

Abstract

KASTANYA, DODDY FEBRIAN. Implementation of a Newton-Krylov Iterative Method to Address Strong Non-Linear Feedback Effects in FORMOSA-B BWR Core Simulator. (Under the direction of Paul J. Turinsky.)

A Newton-BICGSTAB solver has been developed to reduce the CPU execution time of the FORMOSA-B boiling water reactor (BWR) core simulator. The new solver treats the strong non-linearities in the problem explicitly using the Newton's method, replacing the traditionally used nested iterative approach. The Newton's method provides the solver with a higher-than-linear convergence rate, assuming that a good initial estimate of the unknowns is provided. Within each Newton iteration, an appropriately preconditioned BICGSTAB method is utilized for solving the linearized system of equations. Taking advantage of the higher convergence rate provided by the Newton's method and utilizing an efficient preconditioned BICGSTAB solver, we have developed a computationally efficient Newton-BICGSTAB solver to evaluate the three-dimensional, two-group neutron diffusion equations coupled with a two-phase flow model within a BWR core simulator.

The robustness of the solver has been tested against numerous BWR core configurations and consistent results have been observed each time. The best exact Newton-BICGSTAB solver performance, observed when performing calculations on 200 loading patterns (LPs) using an 800 assembly BWR/6 core model, provides an overall speedup of 2.07 to the core simulator, with reference to the traditional approach, *i.e.* outer (fission-source)-inner (red/black line SOR). When solving the same problem using the

traditional approach but with the BICGSTAB solver as the inner iteration solver [traditional (BICGSTAB)], we observed a speedup of 1.85. This means that the Newton-BICGSTAB solver provides an additional 12% increase in the overall speedup over the traditional (BICGSTAB) solver. However, one needs to note that, on average, the exact Newton-BICGSTAB solver provides an overall speedup of around 1.70; whereas, on average, the traditional (BICGSTAB) provides an overall speedup of around 1.60.

An investigation on the feasibility of implementing an inexact Newton-BICGSTAB solver has also been initiated in this research. Results from this study indicate that further reduction in the execution time can likely be obtained through the implementation of an inexact Newton's method. In general, the inexact Newton-BICGSTAB solver can provide speedups of 1.73 to 2.10 with respect to the traditional solver. As a specific example, an inexact Newton-BICGSTAB solver, in which the Jacobian coefficient matrix is approximated and a hybrid Chord-Shamanskii scheme is used to fix the frequency of the Jacobian matrix updates, can provide 2.10 and 2.27 overall and solver portion speedups, respectively. For this particular problem where a single LP of an 800 assembly BWR/6 core model is examined, the speedups gained by utilizing the traditional (BICGSTAB) solver and by utilizing the exact Newton-BICGSTAB solver are 1.63 and 1.76, respectively. This means that the inexact Newton-BICGSTAB solver provides additional speedups of 29% and 19% to the traditional (BICGSTAB) solver and the exact Newton-BICGSTAB solver, respectively.

**Implementation of a Newton-Krylov Iterative Method to
Address Strong Non-Linear Feedback Effects in FORMOSA-B
BWR Core Simulator**

by

DODDY FEBRIAN KASTANYA

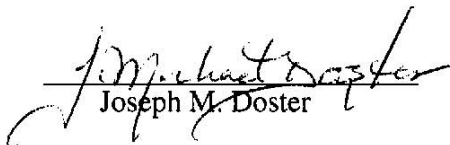
A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

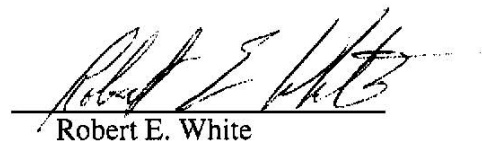
NUCLEAR ENGINEERING

Raleigh

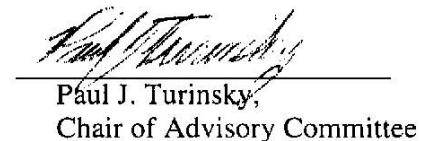
2002

APPROVED BY:


Joseph M. Doster


Robert E. White


Edward W. Davis


Paul J. Turinsky,
Chair of Advisory Committee

*“Tersembunyi ujung jalan
Hampir atau masih jauh
Ku dibimbing tangan Tuhan
Ke negri yang tak ku tau*

*Bapa ajar aku ikut
Apa juga maksudMu
Tak bersangsi atau takut
Beriman tetap teguh”*

*“For my thoughts are not your thoughts, nor are your ways my ways, says the LORD.
For as the heavens are higher than the earth, so are my ways higher than your ways
and my thoughts than your thoughts.”*

Isaiah 55:8-9

For Imelda and Bart.

Acknowledgment

The author would like to express his deepest and sincere appreciation to his advisor, Dr. Paul J. Turinsky, for his invaluable advice, guidance, and mentorship throughout the preparation of this dissertation. The author also wishes to express his gratitude for the financial support from the Department of Energy through the Nuclear Engineering Education Research grant and the Electric Power Research Center at North Carolina State University. The author also wishes to thank Dr. J. Michael Doster, Dr. Robert E. White, and Dr. Edward W. Davis, Jr. for serving as his advisory committee members.

A special thank goes out to his wife, Imelda Ariani, for her love, prayer, patience, and continuous support. The author would also like to thank his family for their support and prayers.

Table of Contents

	Page
List of Tables	vii
List of Figures	ix
1. Introduction.....	1
1.1. Background and Motivation	1
1.2. FORMOSA-B Code Overview	2
1.3. Literature Review	4
1.4. Purpose and Scope of Research	6
2. Methodology	9
2.1. Current Treatment in the FORMOSA-B code	9
2.1.1. Neutronics Model	9
2.1.2. The Thermal Hydraulics Model	10
2.1.3. Traditional Nested Iterative Technique	12
2.2. Timing Study of the Original FORMOSA-B Code	14
2.3. Implementation of Newton's Method	16
2.3.1. Mathematical Background	16
2.3.2. Non-linear Feedback Study	20
2.3.3. System of Equations	24
2.3.4. Structure of the Jacobian Matrix System and Equation Reduction	25
2.3.5. Linearization Error Study	31
2.4. Krylov Methods	35
2.5. The Basic Equations and Linearized System of Equations	36
2.5.1. The Diffusion Equation	36
2.5.2. The Two Phase Thermal Hydraulics Model	41
2.5.3. The Power Constraint	45
2.6. The Newton-Krylov Algorithm	46
2.7. Implementation of Inexact Newton's Methods	49
2.7.1. General Overview of Inexact Newton's Methods	49
2.7.2. The Chord Method	50
2.7.3. The Shamanskii's Method	51
3. Development of Preconditioners	53
3.1. A Block Incomplete LU Factorization	54
3.2. Porsching's Algorithm	56
3.3. Preconditioner 1	57
3.4. Preconditioner 2	59
3.5. Preconditioner 3	62

3.6. Performance of the Preconditioners	64
4. Implementation of BICGSTAB as the Inner Iteration Solver.	71
4.1. Introduction	71
4.2. General Overview of the Inner Iteration.	71
4.3. The Traditional R/B LSOR Solver	73
4.4. The BICGSTAB Solver	74
4.5. Results	76
5. Results from the Implementation of the Newton-BICGSTAB Solver.	83
5.1. Evaluation of a Single Loading Pattern	83
5.2. Evaluation of Multiple Loading Patterns	88
5.3. Results from a Control Rod Pattern Optimization Run	91
5.4. Results from the Inexact Newton's Method Studies	92
5.4.1. Chord and Shamanskii's Methods	93
5.4.2. A Hybrid Chord-Shamanskii Method	95
5.4.3. An Incomplete Jacobian Matrix Approach	98
6. Conclusions and Recommendations for Future Projects.	102
7. References	105
Appendix A: Derivation of the Jacobian matrix system for the EPRI void-quality correlation.	111

List of Tables

	Page
1. Introduction	
2. Methodology	
Table 2.1. Summary of Time Allocation Study.	15
Table 2.2. Cases Examined and Stopping Criteria.	22
Table 2.3. Relative Changes in the Number of Outer Iterations.	23
Table 2.4. Coupling of Variables.	26
3. Development of Preconditioners	
Table 3.1. Preconditioners Performance for a Simplified 368 Assembly BWR/4 Core Model.	65
Table 3.2. Detail Time Allocation (in seconds ¹).	66
Table 3.3. Preconditioner Performance (in seconds ¹) for a Realistic 368 Assembly GE BWR/4 Core Model.	69
4. Implementation of BICGSTAB as the Inner Iteration Solver.	
Table 4.1. R/B LSOR versus BICGSTAB Solvers.	75
Table 4.2. BICGSTAB Performance for Load Follow Calculations.	76
Table 4.3. BICGSTAB Performance for Critical Flow Search Calculations. ...	78
Table 4.4. FORMOSA-B LP-CRP Optimization Results for SDM Constraint for a 368 assembly GE BWR/4 core.	79
Table 4.5. FORMOSA-B CRP Sampling Results for an 800 Assembly BWR/6 Core.	80
5. Results from the Implementation of the Newton-BICGSTAB Solver.	
Table 5.1. The Newton-BICGSTAB Performance on a Single LP Using True Error Stopping Criteria [Utilizing an 800 Assembly BWR/6 Core].	85
Table 5.2. The Newton-BICGSTAB Performance on a Single LP [Utilizing an 800 Assembly BWR/6 Core].	86
Table 5.3. CRP Optimization Results Using an 800 Assembly BWR/6 Core. ..	92
Table 5.4. Inexact Newton's Method Results [Utilizing an 800 Assembly BWR/6 Core].	93
Table 5.5. Multiple LP Results from Testing the Shamanskii's Method [Utilizing an 800 Assembly BWR/6 Core].	94
Table 5.6. The Hybrid Chord-Shamanskii Method Performance [Utilizing an 800 Assembly BWR/6 Core].	96

Table 5.7. Results from Testing the Hybrid Chord-Shamanskii and Incomplete Jacobian Matrix Approach [Utilizing an 800 Assembly BWR/6 Core]. 99	
Table 5.8. Detail Time Allocation from Testing the Hybrid Chord-Shamanskii and Incomplete Jacobian Matrix Approach on a Single LP of 800 Assembly BWR/6 Core.	100
6. Conclusions and Recommendations for Future Projects.	
7. References	
Appendix A: Derivation of the Jacobian matrix system for the EPRI void-quality correlation.	
Table A.1. Variable Dependencies.	135

List of Figures

1. Introduction

2. Methodology

Figure 2.1: The traditional nested iterative algorithm.	13
Figure 2.2: Structure of $\overline{\overline{J_{22}}}$	28
Figure 2.3: Structure of $\overline{\overline{J_{22}^{iz, iz}}}$	29
Figure 2.4: Structure of $\overline{\overline{J_{22}^{iz, iz-1}}}$	29
Figure 2.5: Structure of $\overline{\overline{J_{22}}}$ matrix.	30
Figure 2.6: Structure of the reduced $\overline{\overline{J_{22}}}$ matrix.	30
Figure 2.7: Linearization error of the k_{eff} value.	33
Figure 2.8: Exact vs. approximate fast flux changes for +10% power perturbation.	34
Figure 2.9: Discretization notation for the flow channel.	42
Figure 2.10: The Newton-Krylov algorithm.	47
Figure 2.11: Exact Newton algorithm.	50
Figure 2.12: Chord method.	51
Figure 2.13: Shamanskii's method.	52

3. Development of Preconditioners

Figure 3.1: Porsching’s algorithm.	57
Figure 3.2: Structure of Preconditioner Type 1.	58
Figure 3.3: Preconditioner 1 algorithm.	59
Figure 3.4: Preconditioner 2 algorithm.	61
Figure 3.5: Preconditioner 3 algorithm.	63
Figure 3.6: Convergence Behavior of the Preconditioners.	68

4. Implementation of BICGSTAB as the Inner Iteration Solver.

Figure 4.1: Change in unknowns ordering.	74
Figure 4.2: Optimum CRPs obtained using R/B LSOR inner iteration solver for an 800 assembly BWR/6 core.	81
Figure 4.3: Optimum CRPs obtained using BICGSTAB inner iteration solver for an 800 assembly BWR/6 core.	82

5. Results from the Implementation of the Newton-BICGSTAB Solver.

Figure 5.1: Newton iterations versus stopping criteria.	87
Figure 5.2: Multiple LPs results - 800 F/A GE BWR/6 core.	89
Figure 5.3: Multiple LPs results - 368 F/A GE BWR/4 core.	90
Figure 5.4: Multiple LP results from testing the hybrid Chord-Shamanskii Method.	97

6. Conclusions and Recommendations for Future Projects.

7. References

1. Introduction

1.1. Background and Motivation

The arrangement of fuel assemblies and control materials in the core during the cycle of the nuclear power plant plays an important role in the in-core fuel management. There are substantial economic benefits that can be derived by optimizing the fuel assembly and control material loading patterns. At the same time, the desired optimum fuel assembly and control material loading patterns ought to meet safety and operational requirements.

Since a typical light water reactor has hundreds of fuel assemblies, evaluating every possible combinations of loading patterns becomes computationally prohibitive. For example, a typical pressurized water reactor (PWR) core which has 193 fuel assemblies will have more than 10^{50} possible loading patterns to evaluate. In a boiling water reactor (BWR), the type of reactor analyzed in this research, which can contain close to 800 fuel assemblies and where the positioning of control rods plays an important role in defining a loading pattern, the number of possible loading patterns increases significantly.

FORMOSA-B [40], which stands for **Fuel Optimization for Reloads Multiple Objectives by Simulated Annealing for BWR**, is an in-core nuclear fuel management optimization code which utilizes the simulated annealing (SA) stochastic optimization method to determine an optimum pairing of loading pattern (LP)- control rod pattern (CRP) in a BWR. In evaluating the attractiveness of each candidate of LP-CRP pairing during the course of an optimization, the three-dimensional, two-group neutron diffusion equation model in conjunction with a two-phase flow model, *e.g.* three-equation mixture drift flux model, must be solved as a function of cycle burnup. Consequently, the associated computational burden for this optimization becomes excessive. For example, a FORMOSA-B LP-CRP optimization during which approximately 4000 combinations of LP-CRP pairings are sampled typically takes about 27 CPU hours on a 2 GHz DELL 530 workstation platform.

To reduce the CPU time for optimization runs, a Newton type method has been implemented in the FORMOSA-B code. Through this implementation we try to reduce the number of iterations required to solve the coupled neutron diffusion and thermal hydraulics equations by performing simultaneous updates of the flux distribution, the thermal hydraulics properties, and the k_{eff} value. We believe that addressing the strong non-linearity in the problem explicitly using the Newton's method, aided by an appropriately preconditioned Krylov method for solving the linearized system of equations within the Newton iteration, should reduce the CPU execution time of an optimization run. The treatment for the remaining (weaker) feedback mechanisms will be performed through a nested iterative approach.

Traditionally, the nested iteration approach associated with treating the thermal hydraulic feedbacks (as well as other feedback mechanisms) has been utilized based upon the assumption that it is acceptable to delay the iterative update of the thermal hydraulics feedbacks *via* updating them at some frequency of the fission source iteration. A more in-depth discussion about the traditional nested iterative approach is given in Section 2. Previous studies on Newton type methods within a generalized perturbation theory (GPT) flux iterative formulation have shown that the Newton's method is very effective in treating the thermal hydraulics feedback effects with reference to the rate of convergence [41].

