ell - 1\,, \\ \phi_e^{\ell';i,j,k}(t_n) \quad \text{if } & i+j+k \geq \ell - 1\,, \end{cases} \tag{27}$$

with $i = 0, \ldots, \ell' - 1$, $j = 0, \ldots, \ell' - 1 - i$ and $k = 0, \ldots, \ell' - 1 - i - j$. Similarly, the restriction of a vector $\psi^{\ell'}$ from grid $\Omega^{\ell'}$ to grid Ω^ℓ with $\ell' > \ell$ is defined as

$$\phi_e^{\ell;i,j,k}(t_{n+1}) = \phi_e^{\ell';i,j,k}(t_{n+1})\,, \tag{28}$$

with $i = 0, \ldots, \ell - 1$, $j = 0, \ldots, \ell - 1 - i$ and $k = 0, \ldots, \ell - 1 - i - j$. We remark that the matrix T^{k+1} appearing in the Algorithm 1, can be understood in two different ways. In the first one, it is understood as the matrix T in (19) obtained using $k + 1$ Legendre polynomials. Thus, the solution $\psi^{k+1}(t_{n+1})$ obtained corresponds to the solution of the system at time t_{n+1} using $k + 1$ polynomials with the nodal collocation method. We will refer to this strategy as the geometric multilevel (GML) strategy. On the other hand, taking an algebraic approach, the matrix T can be seen as the restriction of the matrix T^K from grid Ω^K to grid Ω^{k+1}, that is,

$$T^{k+1} = I_K^{k+1} T^K I_{k+1}^K\,. \tag{29}$$

This operation mainly consists in selecting from matrix T^K the elements corresponding to grid Ω^{k+1}. Using matrix T^{k+1} of (29) we obtain what we have called the algebraic multilevel algorithm (AML). We have to remark that matrix T^{k+1} in (29) differs from the matrix obtained in the discretization of the neutron diffusion equation using $k + 1$ polynomials, which is the matrix used in the geometric multilevel method.

3 Numerical Experiments

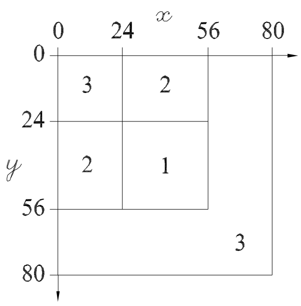

Fig. 3. Quadrant of the TWIGL reactor.

In order to show the performance of the multilevel methods exposed in section 2, we have analyzed two transients of the bidimensional TWIGL reactor which geometry is shown in figure 3 [8,18,13]. This reactor is modeled with two energy groups, one delayed neutron precursor group, and a quarter core symmetry. The two group nuclear constants for the problem are given in table 1.

Region	Group	D_g(cm)	$\Sigma_{ag}(\text{cm}^{-1})$	$\nu\Sigma_{fg}$	Σ_{12}
1	1	1.4	0.01	0.007	0.01
	2	0.4	0.15	0.2	
2	1	1.4	0.01	0.007	0.01
	2	0.4	0.15	0.2	
3	1	1.3	0.008	0.003	0.01
	2	0.5	0.05	0.06	

β_1	λ_1	$\frac{1}{v_1}$	$\frac{1}{v_2}$
0.0064	0.08	10^{-7}	10^{-5}

Table 1
Group constants for the TWIGL reactor.

We have analyzed two different transients for the TWIGL reactor. The first transient consists of decreasing the thermal absorption cross section in region 1 as the ramp function

$$\Sigma_{a2}(t) = 0.15 - \frac{0.0035}{0.2}t \ .$$

The transient is followed during 0.2 seconds.

With the numerical discretizations exposed above, we obtain the coefficients vector ψ. With vector ψ we reconstruct the fast and thermal neutronic fluxes ϕ_1, ϕ_2, and with these fluxes we obtain the relative power for each time step

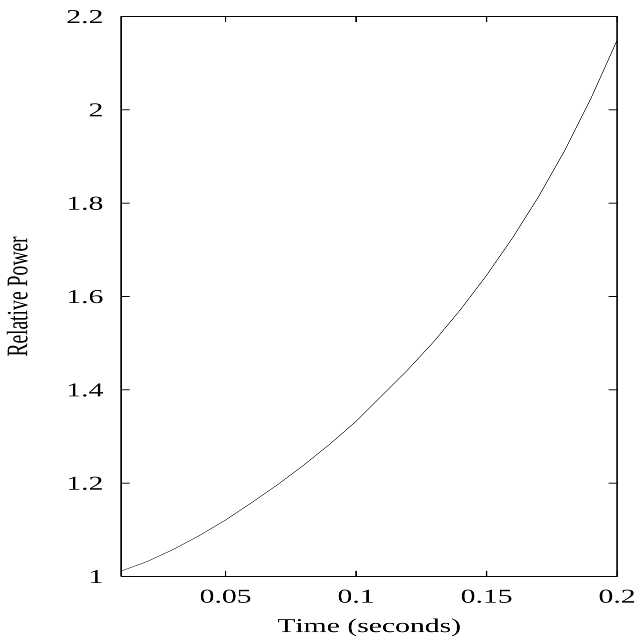

Fig. 4. Relative power evolution for TWIGL transient 1.