1.2. FORMOSA-B Code Overview

As briefly indicated earlier, FORMOSA-B is an in-core fuel management code for BWRs which utilizes the simulated annealing (SA) stochastic optimization method to find an optimum pairing of LP-CRP. In order to facilitate a proper cooling down schedule for the SA algorithm, several thousands of LP-CRP pairings are evaluated in each optimization run. Several objectives are available for the optimization in FORMOSA-B such as maximization of the end-of-cycle (EOC) core reactivity, minimization of the core limiting maximum fraction of linear power density (MFLPD) over the cycle, minimization

of the core maximum fraction of limiting critical power ratio (MFLCPR), maximization of the region average discharge burnup, and minimization of the total reload cost. In addition to the variety of optimization objectives, FORMOSA-B optimization runs are also actively controlled by a set of constraints. This set of constraints, which represents the operating and safety limits, will guarantee that the optimum pairing of LP-CRP obtained at the end of an optimization, provided that enough pairings have been evaluated and a feasible solution exists, satisfies the objective of the optimization as well as all of the constraints.

During an optimization run, it is usually desired to have at least five-to-ten thousand pairs of LP-CRP evaluated to guarantee a near-optimum solution for the problem as the result of the optimization. A family of several near optimum pairings of LP-CRP is also stored in the FORMOSA-B archives file. A core designer could then make his/her own choice from this pool of solutions.

Although the objective of an optimization makes it unique to other optimizations, all optimizations share a common thread. In each optimization, FORMOSA-B must evaluate each pairing of LP-CRP to assess its attractiveness. The evaluation involves solving the three-dimensional, two-group neutron diffusion equation which is coupled with a two-phase flow model. It should also be noted that while it is possible to reduce a PWR model into a two-dimensional model during an optimization run, the strong axial heterogeneity in a BWR and CRP decisions prevent such simplification.

The coupled neutronics/thermal hydraulics equations are highly non-linear. The non-linearity is currently treated using a nested iterative technique. More detailed informations about these nested iterations and the non-linearity of the problem are discussed in Section 2.

FORMOSA-B can employ either the coarse mesh finite difference (CMFD) method or the nodal expansion method (NEM) for solving the discretized three-dimensional problem. Utilizing the NEM as part of the solution technique adds yet another degree of non-linearity to the problem as implemented in the FORMOSA-B code. The evaluation of the NEM corrected spatial coupling coefficients is currently performed as the outer most loop in the nested iteration technique introduced above.

1.3. Literature Review

There are abundant sources of information on works related to in-core fuel management methods for both PWRs and BWRs. A brief review of the historical approaches to the PWR fuel management outlined in Kropaczek and Turinsky [31] or in Kropaczek's dissertation [32] provides a complete discussion in this area. Moore [41] provides equally comprehensive discussions for the historical development of the BWR fuel management area in his dissertation. Papers by DeChaine/Feltus [12, 13], Kiguchi [29], Lin [36], Maldonado/Turinsky [37], Poon/Parks [42], Karve/Turinsky [24], and Keller/Turinsky [26] are some examples of relatively recent publications which provide detailed discussions about current practices in the in-core fuel management area.

As mentioned earlier, FORMOSA-B uses the SA method as its optimization engine. This optimization technique was originally developed based upon the natural phenomena of slow cooling of a solid. This idea was first introduced by Metropolis *et al.* [38] and further refined by Kirkpatrick *et al.* [30] as a tool for solving optimization problems. Plethora of sources are available for both the theoretical and application aspects of the SA method [20, 51, 54, 57]. This optimization technique was first incorporated into the FORMOSA-P code (an in-core fuel management code for PWRs) by Kropaczek [32] and then subsequently adopted by the FORMOSA-B code.

The FORMOSA-B code contains the NEM as an option to solve the discretized three-dimensional problem. Bennewitz [4], Finnemann [18], Wagner [55,56], and Koebke [55] are some of the pioneers of the NEM. Smith [46] has developed another variant of the nodal method called the analytic nodal method (ANM) which is characterized by an analytic solution to the transverse-integrated diffusion equation. Both NEM and ANM are widely adopted in nuclear reactor core simulator codes [1, 34, 47, 49].

Solution of the neutron diffusion equation, which is a part of the BWR core simulator, can be written as a solution of a generalized eigenvalue problem. Wachspress [58] introduced the fission source iteration approach, which is essentially the power method, to solve for the largest eigenvalue, *i.e.* the inverse of the effective multiplication

factor (k_{eff}), and the associated fundamental mode eigenvector, *i.e.* the neutron flux ($\bar{\phi}$), of the generalized eigenvalue problem. This approach is also known as the “outer-inner” iteration since one will need another “inner” iteration scheme, when working in two or three dimensional space, for performing the numerical solution of the eigenvector. In reality, this generalized eigenvalue problem is non-linear since the coefficient matrices are indirectly dependent upon the eigenvector, *i.e.* the neutron flux, through directly the flux or indirectly the power density. One way to treat the non-linearity is to update the coefficient matrix after several outer-inner iteration sequences and repeat the process until the eigenvalue, the eigenvector, and the coefficient matrix converge. This idea leads to the development of the nested iterative approach discussed in Section 2.1.3.

The open literature on the application of Newton’s method for solving the non-linear two-group neutron diffusion equation problem in a core simulator code is practically nonexistent. Other than the application of Newton’s type method within the framework of a GPT flux iterative formulation done by Moore [41], the discussion about the implementation of Newton’s method in a reactor simulator code as intended in this research is nonexistent. Therefore, the results of this research should provide significant contributions in this area to the field of nuclear engineering.

One of the challenges in the implementation of the Newton’s technique in this research is related to the potential of having a singular matrix at the solution. Since one of the key assumptions for the Newton’s method is that the Jacobian is not singular at the solution, we have to ascertain that the singularity of a sub-matrix block does not generate a singular Jacobian matrix. Fortunately, there have been several studies conducted on this specific issue. Decker and Kelley provide detailed theoretical discussions for treating problems related to the singularity of the Jacobian at convergence [14, 15]. In these papers both exact Newton’s method as well as one type of quasi-Newton’s method, namely the Broyden’s method, are analyzed.

In recent years, the Krylov subspace methods have begun to be utilized in the core simulator codes for reactor analysis. Implementation of the preconditioned bi-conjugate gradient stabilized (BICGSTAB) solver, with both block incomplete LU factorization and

domain decomposition preconditioners, into the NESTLE code [52] shows that the BICGSTAB solver significantly outperforms the line successive over relaxation (LSOR) method [22, 23] for completing the inner iterations, the method currently used in FORMOSA-B. Some other recent publications on this subject are available in [17], [21], [48], and [58].

One of the Krylov subspace methods, *e.g.* the conjugate gradient squared (CGS), the BICGSTAB, or the restarted generalized minimal residual (GMRES), will be selected to be the solver of the linearized system of equations inside each Newton iteration in the FORMOSA-B code. Discussions about initial developments and some applications of these Krylov subspace methods can be found in the literatures [5] through [9], [11], [19], [45], [53]. In addition to these sources, Barrett *et al.* provides a compact but thorough discussion on the Krylov methods and also gives some introduction to some families of preconditioners in [3].

1.4. Purpose and Scope of Research

The objective of this research is to develop a robust and computationally efficient Newton-Krylov algorithm to solve the three-dimensional, two-group neutron diffusion equations coupled with a two-phase flow model. This new algorithm has been implemented in the large-scale in-core fuel management optimization code FORMOSA-B replacing the currently utilized traditional nested iterative algorithm. The new algorithm should provide the FORMOSA-B code with the much needed speedup required to make its usage practical in the design process.

We have formulated and implemented a new algorithm in which the Newton's method is utilized to explicitly treat the non-linearities present in the problem. The associated linearized system of equations represented by the Jacobian matrix system within each Newton iteration is solved utilizing one of the Krylov methods.

During this research, we have studied the effect of the stopping criteria used for terminating the Newton as well as the Krylov iterations on the overall convergence rate

and robustness of the solver. We specify these sets of stopping criteria such that the overall running time is minimized and more importantly that the solver is robust. In addition, the possibility of implementing an inexact Newton's method in order to provide an additional speed up to the code has been initiated.

The last important stage in completing this work is the formulation and implementation of an appropriate preconditioner to accompany the Krylov solver in order to minimize CPU time per Newton iteration. In developing the preconditioner, the block structure of the Jacobian matrix system is fully exploited. Moreover, the preconditioner also takes advantage of the underlying physics behind the observed structure.

The theoretical aspects of the development of the Newton-Krylov algorithm are discussed in Section 2. This section summarily introduces the current treatment for the coupled non-linear neutronics/thermal hydraulics model in the FORMOSA-B code. Also discussed in this section are the implementations of exact and inexact Newton's methods, the derivation of the Jacobian matrix system, and the implementation of a Krylov subspace solver. This section introduces the Newton-Krylov algorithm.

Section 3 reports the efforts in formulating an appropriate preconditioner for the Krylov solver. Three types of preconditioners are evaluated representing the evolution of preconditioners during this research. Two fundamental procedures common to these preconditioners are also succinctly discussed. Finally, the performance of these preconditioners are judged and the best one is selected to be implemented in the final version of the Newton-Krylov solver.

Before evaluating the performance of the Newton-Krylov solver, in Section 4 an alternative way of providing speedup to the traditional solver is introduced. Implementing a preconditioned BICGSTAB solver to replace the currently used red/black line SOR solver as the inner iteration solver proves to provide moderate speedups. Results from testing this solver on a single loading pattern as well as from optimization runs are reported in this section.

The results from implementing the Newton-Krylov solver are presented in Section 5. This section begins by selecting the most efficient Newton's method and Krylov solver. The comparisons between the performance of the Newton-Krylov and the

traditional solvers for both a single loading pattern case and several optimization runs are presented in this section.

Finally, Section 6 presents the conclusions regarding this work and provides some recommendations for future works.

2. Methodology

2.1. Current Treatment in the FORMOSA-B code

As indicated earlier, during a FORMOSA-B optimization run, we have to evaluate the three-dimensional, two-group neutron diffusion equations coupled with the three-equation mixture drift flux model for the two-phase flow. The first part is called the neutronics part and the latter part is called the thermal hydraulics part. The neutronics and thermal hydraulics parts are explained in some details in Section 2.1.1 and Section 2.1.2, respectively. In the subsequent sub-sections, the non-linearity in this system of equations will be introduced and the current method used for solving this problem is presented.

2.1.1. Neutronics Model

In order to evaluate the attractiveness of a certain LP-CRP pair, we have to solve the steady state three-dimensional, two-group neutron diffusion equations. These equations, assuming no upscatter term, can be written out for a node l as follows:

$$\begin{aligned} -\nabla D_1^{(l)} \nabla \phi_1^{(l)} + \Sigma_{R1}^{(l)} \phi_1^{(l)} &= \frac{1}{k_{eff}} (\nu \Sigma_{f1}^{(l)} \phi_1^{(l)} + \nu \Sigma_{f2}^{(l)} \phi_2^{(l)}) \\ -\nabla D_2^{(l)} \nabla \phi_2^{(l)} + \Sigma_{a2}^{(l)} \phi_2^{(l)} &= \Sigma_{s12}^{(l)} \phi_1^{(l)} \end{aligned} \quad , \quad (2-1)$$

where:

- $D_g^{(l)}$: diffusion coefficient for neutron energy group g ,
- $\phi_g^{(l)}$: flux for neutron energy group g ,
- $\Sigma_{R1}^{(l)}$: removal cross section for the fast neutron energy group,
- $\nu \Sigma_{fg}^{(l)}$: nu-sigma-fission cross section for neutron energy group g ,

- $\Sigma_{a2}^{(l)}$: absorption cross section for the thermal neutron energy group,
 $\Sigma_{s12}^{(l)}$: scattering cross section from the fast group to the thermal group,
 k_{eff} : effective multiplication factor.

Using NEM, we simplify this three-dimensional problem through transverse integration into three coupled, one-dimensional ordinary differential equations and a nodal balance equation. The utilization of a nodal methodology when implemented in a non-linear manner allows us to still employ the same matrix solver technique used in the coarse mesh finite difference (CMFD) method.

To understand the non-linearity embedded in Eq. (2-1), one needs to realize that the cross sections (Σ_x) are non-linearly dependent directly on the neutron flux and indirectly on the power density. The node-wise power density could be expressed as

$$q^{(l)} = \kappa \Sigma_{f1}^{(l)} \phi_1^{(l)} + \kappa \Sigma_{f2}^{(l)} \phi_2^{(l)}, \quad (2-2)$$

where κ denotes energy released per fission. The direct dependencies of cross sections on flux are due to transient fission products, *e.g.* Pm^{149} and Xe^{135} . The indirect dependencies of cross sections on power density are revealed through their dependencies on the moderator density and fuel temperature. The relationship between these thermal hydraulics variables to the core power will be explained in Section 2.1.2.

2.1.2. The Thermal Hydraulics Model

The main thrust of this sub-section is to briefly discuss some important aspects of the thermal hydraulics model utilized in the FORMOSA-B code to the extent that the discussion provides a clear picture on the relationship between the nodal power and both the moderator density and the fuel temperature. Discussions about the cross section dependencies on these thermal hydraulics variables are given in chapter 4 of [41].

The heart of the two-phase thermal hydraulics model in the FORMOSA-B code is the three-equation mixture drift flux model. There are several assumptions made in the implementation of this model. Firstly, it is assumed that the static pressure at the inlet or the outlet of the flow channels is uniform. Secondly, a subcooled or saturated condition is assumed at the inlet of a channel, *i.e.* a fuel bundle. Finally, a one-dimensional, upward flow is assumed. The last assumption proves to be pertinent in determining the structure of the linearized system of equations related to the implementation of the Newton's method. The full implementation of this model also requires some polynomial fits for the state equations for both subcooled and saturated conditions to provide the necessary closure of equations.

The node effective fuel temperature is determined *via* a polynomial fit of the node linear power density, $P(q^{(l)})$, with a correction term to account for fuel exposure, $\Delta T_{eff}(q^{(l)}, BU^{(l)})$, expressed as follows

$$T_{eff}^{(l)} = (T_{mod}^{(l)} - T_{mod}^{(ref)}) + P(q^{(l)}) + \Delta T_{eff}(q^{(l)}, BU^{(l)}), \quad (2-3)$$

where $T_{eff}^{(l)}$ denotes the node effective fuel temperature, $T_{mod}^{(l)}$ and $T_{mod}^{(ref)}$ denote the moderator temperature for node l and the reference moderator temperature, respectively, $q^{(l)}$ is the linear power density, and $BU^{(l)}$ is the burnup for node l .

While there is a direct expression relating the node relative power to the fuel temperature, the relation between the node relative power and the node moderator density is a little bit more arduous to explain. Instead of using complicated formulas to prove a point that we can indeed relate the node moderator density to the node relative power, the author chooses to construe the relation by describing the physical phenomena linking the two.

During a normal operation of a reactor, heat is actively generated inside the core. This heat will be transferred from the fuel assembly where it is generated to the coolant (moderator) where the heat is then deposited. Do note that the term coolant and moderator are used interchangeably since they are one and the same in light water reactors (LWRs).

The former expression comes about from the fact that water acts as the convective cooling media for the core, while the latter one is related to the fact that water moderates the high energy fission neutrons to lower energies. As heat is added, the water molecules will have higher energy and will be able to move more freely away from each other. Consequently, these activities reduce the number of water molecules present in a unit volume of water. The physical variable representing the total weight of the molecules per unit volume of a certain substance is called mass density or simply density. Therefore, it should be clear that the change in the power density distribution, which alters the amount of heat deposited in the coolant, will have a direct impact on the moderator density.

2.1.3. Traditional Nested Iterative Technique

The traditional method to treat the non-linearity in the implementation of the non-linear NEM is to employ a nested iterative algorithm (see Figure 2.1). In the outer most loop, the spatial coupling coefficients of the NEM are calculated. Inside this loop, the thermal hydraulics updates are performed. The next inner loop of this nested iterative algorithm is the source iteration, which consists of yet another level of iterations (outer-inner iterations). The effective multiplication factor and the fission source term are updated in the outer iteration. The algorithm will then solve the neutron flux distribution for both fast and thermal groups *via* inner iterations. Although we have a coupled system of equations between the two neutron energy groups, as clearly indicated in Eq. (2-1), we can solve first the fast group then followed by the thermal group since there is no upscattering. The Wielandt shift technique, where the eigenvalue of the matrix system is shifted slightly, is also fully utilized to accelerate the convergence rate of the source iteration [50]. Since the Wielandt shift technique introduces thermal to fast energy group coupling, Sutton's [49] method is employed to remove this coupling. The stopping criteria for this iterative approach is performed at the outer most level which provides higher assurance of the convergence of all nested iterations. Currently, the stopping criteria are

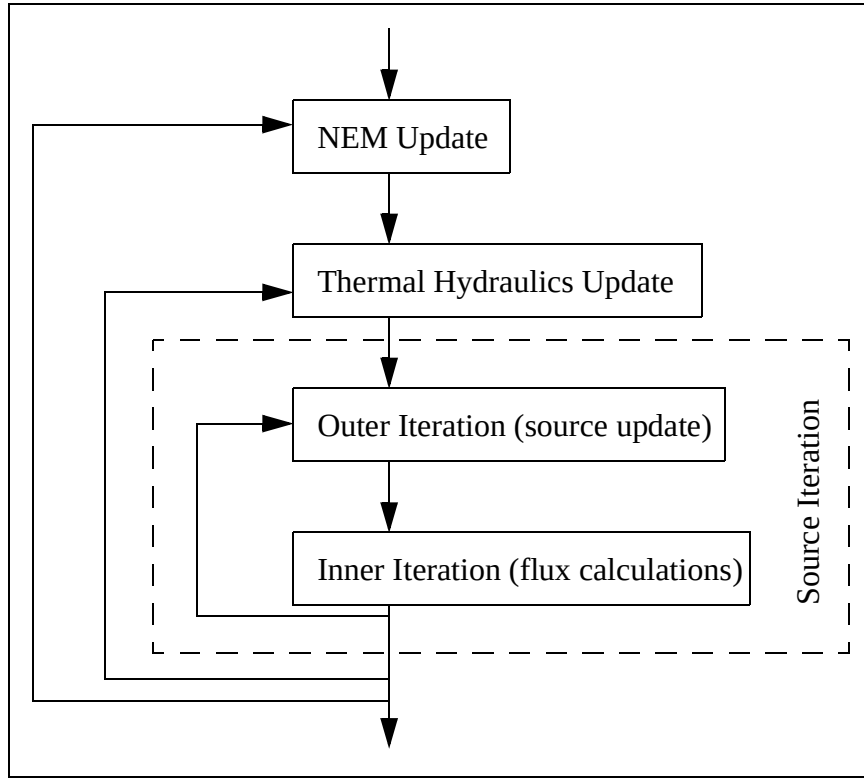


Figure 2.1: The traditional nested iterative algorithm.

imposed on the fission source, the effective multiplication factor, and the moderator density.

The coefficient matrix for each neutron energy group at the inner iteration level has a seven-banded structure. The system of equations at the inner iteration is solved using a red/black line successive over relaxation (R/B LSOR) solver with optimum relaxation parameter values, which are dynamically determined as the coefficient matrix changes due to non-linear feedback effects or criticality search adjustments.

After obtaining the flux distribution, we can calculate the power distribution inside the core which is needed to solve for the distributions of the thermal hydraulics properties of the core. The cross sections are then calculated based upon the updated thermal hydraulics properties. The effect of fission products build up is also accounted for when

performing the cross section calculation. In addition, we also have enough information to update the non-linear NEM coupling coefficients. Do note that the thermal hydraulics updates are performed at a certain frequency of the source iterations and the NEM update will be done after a certain number of thermal hydraulics updates has been performed. The frequencies of updates have been studied carefully to determine an optimum combination of update frequencies which will guarantee the lowest total FLOP count as well as the robustness of the algorithm.

2.2. Timing Study of the Original FORMOSA-B Code

The purpose of this study is to analyze the performance of the original version of FORMOSA-B code and to determine where most of the computational time is spent during an optimization run. There are several objectives which can be selected from for an optimization run in the FORMOSA-B code. In this study, we have examined four optimization objectives, namely the EOC k_{eff} maximization, the maximum fraction of linear power density (MFLPD) minimization, the maximum fraction of limiting critical power ratio (MFLCPR) minimization, and the region average discharge burnup maximization. For each of these objectives, we measure the time spent performing five major functions:

- T/H Calculations: all calculations related to the two-phase mixture drift-flux model.
- Cross Section Calculations: initial cross section setup, cross section updates at each burnup step, cross section updates due to T/H feedbacks, and fission product feedback corrections to the absorption cross section.
- NEM Updates: calculating transverse leakages, quadratic expansion coefficients, and solving the one-node/two-node problems.
- Coefficient Matrix Setup: setting up the coefficient matrix $\overline{\overline{M}}$ for solving the generalized eigenvalue problem $\overline{\overline{M}}\overline{\overline{\phi}} = \lambda\overline{\overline{F}}\overline{\overline{\phi}}$.
- Linear Equation Solver: completing the inner iterations using the R/B LSOR solver.

Table 2.1: Summary of Time Allocation Study.

FORMOSA-B Function	Optimization Objective			
	EOC k_{eff} maximization	MFLPD minimization	MFLCPR minimization	Region average discharge BU maximization
T/H Calculations	18.80%	18.84%	21.06%	20.02%
Cross Section Calculations	13.60%	16.41%	15.56%	14.52%
NEM Updates	12.71%	12.82%	14.45%	13.49%
Coefficient Matrix Setup	4.38%	4.47%	5.05%	4.84%
Linear Solver	43.04%	40.67%	36.48%	38.77%
Others	7.47%	6.79%	7.40%	8.36%

The results from this study are summarized in Table 2.1. Before analyzing the results, do note that each optimization run examines approximately 8,000 loading patterns with a fixed control rod pattern.

From Table 2.1, one can see that for all optimization objectives, approximately 40% and 90% of the total execution time is spent in the linear equation solver and in the overall solver, respectively. This implies that if time spent for the linear solver could be made close to zero, we could speedup the code by only a factor of around 1.6. To obtain higher speedup, we need to seek other sources of speedup which will significantly reduce not only the linear solver time but more importantly the overall solver time. Implementing the Newton's method, which is capable of updating several non-linear feedback properties simultaneously, is the path taken to provide the needed speedup.

2.3. Implementation of Newton's Method

This section is further divided into five sub-sections. In the first two sub-sections, a brief discussion on the mathematical background of the Newton's method and results from studies performed to determine strong non-linear feedback mechanisms are presented. A direct application of the Newton's method to the problem at hand is also concisely discussed. The next sub-section covers the system of equations involved in the model. The structure of the Jacobian matrix system is observed in the ensuing sub-section. This sub-section also discusses how the Jacobian matrix system is simplified by reducing the number of unknowns in the system. Finally, the last sub-section presents results from the linearization error study.

2.3.1. Mathematical Background

Consider a non-linear equation given by

$$\bar{A}(\bar{\Phi}) = \bar{Q}, \quad (2-4)$$

where the non-linear operator $\bar{A}$ operates on the unknown vector $\bar{\Phi}$ whose vector dependence indicates different reactor core properties. Eq. (2-4) includes appropriate boundary conditions and, if an eigenvalue problem is involved, normalization constraint. For the problem at hand, the unknown vector $\bar{\Phi}$ consists of all of the unknowns and can be subdivided into the eigenvalue (λ), where $\lambda = 1/k_{eff}$, the neutron group flux ($\bar{\phi}$), the strong non-linear feedback properties ($\bar{\xi}$), *i.e.* fuel temperature, coolant temperature, and coolant density, and the weak non-linear feedback properties ($\bar{\Psi}$), *i.e.* Xe¹³⁵ and Sm¹⁴⁹ number densities and nodal spatial coupling correction coefficients. The unknown vector can be mathematically written as

$$\bar{\Phi} = [\bar{\phi}^T, \bar{\xi}^T, \lambda, \bar{\Psi}^T]^T. \quad (2-5)$$

The subdivision of the non-linear feedback properties is performed in order to allow different iterative update frequency between the strong and the weak feedback properties. A study for determining which feedback mechanisms are considered strong has been completed and the results are presented in Section 2.3.2. Expressing Eq. (2-4) explicitly in terms of the eigenvalue, neutron flux and feedback properties, we obtain

$$\bar{A}(\bar{\Phi}) = \begin{bmatrix} (\bar{M}(\bar{\xi}, \bar{\Psi}) - \lambda \bar{F}(\bar{\xi}, \bar{\Psi}))\bar{\phi} \\ \bar{T}(\bar{\phi}, \bar{\xi}, \bar{\Psi}) \\ \bar{W}(\bar{\phi}, \bar{\xi}, \bar{\Psi}) \end{bmatrix} = \begin{bmatrix} \bar{0} \\ \bar{Q}_{\xi} \\ \bar{0} \end{bmatrix}. \quad (2-6)$$

The first row of Eq. (2-6) indicates the balance equation for the neutron flux, *i.e.* the spatially discretized form of the three-dimensional, two-group neutron diffusion equation.

The $\bar{M}$ and $\bar{F}$ denote the loss and production operators, respectively. Note that these operators are independent of flux. The second row of Eq. (2-6) represents some form of two-phase fluid mass, energy, and momentum conservation equations, such as the commonly employed drift-flux model, along with the constitutive equations which are used to determine the coolant density distribution in the core, plus the core power level normalization constraint. The third row states the balance equations for the weak non-linear feedback properties.

The traditional nested iterative algorithm for solving Eq. (2-6), as visualized in Figure 2.1, can be mathematically written as

$$\bar{M}(\bar{\xi}_n, \bar{\Psi}_o)\bar{\phi}_{m+1} - \lambda_m \bar{F}(\bar{\xi}_n, \bar{\Psi}_o)\bar{\phi}_m = \bar{0}, \text{ for } m = 0, 1, 2, \dots, \quad (2-7)$$

$$\bar{T}(\bar{\phi}_m, \bar{\xi}_{n+1}, \bar{\Psi}_o) = \bar{Q}_{\xi}, \text{ for } n = 0, 1, 2, \dots, \quad (2-8)$$

and

$$\bar{W}(\bar{\phi}_m, \bar{\xi}_n, \bar{\Psi}_{o+1}) = \bar{0}, \text{ for } o = 0, 1, 2, \dots, \quad (2-9)$$

where m , n , and o denote the fission source, strong non-linear feedback properties, and weak non-linear feedback properties iterative counts, respectively. The indices with tildes

on top of or underneath them indicate the frequency of updates; for example, several fission source iterations could be performed prior to updating the strong non-linear feedback properties.

Ideally, it is desired to update the flux and the strong non-linear feedback properties simultaneously. This can be done through a Newton type method [27, 28]. The implementation of Newton's method in the FORMOSA-B code will be focused only on treating the neutron flux and strong non-linear feedbacks properties. The remaining (weak) non-linear feedbacks will be updated in an iterative fashion as an additional loop outside the Newton iteration loop.

The non-linear system of equations we are interested in solving by the Newton's method can be written as

$$\bar{B}(\bar{\Theta}) \equiv \begin{bmatrix} (\bar{M}(\bar{\xi}, \bar{\Psi}) - \lambda \bar{F}(\bar{\xi}, \bar{\Psi}))\bar{\phi} \\ \bar{T}(\bar{\phi}, \bar{\xi}, \bar{\Psi}) \\ \langle \bar{\kappa} \bar{\Sigma}_f, \bar{\phi} \rangle \end{bmatrix} = \bar{Q} \equiv \begin{bmatrix} \bar{0} \\ \bar{Q}_\xi \\ \bar{Q}_P \end{bmatrix}. \quad (2-10)$$

The core power level constraint equation has been separated out of the $\bar{T}$ operator in Eq. (2-10), with the $\bar{T}$ operator denoting the strong feedback properties. As noted earlier, since an eigenvalue equation is being solved, a constraint equation is required, in this case to normalize the flux. The constraint equation allows the eigenvalue to be introduced as an additional unknown. The exact Newton equation for non-linear equation, Eq. (2-10), suppressing the dependence upon the weak non-linear feedbacks, $\bar{\Psi}$, which are updated in an outer iterative loop, can be expressed as

$$\bar{J}(\bar{\Theta}_m)\bar{\Theta}_{m+1} = \bar{R}(\bar{\Theta}_m), \quad (2-11)$$

where the Jacobian matrix has been defined as

$$J(\bar{\Theta}_m) \equiv \left. \frac{\partial}{\partial \bar{\Theta}} \bar{B}(\bar{\Theta}) \right|_{\bar{\Theta}_m}, \quad (2-12)$$

$$\bar{\Theta}_m \equiv [\bar{\phi}_m^T, \bar{\xi}_m^T, \lambda_m]^T, \quad (2-13)$$

and

$$\bar{R}(\bar{\Theta}_m) \equiv \bar{\bar{J}}(\bar{\Theta}_m) \bar{\Theta}_m - (\bar{B}(\bar{\Theta}_m) - \bar{Q}), \quad (2-14)$$

and index m denotes the Newton iteration count. For our problem, the Jacobian matrix can be written as

$$\bar{\bar{J}}(\bar{\Theta}_m) = \left[\begin{array}{ccc} [\bar{M}(\bar{\xi}) - \lambda \bar{F}(\bar{\xi})] \left[\left(\frac{\partial}{\partial \bar{\xi}} \bar{M}(\bar{\xi}) - \lambda \frac{\partial}{\partial \bar{\xi}} \bar{F}(\bar{\xi}) \right) \bar{\phi} \right] - \bar{F}(\bar{\xi}) \bar{\phi} \\ \frac{\partial}{\partial \bar{\phi}} \bar{T}(\bar{\phi}, \bar{\xi}) & \frac{\partial}{\partial \bar{\xi}} \bar{T}(\bar{\phi}, \bar{\xi}) & 0 \\ \frac{\partial}{\partial \bar{\phi}} \langle \bar{\kappa} \bar{\Sigma}_f \bar{\phi} \rangle & \frac{\partial}{\partial \bar{\xi}} \langle \bar{\kappa} \bar{\Sigma}_f \bar{\phi} \rangle & 0 \end{array} \right] \bigg|_{\bar{\Theta}_m}. \quad (2-15)$$