[10]. The typical relative power evolution for this transient is shown in figure 4.

For the second transient is obtained postulating a evolution of the thermal absorption cross section as

$$\Sigma_{a2}(t) = \begin{cases} 0.15 - \dfrac{0.0035}{0.2}t, & t \leq 0.2 \\ \\ 0.15 + \dfrac{0.0035}{0.2}t, & 0.2 < t \leq 0.4 \\ \\ 0.15, & t > 0.4 \end{cases}$$

The transient is followed during 0.5 seconds. The typical relative power evolution for this transient is shown in figure 5.

The code has been written in FORTRAN and it has been run on a S-CLASS EXEMPLAR. In tables 2 to 7 we show the CPU time spent to simulate the TWIGL transients using different time steps. Tables 2 and 3 show the time spent running the algorithm without the multilevel approach for a number of 5 Legendre polynomials. This algorithm is called NOBACK1 [6].

Tables 4 and 5 show the time spent by the two different multilevel techniques, geometric (GML)and algebraic (AML), for the first transient.

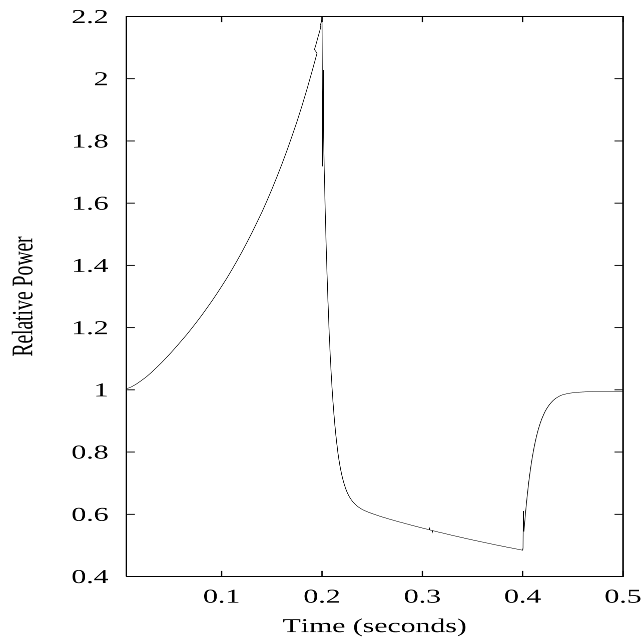

Fig. 5. Relative power evolution for TWIGL transient 2.

Table 2
CPU time (in seconds) spent by NOBACK1 to simulate transient 1.

h (s)	*time (s)*	*solver*
0.001	177	Second Degree
0.005	114	Second Degree
0.01	92	Second Degree
0.001	293	BiCGSTAB
0.005	79	BiCGSTAB
0.01	44	BiCGSTAB

Similarly, tables 6 and 7 show the time for the second transient.

In all the tables h corresponds to the time step used for the integration of the transient in seconds, *time* to the CPU time spent to simulate all the transient in seconds and *solver* the method used to solve the linear system at each time step, either the second degree or the BiCGSTAB methods. For the geometric and algebraic multilevel algorithms it is also shown the number of polynomials, k, used for the multilevel cycle. In this way, the cycle i, j, k means that the multilevel algorithm starts from the initial approximation in the i-th grid in the previous time step, then the problem is solved in the grid j-th, after this the problem is solved in the grid k-th.