Eq. (2-15) denotes a 3x3 block matrix whose block matrix components are denoted by $\bar{\bar{J}}(\bar{\Theta}_m)_{i,j}$ for $i, j = 1, 2, 3$. Using Eq. (2-11) and Eq. (2-15), one obtains the iterative algorithm (in similar fashion to Eqs. (2-7) through (2-9)) as follow

$$\begin{aligned} & \bar{\bar{J}}(\bar{\Theta}_m)_{1,1} \bar{\phi}_{m+1} + \bar{\bar{J}}(\bar{\Theta}_m)_{1,2} \bar{\xi}_{m+1} + \bar{\bar{J}}(\bar{\Theta}_m)_{1,3} \lambda_{m+1} = \\ & (\bar{\bar{J}}(\bar{\Theta}_m)_{1,1} \bar{\phi}_m + \bar{\bar{J}}(\bar{\Theta}_m)_{1,2} \bar{\xi}_m + \bar{\bar{J}}(\bar{\Theta}_m)_{1,3} \lambda_m) - \\ & [\bar{M}(\bar{\xi}_m) - \lambda_m \bar{F}(\bar{\xi}_m)] \bar{\phi}_m \end{aligned}, \quad (2-16)$$

$$\begin{aligned} & \bar{\bar{J}}(\bar{\Theta}_m)_{2,1} \bar{\phi}_{m+1} + \bar{\bar{J}}(\bar{\Theta}_m)_{2,2} \bar{\xi}_{m+1} = \\ & \bar{\bar{J}}(\bar{\Theta}_m)_{2,1} \bar{\phi}_m + \bar{\bar{J}}(\bar{\Theta}_m)_{2,2} \bar{\xi}_m - [\bar{T}(\bar{\phi}_m, \bar{\xi}_m) - \bar{Q}_\xi], \end{aligned} \quad (2-17)$$

and

$$\begin{aligned} & \bar{\bar{J}}(\bar{\Theta}_m)_{3,1} \bar{\phi}_{m+1} + \bar{\bar{J}}(\bar{\Theta}_m)_{3,2} \bar{\xi}_{m+1} = \\ & \bar{\bar{J}}(\bar{\Theta}_m)_{3,1} \bar{\phi}_m + \bar{\bar{J}}(\bar{\Theta}_m)_{3,2} \bar{\xi}_m - [\langle \bar{\kappa} \bar{\Sigma}_f \rangle_m \bar{\phi}_m - Q_P]. \end{aligned} \quad (2-18)$$

Reintroducing the weak non-linear feedback values, they are determined by solving for $\bar{\Psi}$ using

$$\bar{W}(\bar{\Theta}_m, \bar{\Psi}_{o+1}) = \bar{0}. \quad (2-19)$$

This set of equations is clearly more difficult to solve than the original matrix equations associated with the traditional nested iterative approach due to the simultaneous couplings of more unknowns. However, the Newton's method is still attractive since it converges to the solution at a higher-than-linear rate provided that a good initial guess of unknown values is used. Kelley provides copious discussion on the convergence behavior of the exact Newton's method in his book [27]. Discussion on the rate of convergence for the inexact Newton's method can be found in [16].

2.3.2. Non-linear Feedback Study

As indicated in the previous section, the non-linear feedbacks are categorized into strong and weak ones. Therefore, prior to implementing the Newton-Krylov solver into the FORMOSA-B code, we investigated the strengths of the non-linearities in the systems of equations solved. We would like to precisely identify the strong non-linear feedbacks for which a Newton-type treatment is more computationally advantageous than the currently utilized nested iteration algorithm.

To answer this question, we have performed a comprehensive study to determine which feedback mechanisms can be categorized as strong feedbacks. If the total number of outer (fission-source) iterations increases noticeably when a non-linear feedback effect is treated, then the feedback is categorized as a strong feedback. The non-linear feedbacks associated with the thermal hydraulics, coolant inlet flow redistribution, transient fission products, and the non-linear nodal iterative method spatial coupling correction coefficients have been examined. The behaviors of the non-linear feedbacks on four core types, namely a 368 assembly GE BWR/4 core, an 800 assembly GE BWR/6 core, a 724

assembly GE BWR/3 core, and a 560 assembly GE BWR/4 core, have been evaluated to verify the consistency of the results.

Table 2.2 presents the cases examined during this study and the stopping criteria utilized. The results of this study are summarized in Table 2.3. Included in this table are results from performing the study using two different sets of stopping criteria. The cases examined are chosen such that the individual effect of a certain feedback can be isolated (cases 1, 2, 3, and 5) as well as the aggregate effect of several feedbacks (cases 4 and 6 through 10). Note that the ratio is defined as

$$Ratio = \frac{n_{outer}^{Case}}{n_{outer}^{Ref}}, \quad (2-20)$$

where n_{outer}^{Case} denotes the number of outer iterations for a particular combination of non-linear feedbacks being tested and n_{outer}^{Ref} denotes the number of outer iterations without nodal, thermal hydraulics, coolant inlet flow redistribution, and critical flow search feedbacks treated.

Averaged over the four core types for Case 1, utilization of the NEM in place of the CMFD method increases the number of outer iterations on average by 12% and 19% for the loose and tighter stopping criteria, respectively. Due to these moderate changes in the number of iterations, this feedback mechanism is categorized as a weak feedback.

Case 2 shows the effect of thermal hydraulics feedback without flow redistribution, critical flow search, and spacer void model. Averaged over four core types, this feedback effect is considered a strong feedback mechanism since it increases the number of outer iteration by around 50%, regardless of the stopping criteria employed.

Comparing the ratio values for Cases 2 and 3, one can see that performing the inlet flow redistribution, such that the core has uniform inlet and outlet pressures for all flow channels, does not significantly change the number of outer iterations. Similar observation can also be made on Cases 6 and 7, where the nodal method is utilized for solving the three-dimensional, two-group neutron diffusion equation instead of the CMFD method. Therefore, the flow redistribution feedback is categorized as a weak feedback.

Table 2.2: Cases Examined and Stopping Criteria.

Case	NEM	TH	Flow Redistribution	Critical Flow	Spacer Void	Fission Product
1	On	Off	Off	Off	Off	On
2	Off	On	Off	Off	Off	
3	Off	On	On	Off	Off	
4	Off	On	On	On	Off	
5	Off	On	Off	Off	On	
6	On	On	Off	Off	Off	
7	On	On	On	Off	Off	
8	On	On	On	On	Off	
9	On	On	On	On	On	
10	On	On	On	On	Off	Off
Stopping Criteria						
$\frac{ k_{eff}^{(q)} - k_{eff}^{(q-1)} }{k_{eff}^q} \leq \epsilon_k; \frac{\ \rho^{(q)} - \rho^{(q-1)}\ _2}{\ \rho^{(q)}\ _2} \leq \epsilon_\rho; \frac{\ [\bar{F}\bar{\phi}]^{(q)} - [\bar{F}\bar{\phi}]^{(q-1)}\ _2}{\ [\bar{F}\bar{\phi}]^{(q)}\ _2} \leq \epsilon_F$						
Note: q and $q - 1$ are the two consecutive outer most loop iterations.						
				ϵ_F	ϵ_k	ϵ_ρ
Loose Stopping Criteria				5e-4	1e-4	5e-2
Tighter Stopping Criteria				1e-4	5e-5	2e-4

Table 2.3: Relative Changes in the Number of Outer Iterations.

Case	Core Type			
	368 assembly	560 assembly	724 assembly	800 assembly
Loose Stopping Criteria				
1	1.20	1.11	1.06	1.10
2	1.69	1.41	1.31	1.56
3	1.67	1.38	1.33	1.61
4	2.62	1.95	2.44	1.92
5	1.72	1.40	n/a	1.56
6	1.69	1.41	1.33	1.54
7	1.67	1.38	1.35	1.61
8	2.42	1.75	2.27	1.76
9	2.33	1.98	n/a	2.09
10	1.96	1.60	1.39	1.81
Tighter Stopping Criteria				
1	1.32	1.29	1.03	1.10
2	1.96	1.46	1.34	1.54
3	1.85	1.51	1.32	1.62
4	2.65	2.15	2.22	1.96
5	2.52	1.59	n/a	1.66
6	1.85	1.54	1.30	1.57
7	1.85	1.54	1.30	1.60
8	2.44	1.97	2.17	1.75
9	2.97	2.19	n/a	2.18
10	2.23	1.75	1.40	1.85

Observing the ratio values of Cases 3 and 4 as well as Cases 7 and 8, one can see that the critical flow search feedback mechanism can actually be categorized as a strong feedback. This feedback adds 9% to 83% increase in the number of outer iterations on top of the already strong thermal hydraulics feedbacks. However, in this research the critical flow search feedback will be treated in a nested iterative fashion with other weak feedbacks.

Although employing the spacer void model does not change the number of outer iteration considerably, it will still be treated using the Newton's method since the spacer void model directly affects the coolant density distribution which is a part of the thermal hydraulics model. Finally, comparing the ratio values of Cases 10 and 8, with the exception for the 724 assembly GE BWR/3 core, the fission product feedback causes moderate changes in the number of outer iteration and, hence, is categorized as a weak feedback.

Based on these results, we concluded that the thermal hydraulics feedbacks are considered strong feedbacks and, therefore, their associated unknowns will be treated using the Newton's method. The remaining weak feedbacks will be treated in a nested iterative fashion.

2.3.3. System of Equations

To fully understand the complexity in solving the coupled matrix equations (Eq. (2-16), Eq. (2-17), and Eq. (2-18)), one needs to examine the components of this matrix system. Extended discussion on the basic equations composing the three-equation mixture drift flux model used in the FORMOSA-B code can be found in Moore's dissertation [41] and will not be repeated here. Based on these equations, we can derive the Jacobian matrix system whose structure is discussed in the next sub-section.

2.3.4. Structure of the Jacobian Matrix System and Equation Reduction

Assuming that a three-equation, two-phase drift flux model is utilized, the strong non-linear feedbacks can be written as follow

$$\bar{\xi} = [\bar{\alpha}^T, \bar{Z}^T, \bar{\rho}^T, \bar{C}_0^T, \bar{V}_{gj}^T, \bar{x}_f^T, \bar{x}_{nvq}^T, \bar{x}_{eq}^T, \bar{u}^T, \bar{P}^T, \bar{G}^T]^T, \quad (2-21)$$

where: α is the void fraction,

$Z \equiv \sqrt{T_f}$, and T_f is the fuel temperature,

ρ is the moderator density,

C_0 and V_{gj} are mixture void-quality parameters,

x_f is the flow quality,

x_{nvq} is the quality at net vapor generation,

x_{eq} is the equilibrium quality,

u is the mixture internal energy,

P is the mixture pressure, and

G is the mixture mass flux.

Table 2.4 shows the coupling for the neutron flux and the strong non-linear feedback properties, where an X denotes a coupling in the governing equations. The relations shown represent the couplings when the Lellouche-Zolotar EPRI void-quality correlation is utilized. The couplings shown in this table represent all possible couplings between the variables regardless of the condition of the fluid. Reduced couplings exist for single-phase flow or two-phase flow where thermal equilibrium exists. One more important thing to note about this table is the fact that while the variables' interdependencies are nicely visualized, the additional complexities regarding the spatial coupling between nodes have not been revealed yet.

Table 2.4: Coupling of Variables.

Governing Equations	Unknowns											
	ϕ	α	Z	ρ	C_0	V_{gj}	x_f	x_{nvg}	x_{eq}	u	P	G
ϕ	X		X	X								
α		X			X	X	X				X	X
Z	X		X	X						X	X	
ρ		X		X						X	X	
C_0		X			X							
V_{gj}		X				X					X	
x_f							X	X	X			
x_{nvg}	X		X	X				X			X	X
x_{eq}									X	X	X	
u	X	X	X	X		X				X	X	X
P				X			X				X	X
G												X

Following the notations in Eqs. (2-16) through (2-18), we can symbolically write the linearized system of equations to be solved as:

$$\begin{bmatrix} \bar{J}_{11} & \bar{J}_{12} & \bar{J}_{13} \\ \bar{J}_{21} & \bar{J}_{22} & 0 \\ \bar{J}_{31} & \bar{J}_{32} & 0 \end{bmatrix} \begin{bmatrix} \delta\bar{\phi} \\ \delta\bar{\xi} \\ \delta\bar{\lambda} \end{bmatrix} = \begin{bmatrix} \bar{R}_1 \\ \bar{R}_2 \\ \bar{R}_3 \end{bmatrix}, \quad (2-22)$$

where $\delta\bar{\phi}$, $\delta\bar{\xi}$, and $\delta\bar{\lambda}$ denote the changes in successive Newton iterative values, *e.g.* $\delta\bar{\phi} \equiv \bar{\phi}_{m+1} - \bar{\phi}_m$. Before discussing the method for solving this system of equations, we

would like to observe the structure of the block sub-matrices. Since the Krylov technique will be utilized to solve this system of equations, the information about the structure of the block sub-matrices is crucial in developing the preconditioner to accompany the Krylov solver.

The $\overline{\overline{J}}_{11}$ sub-matrix represents the Jacobian of the discretized three-dimensional, two-group neutron diffusion equation and has a seven-block-banded structure. Note that unlike the treatment in the traditional solver, both energy groups are solved simultaneously producing the 2x2 block structure. The structure of $\overline{\overline{J}}_{11}$ sub-matrix suggests that a family of Block Incomplete LU (BILU) Decomposition could be utilized as part of the preconditioner.

The structure of the $\overline{\overline{J}}_{22}$ matrix will be examined next. First note that since BWRs employ closed flow channels and inlet flow redistribution is not being treated by the Newton's method, all flow channels are decoupled at the Newton iterative level. This implies that each flow channel can be treated separately. The general structure of $\overline{\overline{J}}_{22}$ for a flow channel can be seen on Figure 2.2, where $iz = 1$ denotes the bottom node and $iz = NZ$ denotes the top node.

Note that the sub-matrix $\overline{\overline{J}}_{22}^{iz, iz}$ represents the Jacobian of the thermal hydraulics equations at axial node iz with respect to the thermal hydraulics properties of axial node iz , and the sub-matrix $\overline{\overline{J}}_{22}^{iz, iz-1}$ represents the Jacobian of the thermal hydraulics equations at the axial node iz with respect to the thermal hydraulics properties of axial node $iz - 1$. Also note that in this analysis, the thermal hydraulics variables are arranged as follow:

$$(\overline{\xi})^{iz} = [\alpha^{iz} \ Z^{iz} \ \rho^{iz} \ C_0^{iz} \ V_{gj}^{iz} \ x_f^{iz} \ x_{nvg}^{iz} \ x_{eq}^{iz} \ u^{iz} \ P^{iz} \ G^{iz}]^T. \quad (2-23)$$

The structures of $\overline{\overline{J_{22}^{iz, iz}}}$ and $\overline{\overline{J_{22}^{iz, iz-1}}}$ are shown in Figures 2.3 and 2.4, respectively.

Using this information, we can visualize the structure of the overall $\overline{\overline{J_{22}}}$ matrix, which is shown in Figure 2.5 for an example where the z-line has five axial nodes. Note that in solving Eq. (2-22), the Jacobian matrix of the coupled thermal hydraulics equations, which consist of eleven variables, are algebraically reduced to four equations and four unknowns (*i.e.*, pressure, density, internal energy, and void fraction) per axial node. The structure of the reduced system of equations is shown in Figure 2.6, again assuming five axial nodes.

Since each block of the reduced matrix is a 4-by-4 matrix, the inverse of each diagonal block can be easily calculated. Therefore, using a forward sweep technique, the action of $\left\{ \overline{\overline{J_{22}}} \right\}^{-1}$ (in the reduced form) can be determined. The values for the other thermal hydraulics variables can then be calculated using a series of back substitutions.

$$\begin{bmatrix} \overline{\overline{J_{22}^{11}}} & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ \overline{\overline{J_{22}^{21}}} & \overline{\overline{J_{22}^{22}}} & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \overline{\overline{J_{22}^{32}}} & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \overline{\overline{J_{22}^{iz, iz-1}}} & \overline{\overline{J_{22}^{iz, iz}}} & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & \overline{\overline{J_{22}^{iz+1, iz}}} & \overline{\overline{J_{22}^{iz+1, iz+1}}} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & \overline{\overline{J_{22}^{NZ-1, NZ-1}}} & 0 \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & \overline{\overline{J_{22}^{NZ, NZ-1}}} & \overline{\overline{J_{22}^{NZ, NZ}}} \end{bmatrix}$$

Figure 2.2: Structure of $\overline{\overline{J_{22}}}$.

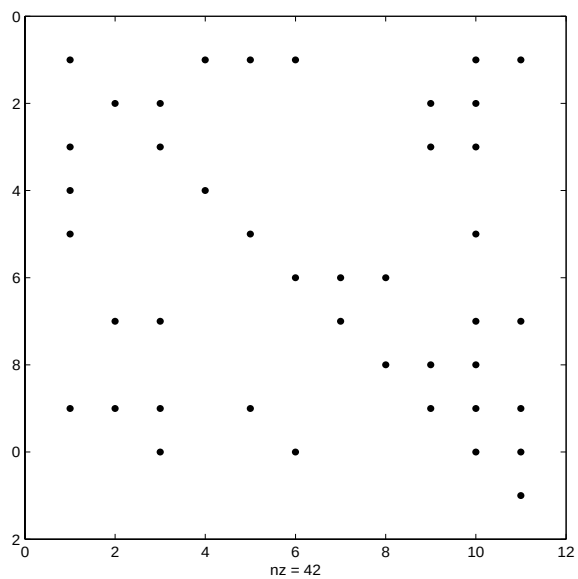


Figure 2.3: Structure of $\overline{\overline{J_{22}^{iz, iz}}}$.

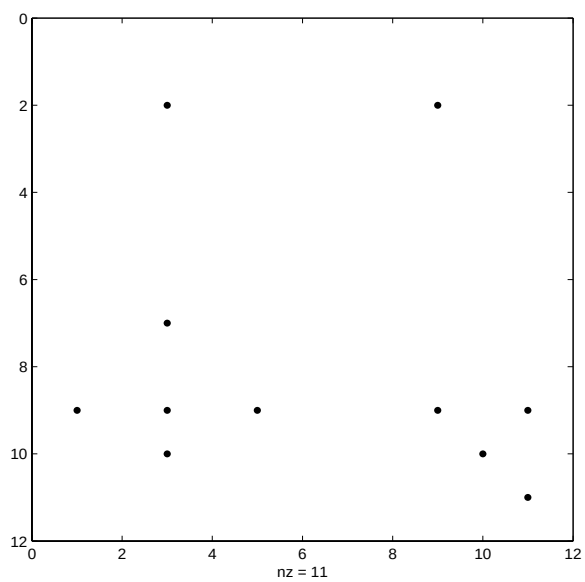


Figure 2.4: Structure of $\overline{\overline{J_{22}^{iz, iz-1}}}$.

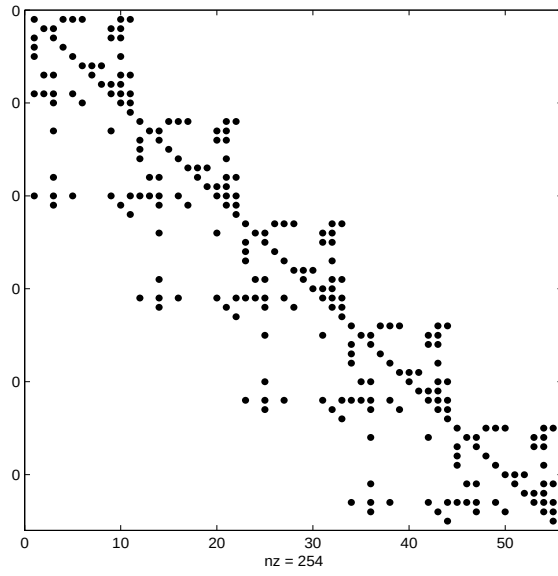


Figure 2.5: Structure of $\overline{\overline{J}}_{22}$ matrix.

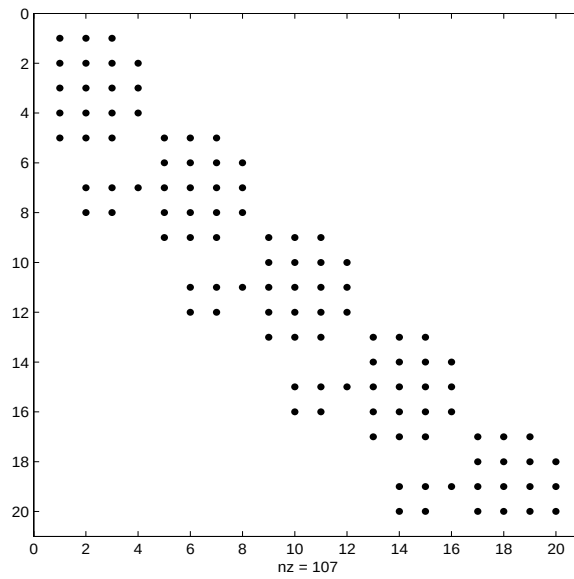


Figure 2.6: Structure of the reduced $\overline{\overline{J}}_{22}$ matrix.

2.3.5. Linearization Error Study

In order to assess the feasibility of employing the Newton's method for treating the strong non-linearities in the FORMOSA-B model, we would like to first understand how accurate the first order approximation is for our problem. A linearization error study has been performed to answer this question. In addition to observing how significant the errors introduced by using the first-order Newton's method are, this study was performed to also check the singularity property of the Jacobian matrix system at convergence and to serve as a tool to validate the derivation and implementation of the linearized system of equations.

The linearization error study was performed on a 368 assembly GE BWR/4 core. Since this study is focused on a single Newton step, the FORMOSA-B options for treating non-linear feedbacks not explicitly treated with the Newton's method, *e.g.* inlet coolant flow redistribution, critical flow search, and fission products, are turned off. This study begins by obtaining the solution at the 100% power level. Then, we performed linearization around this solution, *i.e.* evaluate the Jacobian matrix at 100% power level, and subsequently solve for the changes in the eigenvalue, two-group neutron flux, moderator pressure, density, void fraction, and internal energy distribution at a perturbed core power level, $100 \pm x\%$, *via* one Newton iteration. We also solve for these core variables at the perturbed power level using a traditional iterative solver. The difference between the exact changes, *i.e.* traditional solver, and the approximate changes, *i.e.* one Newton iteration, in these variables are defined as the linearization errors.

It is anticipated that the linearization errors will grow as the value of the core power level perturbation increases. However, since the two-group neutron flux and the thermal hydraulics properties have spatial dependencies, it is not trivial to succinctly quantify the degradation in the quality of the solutions as a result of applying a larger power perturbation. The error in the eigenvalue as a function of core power level perturbation size is selected to serve as the overall measure of linearization error. This is a compact manner to do the linearization error analysis since the eigenvalue is a unique

quantity for each power level and, hence, the accuracy in predicting this value can be easily plotted as a function of core power level.

The comparison between the approximate and the exact change in k_{eff} as well as the error of the predicted change in k_{eff} are presented in the top and bottom plots of Figure 2.7, respectively. From these graphs, one can see that utilizing one iteration of the first order Newton's method, the change in the eigenvalue can be quite accurately predicted. As expected, the error grows as the core being examined moves further away from the reference 100% core power level. However, one should keep in mind that during the course of Newton iterations, the fluctuations in core power will not likely come close to that caused by the full range of core power level perturbation analyzed in this study. This follows since the Jacobian matrix will be updated at each Newton iteration using the latest iterative values. Figure 2.8 shows the comparisons of the approximate and exact changes in the fast flux distribution for four radial locations (as shown in the core map below the plots) when the core power level is perturbed by +10%, *i.e.* 110% core power level. Although these plots represent an extreme perturbation in the core power, the first-order approximation still predicts the axial power distribution reasonably well. Note that slightly larger difference are observed in radial location "04"; however, recognizing the size of the relative power, one can conclude that these larger differences are not too important since the absolute differences are still relatively small. Based on these results, it is safe to assume that the errors introduced by the first-order approximation are insignificant.

The other thing learned from this study is that the Jacobian matrix is not singular. The singularity of the Jacobian matrix becomes a concern since the $\overline{\overline{J}}_{11}$ block, defined as $(\overline{\overline{M}}(\overline{\xi}, \overline{\Psi}) - \lambda \overline{\overline{F}}(\overline{\xi}, \overline{\Psi}))$, is singular when the value of λ is the true eigenvalue associated with the generalized eigenvalue problem. This is indeed the case for our linearization error study since we utilize the eigenvalue problem solution (at 100% core power level) as our reference. However, in reality additional terms are added to $\overline{\overline{J}}_{11}$ to account for cross-section changes due to fuel temperature changes. This follows since the cross-section is a

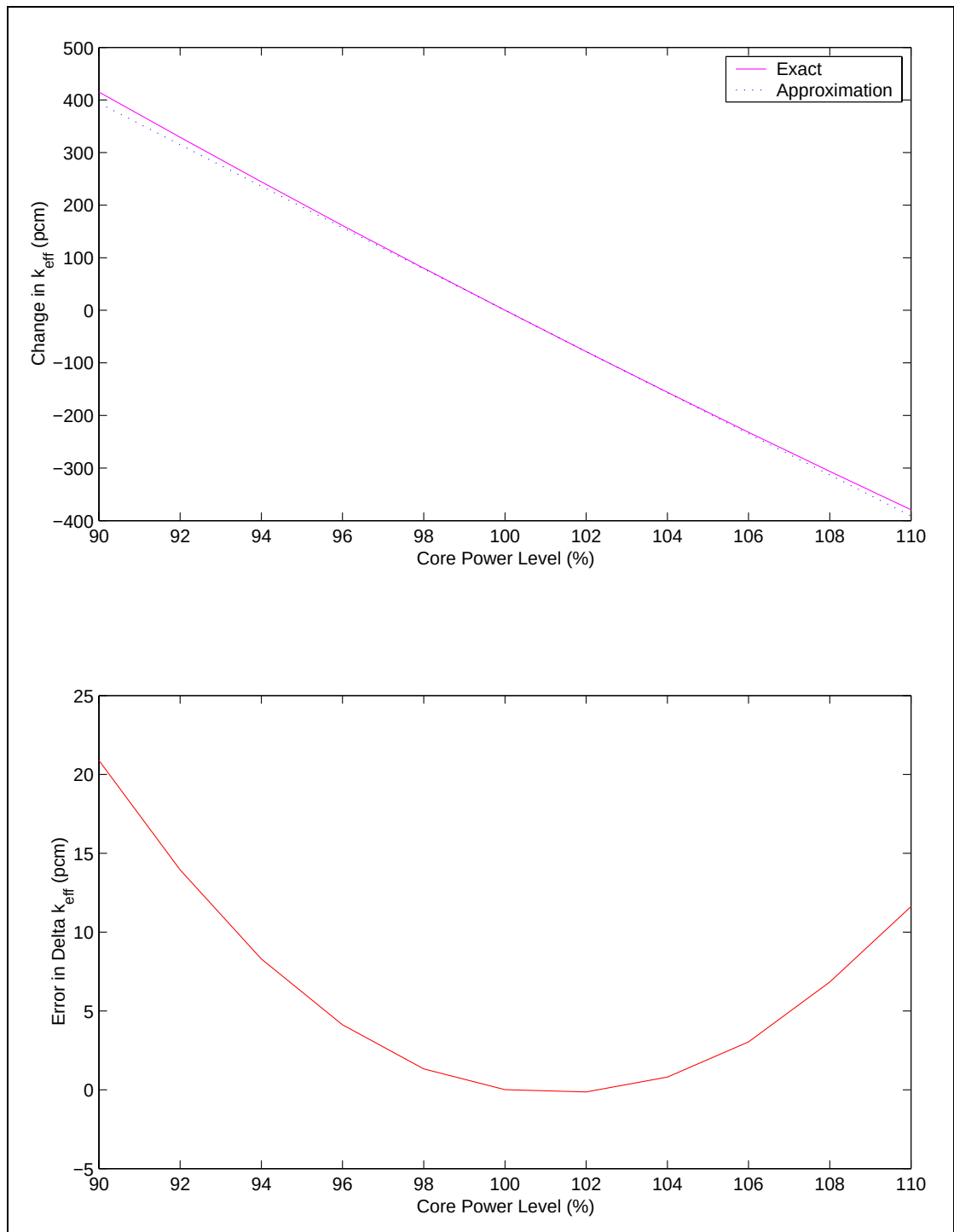


Figure 2.7: Linearization error of the k_{eff} value.

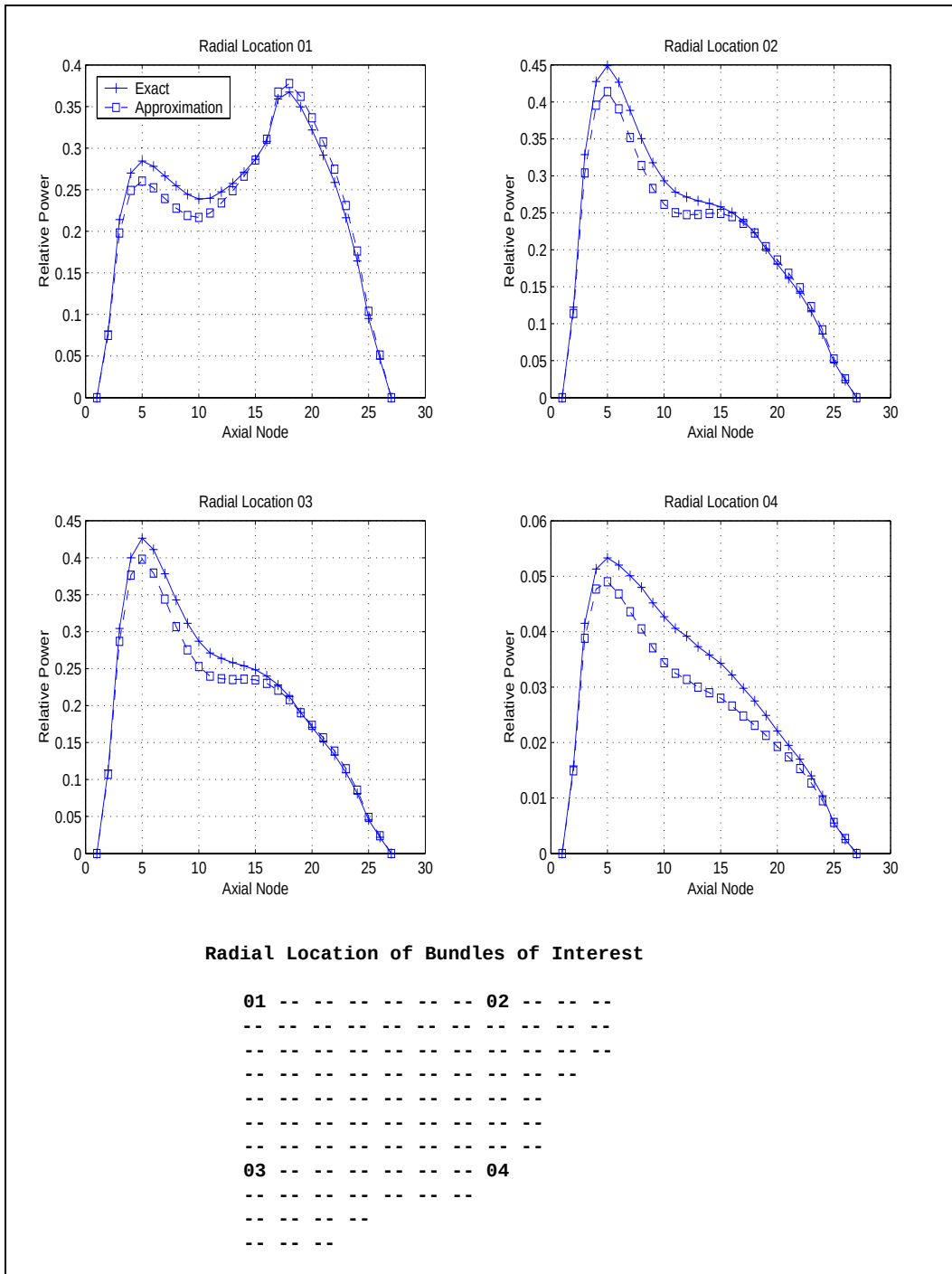


Figure 2.8: Exact vs. approximate fast flux changes for +10% power perturbation.

function of fuel temperature, which in turn is a function of power density, which in turn is a function of flux, and we reduce the cross-section dependence to flux. These additional terms when added to the $\bar{\bar{J}}_{11}$ block prevent this block from being singular.