With respect to the use of GML method we observe that for small time steps, 0.001 and 0.005 seconds, since the solution of the linear systems takes a small

Table 3
CPU time (in seconds) spent by NOBACK1 to simulate transient 2.

h(s)	*time(s)*	*solver*
0.001	521	Second Degree
0.005	252	Second Degree
0.01	184	Second Degree
0.001	555	BiCGSTAB
0.005	155	BiCGSTAB
0.01	94	BiCGSTAB

Table 4
CPU time (in seconds) spent by GML to simulate transient 1.

h (s)	*time (s)*	*cycle*	*solver*
0.001	236	5,2,3,4,5	Second Degree
0.001	150	5,3,5	Second Degree
0.005	61	5,2,3,4,5	Second Degree
0.005	50	5,3,5	Second Degree
0.01	31	5,2,3,4,5	Second Degree
0.001	298	5,2,3,4,5	BiCGSTAB
0.001	228	5,3,5	BiCGSTAB
0.005	75	5,2,3,4,5	BiCGSTAB
0.005	55	5,3,5	BiCGSTAB
0.01	36	5,2,3,4,5	BiCGSTAB

number of iterations with NOBACK1, the use of GML with long multilevel cycle leads to worse simulation times than the algorithm without the multilevel approach, as it can be observed in tables 4 and 6. This is due to the fact that only an small reduction on the number of iterations necessary for the convergence on the finest grid (5 Legendre polynomials) can be obtained, and the work spent by the multilevel cycle can not be compensated. In these cases, it was necessary the use of a shorter cycle $(5, 3, 5)$ to beat the time spent by NOBACK1.

In contrast, AML method generally obtains worse simulation times than the method NOBACK1. Only when the solution of the linear systems was obtained using the Second Degree method, and a long integration time step was used, the AML method was able to obtain smaller simulation times than the NOBACK1 algorithm, as it can be observed in tables 5 and 7. As it was pointed out above for the GML method, the use of shorter multilevel cycles

Table 5
CPU time (in seconds) spent by AML to simulate transient 1.

h (s)	*time (s)*	*cycle*	*solver*
0.001	353	5,2,3,4,5	Second Degree
0.001	312	5,3,5	Second Degree
0.005	95	5,2,3,4,5	Second Degree
0.005	70	5,3,5	Second Degree
0.01	35	5,2,3,4,5	Second Degree
0.001	398	5,2,3,4,5	BiCGSTAB
0.001	350	5,3,5	BiCGSTAB
0.005	107	5,2,3,4,5	BiCGSTAB
0.005	87	5,3,5	BiCGSTAB
0.01	51	5,2,3,4,5	BiCGSTAB

Table 6
CPU time (in seconds) spent by GML to simulate transient 2.

h(s)	*time(s)*	*cycle*	*solver*
0.001	674	5,2,3,4,5	Second Degree
0.001	387	5,3,5	Second Degree
0.005	151	5,2,3,4,5	Second Degree
0.005	108	5,3,5	Second Degree
0.01	76	5,2,3,4,5	Second Degree
0.001	713	5,2,3,4,5	BiCGSTAB
0.001	561	5,3,5	BiCGSTAB
0.005	152	5,2,3,4,5	BiCGSTAB
0.005	154	5,3,5	BiCGSTAB
0.01	77	5,2,3,4,5	BiCGSTAB

for small integration time steps translated into smaller simulation times.

On the other hand, comparing the Second Degree and the BiCGSTAB methods, it can be observed that for the multilevel algorithms, the use of the Second Degree method leads to better simulation times. In contrast for the NOBACK1 method this is only true for the smallest time step simulated, i.e, 0.001 seconds.

It can be concluded that GML method combined with the Second Degree

Table 7
CPU time (in seconds) spent by AML to simulate transient 2.

h(s)	*time(s)*	*cycle*	*solver*
0.001	920	5,2,3,4,5	Second Degree
0.001	795	5,3,5	Second Degree
0.005	224	5,2,3,4,5	Second Degree
0.005	189	5,3,5	Second Degree
0.01	111	5,2,3,4,5	Second Degree
0.001	925	5,2,3,4,5	BiCGSTAB
0.001	834	5,3,5	BiCGSTAB
0.005	227	5,2,3,4,5	BiCGSTAB
0.005	208	5,3,5	BiCGSTAB
0.01	127	5,2,3,4,5	BiCGSTAB

method for solving the linear systems is the best algorithm to simulate both transients, independently of the time step used to integrate the neutron diffusion equation.

4 Conclusions

In this paper, we have presented two multilevel strategies for the integration of the time dependent neutron diffusion equation, called geometric multilevel method (GML) and algebraic multilevel method (AML). These methods are based on the nodal collocation method for the discretization of the spatial part of the neutron diffusion equation and relies on the selection of different levels or grids depending on the number of polynomials used for the polynomials expansions in the nodal collocation method.

To investigate the performance of the methodologies, two transients for the TWIGL reactor have been analyzed and it has been shown that the multilevel techniques are advantageous, specially for the geometric approach, when compared with the algorithm NOBACK1 which does not use any multilevel strategy. This behaviour becomes even more significant when a large integration time step and a short multilevel cycle are used.