2.4. Krylov Methods

Krylov methods are categorized as non-stationary iterative methods. For this family of methods, information needed to advance from one iteration to the next keeps changing. This means that when solving Eq. (2-11) for a fixed Newton iteration m , allowing us to suppress the m dependence and rewrite Eq. (2-11) as $\bar{\bar{J}}\bar{\Theta} = \bar{R}$, we can express the n^{th} Krylov iteration of $\bar{\Theta}$ as

$$\bar{\Theta}_{(n)} = \bar{\bar{G}}_n \bar{\Theta}_{(n-1)} + \bar{c}, \quad (2-24)$$

where $\bar{\bar{G}}_n$ is the iterative matrix and $\bar{c}$ is the remainder. When using unpreconditioned Krylov methods, the n^{th} iteration of $\bar{\Theta}$ is an element of

$$\bar{\Theta}_{(0)} + \text{span}\left\{\bar{r}_{(0)}, \bar{\bar{J}}\bar{r}_{(0)}, \dots, \bar{\bar{J}}^{n-1}\bar{r}_{(0)}\right\}, \quad (2-25)$$

which minimizes some norm measuring the distance between $\bar{\Theta}_{(n)}$ and the true solution, $\bar{\Theta}$. In Eq. (2-25), the initial residual ($\bar{r}_{(0)}$) is defined as

$$\bar{r}_{(0)} = \bar{R} - \bar{\bar{J}}\bar{\Theta}_{(0)}, \quad (2-26)$$

where $\bar{\Theta}_{(0)}$ represents the initial guess and $\text{span}\left\{\bar{r}_{(0)}, \bar{\bar{J}}\bar{r}_{(0)}, \dots, \bar{\bar{J}}^{n-1}\bar{r}_{(0)}\right\}$ is the n^{th} Krylov subspace. How the coefficients (multipliers) of the bases for a particular Krylov subspace are calculated is what makes one Krylov method unique from the others. A

discussion about the initial development of Krylov methods can be found in the literature [19, 45, 53]. Barrett *et. al.* [3] provides a compact but thorough discussion of Krylov methods and also gives an introduction to some families of preconditioner matrices.

For this research, we require Krylov methods which can solve a non-symmetric coefficient matrix since our Jacobian matrix is non-symmetric. We decided to examine three such Krylov methods, namely the bi-conjugate gradient stabilized (BICGSTAB), conjugate gradient squared (CGS), and restart generalized minimal residual (GMRES). The most computationally efficient solver will be employed in the final implementation of the Newton-Krylov solver.

2.5. The Basic Equations and Linearized System of Equations

In this subsection, an example of the derivation for the linearized system of equations is given. A simplified one-dimensional model is employed where a single fuel assembly is analyzed. Extensions to three-dimensional problem are straightforward. This subsection is further divided into several parts which introduce the basic equations. The linearized equations for the two-group neutron diffusion equation are provided as examples. Although the constitutive equations forming the two phase flow model and the equation for the power constraint are given in this section, the derivations for those equations are presented in the appendix.

2.5.1. The Diffusion Equation

The discretized two group neutron diffusion equation in 1-D using the finite difference method can be written as follow:

Fast Group

$$-\frac{D_1^k A}{\Delta z} \phi_1^{k-1} + \left[\frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k - \frac{D_1^k A}{\Delta z} \phi_1^{k+1} - \lambda v \Sigma_{f2}^k V \phi_2^k = 0$$

(2-27)

Thermal Group

$$-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} + \left[\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right] \phi_2^k - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} - \Sigma_{s12}^k V \phi_1^k = 0 \quad . \quad (2-28)$$

Before linearizing these equations, recall that all of the cross sections are a function of square root of fuel temperature and density: $D_g^k(\rho, Z)$ and $\Sigma_{xg}^k(\rho, Z)$. Also recall that fuel temperature T_{eff} is given by Eq.(2-3), which denotes it as a function of coolant temperature, which in turn is a function of internal energy, allowing us to utilize the reduced set of thermal hydraulics unknowns.

The linearized fast group equation can be written as

$$\begin{aligned} J_{\phi_1^k, \phi_1^{k-1}} \delta \phi_1^{k-1} + J_{\phi_1^k, \phi_1^k} \delta \phi_1^k + J_{\phi_1^k, \phi_1^{k+1}} \delta \phi_1^{k+1} + J_{\phi_1^k, \rho_p} \delta \rho_{k+1/2} + \\ J_{\phi_1^k, \rho_m} \delta \rho_{k-1/2} + J_{\phi_1^k, u_p} \delta u_{k+1/2} + J_{\phi_1^k, u_m} \delta u_{k-1/2} + \\ J_{\phi_1^k, \phi_2^k} \delta \phi_2^k + J_{\phi_1^k, \lambda} \delta \lambda = 0 \end{aligned} \quad , \quad (2-29)$$

where:

$$J_{\phi_1^k, \phi_1^{k-1}} = -\frac{D_1^k A}{\Delta z}$$

$$J_{\phi_1^k, \phi_1^{k+1}} = -\frac{D_1^k A}{\Delta z}$$

$$\begin{aligned}
J_{\phi_1^k, \phi_1^k} &= \frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V + \\
&\quad \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \phi_1} - \\
&\quad \left(\lambda \frac{\partial}{\partial Z} v \Sigma_{f2}^k \phi_2 + \frac{2A}{\Delta z} \left[\frac{\partial D_1^k}{\partial Z} \right] (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial \phi_1} \\
J_{\phi_1^k, \phi_2^k} &= -\lambda v \Sigma_{f2}^k + \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \phi_2} - \\
&\quad \left(\lambda \frac{\partial}{\partial Z} v \Sigma_{f2}^k \phi_2 + \frac{2A}{\Delta z} \left[\frac{\partial D_1^k}{\partial Z} \right] (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial \phi_2} \\
J_{\phi_1^k, \rho_p} &= \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \rho_{k+1/2}} + \\
&\quad \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left[\frac{2D_1^k A}{\Delta z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k - \\
&\quad \left(\left(\lambda \frac{\partial}{\partial Z} v \Sigma_{f2}^k \phi_2 + \frac{2A}{\Delta z} \frac{\partial D_1^k}{\partial Z} (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial \rho_{k+1/2}} \right) - \\
&\quad \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left(\lambda v \Sigma_{f2}^k \phi_2 + \frac{2A}{\Delta z} D_1^k (\phi_1^{k-1} + \phi_1^{k+1}) \right)
\end{aligned}$$

$$\begin{aligned}
J_{\phi_1^k, \rho_m} &= \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta Z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \rho_{k-1/2}} + \\
&\quad \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left[\frac{2D_1^k A}{\Delta Z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k - \\
&\quad \left(\left(\lambda \frac{\partial}{\partial Z} v \Sigma_{f2}^k \phi_2 + \frac{2A}{\Delta Z} \frac{\partial D_1^k}{\partial Z} (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial \rho_{k-1/2}} \right) - \\
&\quad \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left(\lambda v \Sigma_{f2}^k \phi_2^k + \frac{2A}{\Delta Z} D_1^k (\phi_1^{k-1} + \phi_1^{k+1}) \right) \\
J_{\phi_1^k, u_p} &= \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta Z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial u_{k+1/2}} - \\
&\quad \frac{\partial}{\partial Z} \left(\lambda v \Sigma_{f2}^k \phi_2^k + \frac{2A}{\Delta Z} D_1^k (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial u_{k+1/2}} \\
J_{\phi_1^k, u_m} &= \frac{\partial}{\partial Z} \left[\frac{2D_1^k A}{\Delta Z} + \Sigma_{R1}^k V - \lambda v \Sigma_{f1}^k V \right] \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial u_{k-1/2}} - \\
&\quad \frac{\partial}{\partial Z} \left(\lambda v \Sigma_{f2}^k \phi_2^k + \frac{2A}{\Delta Z} D_1^k (\phi_1^{k-1} + \phi_1^{k+1}) \right) \frac{\partial T \partial Z}{\partial Z \partial u_{k-1/2}} \\
J_{\phi_1^k, \lambda} &= - (v \Sigma_{f1}^k V \phi_1^k + v \Sigma_{f2}^k V \phi_2^k)
\end{aligned}$$

The linearized thermal group equation can be written as:

$$\begin{aligned}
&J_{\phi_2^k, \phi_2^{k-1}} \delta \phi_1^{k-1} + J_{\phi_2^k, \phi_2^k} \delta \phi_1^k + J_{\phi_2^k, \phi_2^{k+1}} \delta \phi_1^{k+1} + J_{\phi_2^k, \rho_p} \delta \rho_{k+1/2} + \\
&J_{\phi_2^k, \rho_m} \delta \rho_{k-1/2} + J_{\phi_2^k, u_p} \delta u_{k+1/2} + J_{\phi_2^k, u_m} \delta u_{k-1/2} + J_{\phi_2^k, \phi_1^k} \delta \phi_2^k = 0
\end{aligned} \tag{2-30}$$

where:

$$J_{\phi_2^k, \phi_2^{k-1}} = -\frac{D_2^k A}{\Delta z}$$

$$J_{\phi_2^k, \phi_2^{k+1}} = -\frac{D_2^k A}{\Delta z}$$

$$J_{\phi_2^k, \phi_2^k} = \frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V + \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] \frac{\partial T \partial Z}{\partial Z \partial \phi_2} -$$

$$\frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \phi_2} + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \frac{\partial T \partial Z}{\partial Z \partial \phi_2}$$

$$J_{\phi_2^k, \phi_1^k} = -\Sigma_{s12}^k V + \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] \frac{\partial T \partial Z}{\partial Z \partial \phi_1} -$$

$$\frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k \frac{\partial T \partial Z}{\partial Z \partial \phi_1} + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \frac{\partial T \partial Z}{\partial Z \partial \phi_1}$$

$$J_{\phi_2^k, u_p} = \left\{ \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] - \frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \right\} \times$$

$$\frac{\partial T \partial Z}{\partial Z \partial u_{k+1/2}}$$

$$J_{\phi_2^k, u_m} = \left\{ \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] - \frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \right\} \times$$

$$\frac{\partial T \partial Z}{\partial Z \partial u_{k-1/2}}$$

$$\begin{aligned}
J_{\phi_2^k, \rho_p} &= \left\{ \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] - \frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \right\} \frac{\partial T}{\partial Z} \times \\
&\quad \frac{\partial Z}{\partial \rho_{k+1/2}} + \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left\{ \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k - \frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} - \Sigma_{s12}^k V \phi_1^k \right\} \\
J_{\phi_2^k, \rho_m} &= \left\{ \frac{\partial}{\partial Z} \left[-\frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} \right] - \frac{\partial}{\partial Z} \Sigma_{s12}^k V \phi_1^k + \frac{\partial}{\partial Z} \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k \right\} \frac{\partial T}{\partial Z} \times \\
&\quad \frac{\partial Z}{\partial \rho_{k-1/2}} + \frac{1}{2} \frac{\partial}{\partial \rho_{avg}} \left\{ \left(\frac{2D_2^k A}{\Delta z} + \Sigma_{a2}^k V \right) \phi_2^k - \frac{D_2^k A}{\Delta z} \phi_2^{k-1} - \frac{D_2^k A}{\Delta z} \phi_2^{k+1} - \Sigma_{s12}^k V \phi_1^k \right\}
\end{aligned}$$

2.5.2. The Two Phase Thermal Hydraulics Model

The basic equations involved in the two-phase thermal hydraulics model are the three-equations mixture drift flux model. Assuming the steady state condition and using a staggered-mesh scheme for the quantities (see Figure 2.9), the discretization of the flow channel gives the following form of the mixture equations. Note that this set of equations forms a coupled non-linear system of equations; hence, the application of Newton's method to this problem is suitable.

Mixture Continuity

$$(GA_x)_{k+1/2} = (GA_x)_{k-1/2} \quad (2-31)$$

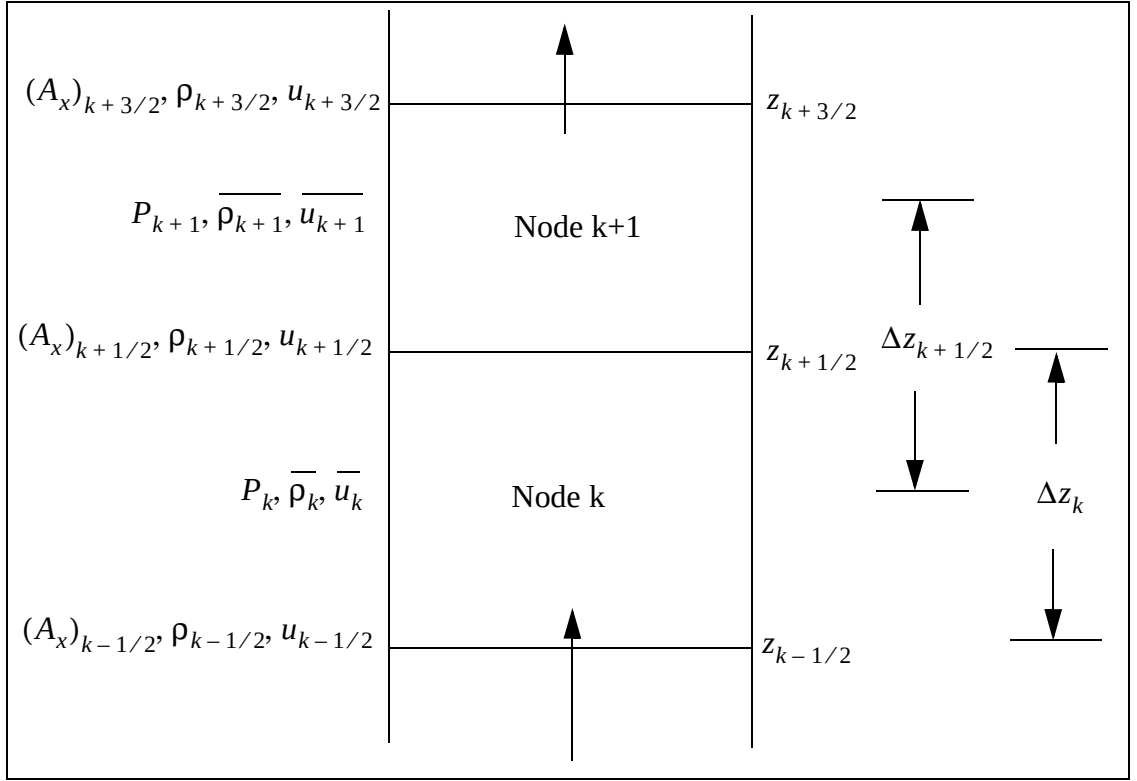


Figure 2.9: Discretization notation for the flow channel.

Mixture Momentum

$$\begin{aligned}
 P_{k+1} = P_k & - \frac{\Delta z_{k+1/2}}{\Delta z_k A_{x,k+1/2}} \left[\frac{(GA_x)_{k+1/2}^2}{(\rho A_x)_{k+1/2}} - \frac{(GA_x)_{k-1/2}^2}{(\rho A_x)_{k-1/2}} \right] - \\
 & \Delta z_{k+1/2} \frac{(f_{1\phi})_{k+1/2} (\phi_{fric}^2)_{k+1/2} (GA_x)_{k+1/2}^2}{2(D_e \rho_l)_{k+1/2} (A_x)_{k+1/2}^2} - \\
 & \frac{(K_{1\phi})_{k+1/2} (\phi_{local}^2)_{k+1/2} (GA_x)_{k+1/2}^2}{2(\rho_l)_{k+1/2} (A_x)_{k+1/2}^2} - \Delta z_{k+1/2} \rho_{k+1/2} g - \\
 & \frac{1}{(A_x)_{k+1/2}} \left(\left[\frac{\rho_l \rho_g}{\rho} \frac{\alpha V_{gj}^2}{(1-\alpha)} A_x \right]_{k+1/2} - \left[\frac{\rho_l \rho_g}{\rho} \frac{\alpha V_{gj}^2}{(1-\alpha)} A_x \right]_{k-1/2} \right)
 \end{aligned} \tag{2-32}$$

Mixture Internal Energy

$$\begin{aligned}
 u_{k+1/2} &= \left[\frac{(GA_x)_{k-1/2}}{(GA_x)_{k+1/2}} \right] u_{k-1/2} + \frac{q''' \bar{A}_x \Delta z_k}{(GA_x)_{k+1/2}} - \\
 &P_k \left[\frac{1}{\rho_{k+1/2}} - \left[\frac{(GA_x)_{k-1/2}}{(GA_x)_{k+1/2}} \right] \frac{1}{\rho_{k-1/2}} \right] - \\
 &\frac{P_k}{(GA_x)_{k+1/2}} \left(\left[\frac{\rho_l - \rho_g}{\rho} \alpha V_{gj} A_x \right]_{k+1/2} - \left[\frac{\rho_l - \rho_g}{\rho} \alpha V_{gj} A_x \right]_{k-1/2} \right) - \\
 &\frac{1}{(GA_x)_{k+1/2}} \left(\left[\frac{\rho_l \rho_g}{\rho} (u_g - u_l) \alpha V_{gj} A_x \right]_{k+1/2} - \left[\frac{\rho_l \rho_g}{\rho} (u_g - u_l) \alpha V_{gj} A_x \right]_{k-1/2} \right)
 \end{aligned} \tag{2-33}$$

State Equation

$$\rho_{k+1/2} = \rho(u_{k+1/2}, P_k) \tag{2-34}$$

In the two-phase region of the flow, the void-quality relationship accounting for the slip between the phases is given by the Zuber-Findlay [60] correlation:

$$\alpha = \frac{x_f}{C_0 \left(x_f + \frac{\rho_g}{\rho_l} (1 - x_f) \right) + \frac{\rho_g V_{gj}}{G}}, \tag{2-35}$$

where x_f is the flow quality. Another void-quality correlation examined during this research is the iterative EPRI model of Lellouche-Zolotar. The derivations of the Jacobian system of equations for the EPRI correlations are given in the Appendix section. When a thermal equilibrium exists between the liquid and vapor phases, the flow quality is simply the equilibrium quality (x_{eq}) which is determined by the following formula:

$$x_{eq} = \frac{h - h_f}{h_g - h_f}, \quad (2-36)$$

where h , h_f , and h_g are the node, the saturated liquid, and the saturated vapor enthalpies, respectively. However, prior to reaching the equilibrium state, some subcooled boiling region could exist in the channel. Within this region some bubble nucleations form on the surfaces of the fuel rods. The quantity of these bubbles increases as the flow moves up the channel and should be accounted for in determining the flow quality. The point where these bubbles begin to form is called the subcooled void departure or the point of net vapor generation. To predict the quality that exists at the point of net vapor generation (x_{nvg}), one can use the EPRI subcooled void formation model [35].

Having calculated the equilibrium and the net vapor generation qualities, one can then determine the flow quality by using the hyperbolic tangential fit introduced in Lahey and Moody [33]:

$$x_f = \frac{x_{eq} - x_{nvg} \left(1 - \tanh \left(1 - \frac{x_{eq}}{x_{nvg}} \right) \right)}{1 - x_{nvg} \left(1 - \tanh \left(1 - \frac{x_{eq}}{x_{nvg}} \right) \right)}. \quad (2-37)$$

Once an equilibrium state has been reached, the flow quality is set to the equilibrium quality and the calculation using the fit is bypassed.

The last pieces needed to complete the calculation of the void using Eq.(2-35) are the values of C_0 and V_{gj} . The simplest model to determine these quantities is that proposed by Zuber and Findlay [60]:

$$C_0 = 1.13 \text{ and } V_{gj} = 1.41 \left(\frac{(\rho_l - \rho_g)}{\rho_l^2} \sigma g \right)^{1/4}. \quad (2-38)$$

Other widely accepted models for predicting these quantities are the Chexal-Lellouche iterative method [10] and the EPRI (Lellouche-Zolotar) iterative model.

As part of the momentum equation, the single phase friction factor ($f_{1\phi}$) needs to be calculated. For a typical BWR application, the following formula is used:

$$f_{1\phi} = 0.186 \times Re^{-0.2} , \quad (2-39)$$

where Re is the Reynolds number. The two phase multiplier for the frictional loss (ϕ_{fric}^2) is determined using the Columbia correlation [43]:

$$\phi_{fric}^2 = 1 + x_f \left(\frac{\rho_l}{\rho_g} - 1 \right) C , \quad (2-40)$$

where:

$$C = \begin{cases} 1.02 \times x_f^{-0.175} \times G^{-0.45} & \text{if } P > 600 \text{ psi} \\ 0.357 \times \left(1 + 10 \frac{P}{P_c} \right) \times x_f^{-0.175} \times G^{-0.45} & \text{if } P \leq 600 \text{ psi} \end{cases} ,$$

and P_c is the critical pressure.

For the local loss, the two phase multiplier (ϕ_{local}^2) takes the following form:

$$\phi_{local}^2 = 1 + E \times x_f \left(\frac{\rho_l}{\rho_g} - 1 \right) , \quad (2-41)$$

where E is a constant.

2.5.3. The Power Constraint

The two-group neutron flux distribution obtained from solving the coupled neutronics/thermal hydraulics equations discussed above is normalized such that the required core power level is satisfied, *i.e.* the following relation must be satisfied:

$$\sum_{l=1}^L \sum_{g=1}^G \kappa \Sigma_{fg}^l \phi_g^l V^l = Q_P, \quad (2-42)$$

where $\kappa \Sigma_{fg}^l$ is the product of the amount of energy produced per fission event and fission cross-section in energy group g at node l , ϕ_g^l is the neutron flux for energy group g at node l , and V^l is the node volume of node l .

2.6. The Newton-Krylov Algorithm

The algorithm for a Newton-Krylov solver implemented in FORMOSA-B is depicted in Figure 2.10. The Newton iteration performs simultaneous updates of the eigenvalue λ ($\lambda = 1/k_{eff}$), two-group neutron flux (ϕ), and thermal hydraulics properties. Note that the fission source iterations have been completely eliminated. This is possible since the singular matrix system that appears in the eigenvalue problems does not appear in the Newton's method, *i.e.* the Jacobian matrix is non-singular. Although this figure is fairly self-explanatory, there are two items which should be pointed out. First, the weak non-linear feedback properties are being updated at a certain frequency of Newton (outer) iterations. Second, the box labeled “Setup the Jacobian matrix system and perform the LU factorization” is thicker than the other boxes to indicate that these processes are done at a certain update frequency depending upon whether an exact or inexact Newton's method is utilized.

Given that the Newton-Krylov algorithm is a nested iterative algorithm, a specific stopping criteria is needed for each level of the nested iterative loops. As the stopping criteria for the Krylov solver, we require the L_2 -norm of the reduction in the residual to be less than a specified value

$$\frac{\|\bar{r}_{(m,n)}\|_2}{\|\bar{r}_{(m,0)}\|_2} \leq \epsilon_{Krylov}, \quad (2-43)$$

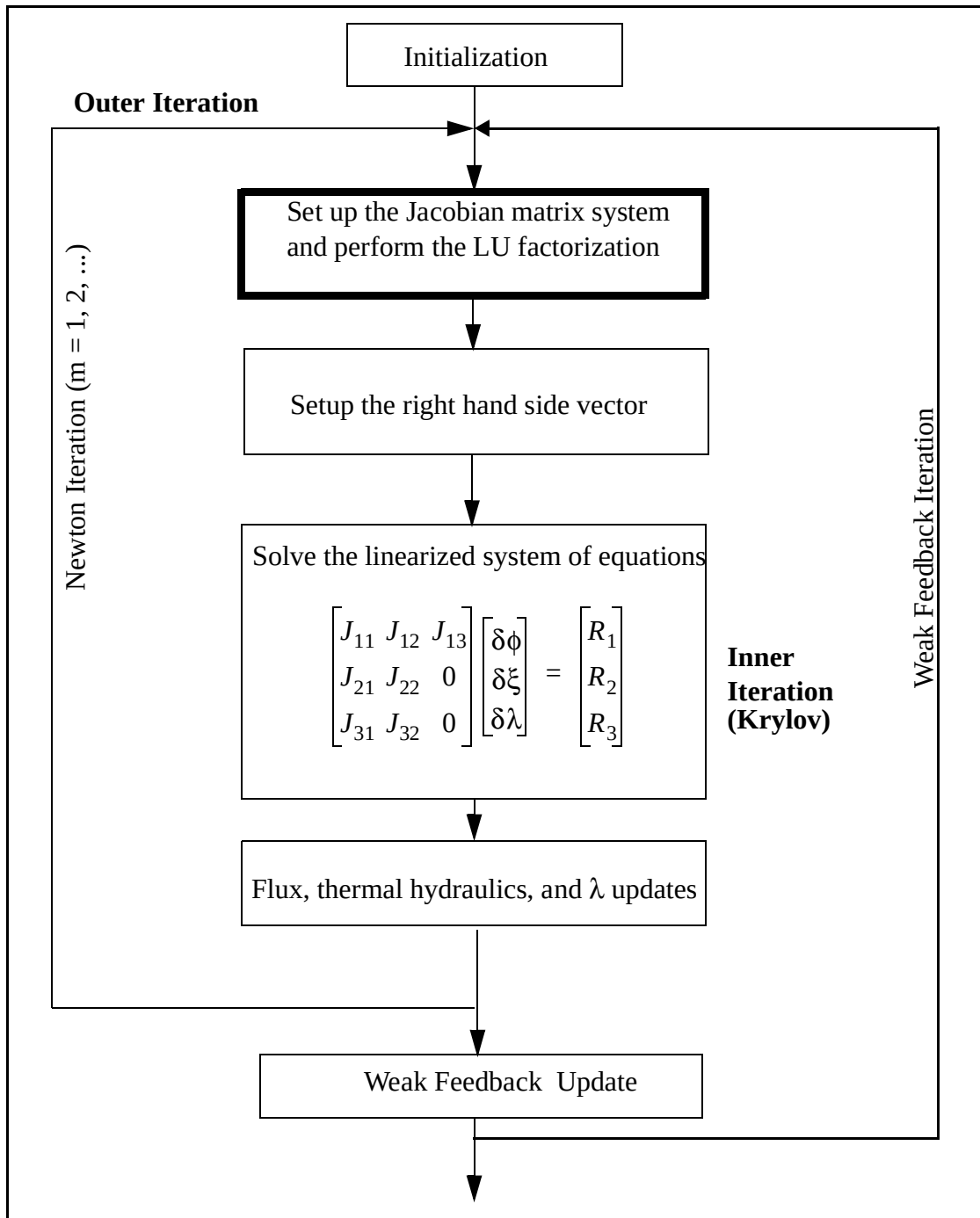


Figure 2.10: The Newton-Krylov algorithm.

where

$$\bar{r}_{(m,n)} = \begin{bmatrix} \bar{R}_{(m)_1} \\ \bar{R}_{(m)_2} \\ \bar{R}_{(m)_3} \end{bmatrix} - \begin{bmatrix} \bar{J}_{11} & \bar{J}_{12} & \bar{J}_{13} \\ \bar{J}_{21} & \bar{J}_{22} & 0 \\ \bar{J}_{31} & \bar{J}_{32} & 0 \end{bmatrix} \begin{bmatrix} \delta \bar{\phi}_{(m,n)} \\ \delta \bar{\xi}_{(m,n)} \\ \delta \bar{\lambda}_{(m,n)} \end{bmatrix}, \quad (2-44)$$

and the pair (m, n) denotes the (Newton, Krylov) iteration step counts. Note that when this Krylov iteration stopping criteria is satisfied, say at Krylov iteration $\hat{n}$, we denote $(m) = (m, \hat{n})$. For terminating the Newton iterations, the following criteria is used

$$\frac{\|\bar{M}(\bar{\xi}_m, \bar{\Psi}_o)\bar{\phi}_m - \lambda_m \bar{F}(\bar{\xi}_m, \bar{\Psi}_o)\bar{\phi}_m\|_2}{\|\lambda_m \bar{F}(\bar{\xi}_m, \bar{\Psi}_o)\bar{\phi}_m\|_2} \leq \epsilon_{Newton}. \quad (2-45)$$

We realize that this stopping criteria only checks the neutron diffusion equation solution while the Jacobian matrix system also includes thermal hydraulics properties. Hence, this stopping criteria was formulated by assuming that when the two-group fluxes and eigenvalue converge to sufficient accuracy, the thermal hydraulics properties have already converged to the accuracy desired. We have performed numerical tests to show that this is indeed a valid assumption. Another advantage of utilizing this stopping criteria is related to the fact that the term whose norm is taken and used as the numerator in Eq. (2-45) is readily available since it is part of the right hand side vector for the subsequent Newton iteration. Therefore, evaluation of the Newton iterations' stopping criteria does not add much to the computational cost.

Similar to the traditional solver, to terminate the overall solver, we check the convergence of the eigenvalue, fission neutron source density, and moderator density, in the outer most loop of the nested loops, *i.e.* the weak non-linear feedback loop. Mathematically, these stopping criteria can be written as

$$\frac{|k_{eff}^{(o)} - k_{eff}^{(o-1)}|}{k_{eff}^{(o)}} \leq \epsilon_k, \quad (2-46)$$

$$\frac{\|[\bar{\bar{F}}\bar{\phi}]^{(o)} - [\bar{\bar{F}}\bar{\phi}]^{(o-1)}\|_2}{\|[\bar{\bar{F}}\bar{\phi}]^{(o)}\|_2} \leq \epsilon_F, \quad (2-47)$$

and

$$\frac{\|\rho^{(o)} - \rho^{(o-1)}\|_2}{\|\rho^{(o)}\|_2} \leq \epsilon_\rho, \quad (2-48)$$

where the superscript o denotes the weak non-linear feedback iterative count.