On the other hand, the multilevel methodology provides the basic tools to implement integration methods which can vary the spatial accuracy at each time step. In this way, if we use a multilevel strategy, it is possible to implement a control for the spatial accuracy similar to the integration time step control already implemented in most of the production simulators. This control would improve the performance of nuclear plant simulators to analyze transients where strong variations of the neutronic flux take place.

References

[1] Beam, T.M., Ivanov, K.N., Baratta, A.J., Finnemann, H., Nodal Kinetics Model Upgrade in the Penn State coupled TRAC/NEM Codes. Ann. Nucl. Energy, 1999, 26:1205-1219.

[2] Briggs, W.L., A Multigrid Tutorial. SIAM, Philadelphia, 1987.

[3] Chen, G.S., Christenson, J.M., Yang, D.Y., Application of the Preconditioned Transpose-Free Quasi-Minimal Residual Method for Two-Group Reactor Kinetics. Ann. Nucl. Energy, 1996, 24(5):339-359.

[4] Finnemann, H., Bennewitz, F., Wagner, M.R. Interface Current Techniques for Multidimensional Reactor Calculations, Atomkernenergie (ATKE), 1977, 30:123-128.

[5] Fletcher, R., Conjugate gradient methods for indefinite systems. Proceedings of the Dundee Biennal Conference on Numerical Analysis 1974. In G. A. Watson, editor, Springer Verlag, New York, 1975, 73–89.

[6] Ginestar, D., *et al.* High Order Backward Discretization for the Neutron Diffusion Equation. Ann. Nucl. Energy, 1998, 25(1-3):47-64.

[7] Hackbusch, W., Trottenberg, U., Multigrid Methods. Springer-Verlag, Berlin, 1982.

[8] Hageman, L.A., Yasinski, J.B., Comparison of alternating-direction time-differencing methods and other implicit methods for the solution of the neutron group diffusion equations. Nucl. Sci. Eng., 1969, 38(8).

[9] Hébert, A., Development of the nodal collocation method for solving the neutron diffusion equation. Ann. Nucl. Energy, 1987, 14(10):527-541.

[10] Henry, A. F., Nuclear-Reactor Analysis. The MIT Press, 1982.

[11] Jespersen D., Multigrid methods for partial differential equations. In studies in Numerical Analysis, Volume 24 of Studies in Mathematics, Mathematical Association of America, 1984

[12] Lanczos C., Solution of systems of linear equations by minimized iterations. Journal of Research of the National Bureau of Standars, 1952 49:33–53.

[13] Langenbuch, S., Maurer, W., Werner, W., Coarse-mesh flux-expansion method for the analysis of space-time effects in large light water reactor cores. Nucl. Sci. and Eng., 1977, 63:437–456.

[14] McCormick, S.F., Multigrid Methods. Volume 3 of the SIAM Frontiers Series, SIAM, Philadelphia, 1987.

[15] Pershagen B., Light Water Reactor Safety. Pergamon Press, Oxford, 1989.

[16] Russell, R.D. A comparison of collocation and finite differences for two point boundary problems. SIAM Journal on Numerical Analysis, 1977, 14(1):19–39.

[17] Saad Y., Iterative methods for sparse linear systems. The PWS Series in Computer Science, PWS, Boston, 1996.

[18] Song J-W., Kim J-K., An efficient nodal method for transient calculations in light water reactors. Nuclear Technology, 1993, 103:157–167.

[19] Sonneveld P., CGS, a fast Lanczos-type solver for nonsymmetric linear systems. SIAM J. Sci. and Stat. Comp., 1989, 10(1):36–52.

[20] Van der Vorst, H.A., A Fast and Smoothly Converging Variant of Bi-CG for the solution of non-symmetric linear systems. SIAM J. Sci. Comp., 1992, 13:631-644.

[21] Verdú, G., *et al.,* 3D $\lambda-$modes of the neutron diffusion equation. Ann. Nucl. Energy, 1994, 21(7):405-421.

[22] Verdú, *et al.,* A consistent multidimensional nodal method for transient calculations. Ann. Nucl. Energy, 1995, 22(6):395-411.

[23] Vidal, V., et al., Optimization of the Subspace Iteration Method for the Lambda Modes Determination of a Nuclear Power Reactor. J. Nuclear Science and Technology, 1997, 34(9):939-947.

[24] Weston, J.R., Stacey, M., Space-Time Nuclear Reactor Theory. Academic Press, New York and London, 1970.