2.7. Implementation of Inexact Newton's Methods

2.7.1. General Overview of Inexact Newton's Methods

Consider Eq. (2-10), a non-linear system of equations

$$\bar{\Gamma}(\bar{\Theta}) \equiv \bar{B}(\bar{\Theta}) - \bar{Q} = \bar{0}. \quad (2-49)$$

In the implementation of the exact Newton's method, the Jacobian ($\bar{\bar{J}}_\Gamma$) of the $\bar{\Gamma}$ operator is first determined. If necessary, the Jacobian matrix is factorized (actually the preconditioner matrix which approximates $\bar{\bar{J}}_\Gamma$) so that the results of the factorization can later be used as part of the algorithm for solving the linearized system of equations. Regardless of the solver used, the following linearized system of equations needs to be solved to obtain the next iterative values of the unknown ($\bar{\Theta}_{(m+1)}$):

$$\bar{\Theta}_{(m+1)} = \bar{\Theta}_{(m)} - \left(\bar{\bar{J}}_\Gamma \Big|_{\bar{\Theta}_{(m)}} \right)^{-1} \bar{\Gamma}(\bar{\Theta}_{(m)}), \quad (2-50)$$

where m is the Newton iterative count and $\bar{\bar{J}}_{\Gamma}|_{\bar{\Theta}_{(m)}}$ is the Jacobian of $\bar{\Gamma}$ evaluated at the current iterative value of the unknowns ($\bar{\Theta}_{(m)}$). The algorithm for the overall Newton's method is shown in Figure 2.11.

2.7.2. The Chord Method

The costs for computing the Jacobian coefficient matrix and its factorization are significant. One approach to reduce this cost is to move these operations outside the loop indicated on Figure 2.11. This implies that the linear approximation of $\bar{\Gamma}(\bar{\Theta}) = \bar{0}$ solved at each Newton iteration has the derivative (Jacobian coefficient matrix) determined at only the initial iteration. This method is called the Chord method [27] and the algorithm for this method is shown in Figure 2.12. For the Chord method, the linearized system of equations solved at each iteration can be symbolically written as

$$\bar{\Theta}_{(m+1)} = \bar{\Theta}_{(m)} - \left(\bar{\bar{J}}_{\Gamma}|_{\bar{\Theta}_{(0)}} \right)^{-1} \bar{\Gamma}(\bar{\Theta}_{(m)}). \quad (2-51)$$

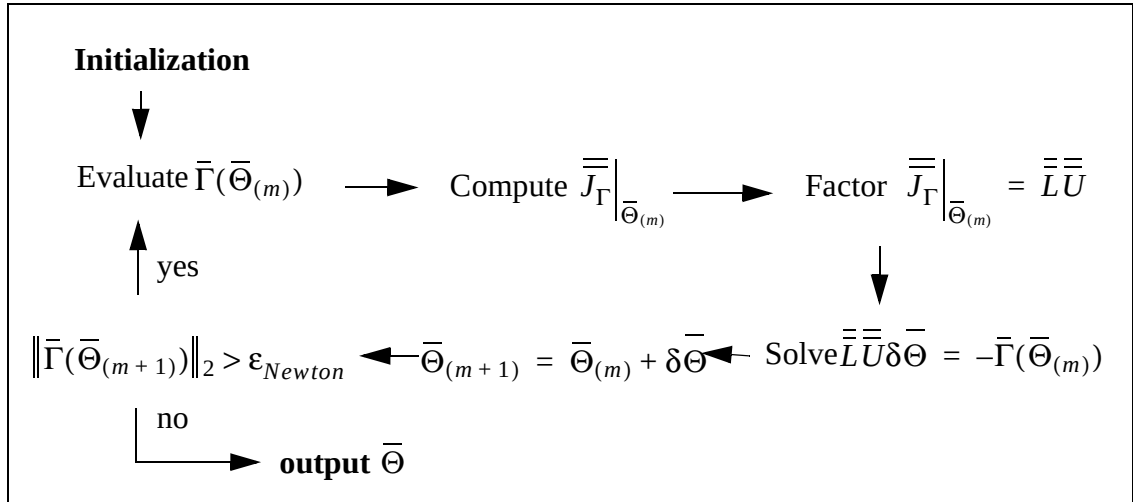


Figure 2.11: Exact Newton algorithm.

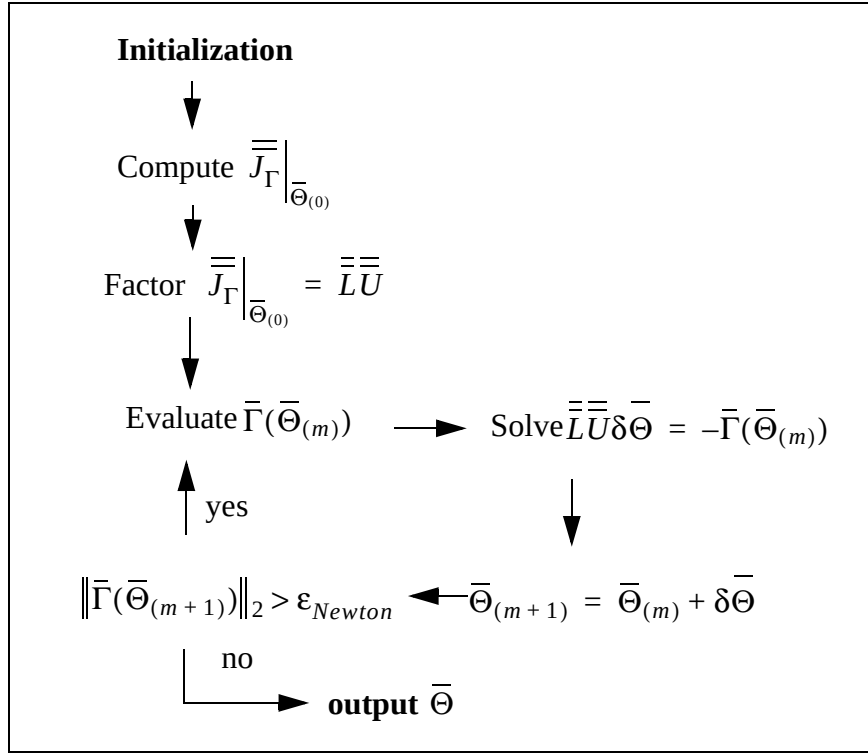


Figure 2.12: Chord method.

Although the cost per Newton iteration can be considerably reduced by utilizing the Chord method, there is a trade-off between this reduction and the increase in the total number of Newton iterations due to the approximation introduced by this method. Therefore, a successful execution of a Chord method will highly depend upon the efficiency of the linearized equations solver. The Chord method also has a higher likelihood of not converging due to misdirection of the search.

2.7.3. The Shamanskii's Method

The Shamanskii's method [27] is introduced to help alleviate the excessive increase in the Newton iterations resulting from utilizing the Chord method by providing a better approximation of the Jacobian coefficient matrix. This is accomplished by updating the Jacobian coefficient matrix at a certain frequency. The algorithm for this method is

shown in Figure 2.13. For the Shamanskii's method, the linearized system of equations solved at each iteration can be written as

$$\bar{\Theta}_{(m+1)} = \bar{\Theta}_{(m)} - \left(\bar{J}_{\Gamma} \Big|_{\bar{\Theta}_{(\tilde{m}(m))}} \right)^{-1} \bar{\Gamma}(\bar{\Theta}_{(m)}), \quad (2-52)$$

where $\tilde{m}(m)$ indicates the past Newton iteration during which the Jacobian matrix is updated for the m^{th} Newton iteration. The number of Newton iterations for the Shamanskii's method should fall between those of exact Newton and Chord methods.

In utilizing the Shamanskii's method, a parametric study should be performed to determine the optimum frequency for the Jacobian matrix update. An update frequency of N denotes that the Jacobian matrix is updated every N Newton iterations. Therefore, using an update frequency of 1 makes the Shamanskii's method equivalent to the exact Newton's method and using an update frequency of ∞ makes it equivalent to the Chord method.

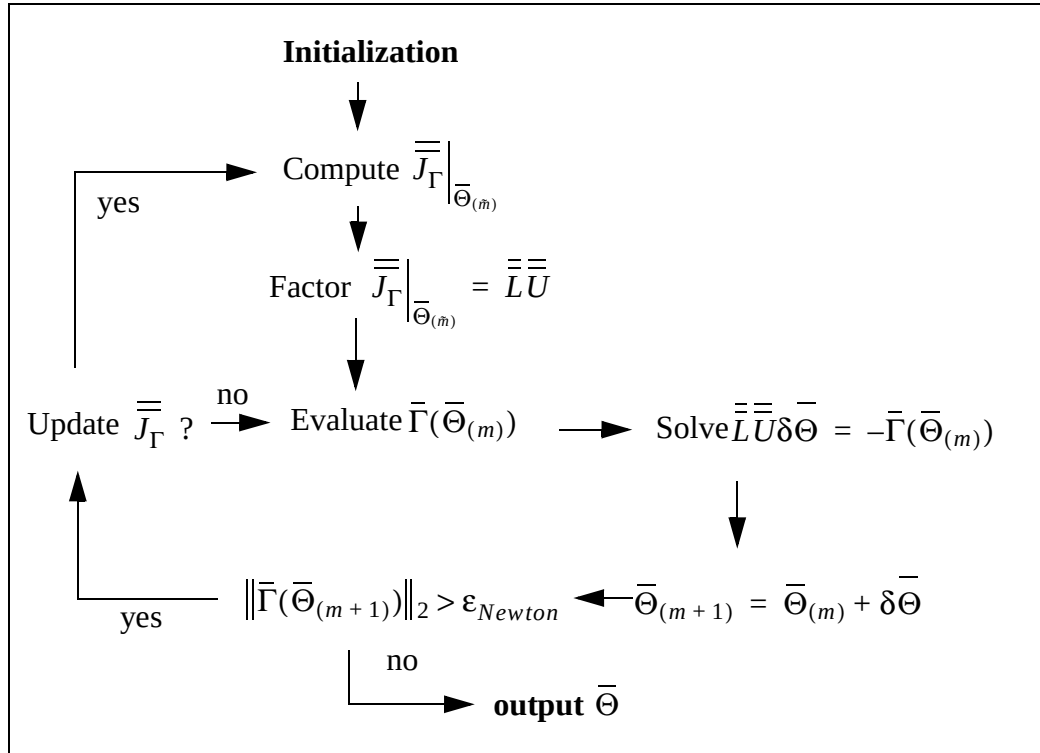


Figure 2.13: Shamanskii's method.

3. Development of Preconditioners

In general, the rate of convergence of iterative methods depends on the spectral properties of the coefficient matrix. For Krylov methods, the condition number of the coefficient matrix plays an important role in determining the rate of convergence. One may attempt to transform the original linear system into an equivalent system in the sense that it still has the same solution, but it has a more favorable condition number. Such a transformation can be obtained by utilizing a preconditioner matrix. For example, if matrix P approximate the coefficient matrix A in some sense, then the transformed system of equations

$$P_L^{-1}Ax = P_L^{-1}b \quad (3-1)$$

has the same solution as the original system of equations $Ax = b$, but the condition number of $P_L^{-1}A$ may be more favorable. This approach is called left-preconditioning. Another approach is to solve the following system of equations

$$\tilde{A}\tilde{x} = b, \quad (3-2)$$

where $\tilde{A} \equiv AP_R^{-1}$ and $\tilde{x} \equiv P_R x$. This approach is called right-preconditioning and also produces the same solution as the original linear system.

One needs to realize that utilizing a preconditioner in a Krylov method adds some extra computational time both in the initial setup and during the iterations for applying it. However, there is a trade-off between the increasing computational time associated with constructing and applying the preconditioner and the advantage of increased convergence rate. Therefore, the focus of formulating a preconditioner is to minimize the additional incremental computational time while maximizing the convergence rate so as to reduce the overall computational time.

Taking advantage of the structure of the coefficient matrix is a crucial factor in devising a preconditioner. Another essential factor in the development of an effective

preconditioner is an understanding of the physics behind the constitutive equations forming the linear system. For example, in approximating the original coefficient matrix, one can eliminate some elements of the matrix which are known to be not too important in the sense that ignoring them will still produce a solution close to the exact solution.

3.1. A Block Incomplete LU Factorization

In formulating the preconditioners described in this chapter, a block incomplete LU (BILU) factorization idea [22] is utilized as a part of the preconditioners. The factorization was selected to be incorporated in the preconditioner since it is a straightforward process which requires a predetermined fixed number of operations dictated by the size of the problem and can capture the stronger spatial and/or energy coupling of unknowns. Also note that the size of the matrix block to be factorized also varies depending upon the arrangement of the unknowns.

Although detailed derivations of the BILU factorization will not be given here, the author will briefly discuss the basic idea of this factorization. Consider a linear system of equations, derived from the discretization of the three-dimensional, two-group neutron diffusion equation solved during the inner iteration part of the “outer-inner” iteration, which can be written as follow

$$\bar{\Delta}^{(3)} \bar{\psi} = \bar{\sigma}. \quad (3-3)$$

The superscript “(3)” in $\bar{\Delta}^{(3)}$ indicates that the coefficient matrix represents the three-dimensional space modeled by the equation. The coefficient matrix has a block septa-diagonal structure, assuming a proper ordering of the unknowns, and the individual blocks composing the coefficient matrix are 2x2 matrices, representing the two energy groups. Realizing that our discussion involves a lot of matrix blocks with different sizes, we assume that the model has Λ nodes in the x-, y-, and z- directions so that it is easier to describe various block sizes.

If we group each vertical plane composing the three-dimensional model, we can view the $\bar{\bar{\Delta}}^{(3)}$ coefficient matrix as a block tri-diagonal matrix with an individual block size of $2\Lambda^2 \times 2\Lambda^2$. The coefficient matrix can now be written as follow:

$$\bar{\bar{\Delta}}^{(3)} = \begin{bmatrix} \bar{\bar{\Delta}}_1^{(2)} & \bar{\bar{U}}_1^{(3)} & & & \\ \bar{\bar{L}}_2^{(3)} & \bar{\bar{\Delta}}_2^{(2)} & \bar{\bar{U}}_2^{(3)} & & \\ & \dots & \dots & \dots & \\ & & \bar{\bar{L}}_{\Lambda-1}^{(3)} & \bar{\bar{\Delta}}_{\Lambda-1}^{(2)} & \bar{\bar{U}}_{\Lambda-1}^{(3)} \\ & & & \bar{\bar{L}}_{\Lambda}^{(3)} & \bar{\bar{\Delta}}_{\Lambda}^{(2)} \end{bmatrix}, \quad (3-4)$$

where the $\bar{\bar{\Delta}}^{(2)}$ represents the coefficient matrix of each two-dimensional vertical plane. Technically, we can completely factorize $\bar{\bar{\Delta}}^{(3)}$ by employing the following recursive relation:

$$\bar{\bar{\Omega}}_1 = \bar{\bar{\Delta}}_1^{(2)} \quad (3-5)$$

$$\bar{\bar{\Omega}}_l = \bar{\bar{\Delta}}_l^{(2)} - \bar{\bar{L}}_l^{(3)} \bar{\bar{\Omega}}_{l-1} \bar{\bar{U}}_{l-1}^{(3)}, \text{ for } l = 1 \dots \Lambda, \quad (3-6)$$

to obtain the LU factorization:

$$\bar{\bar{\Delta}}^{(3)} = (\bar{\bar{L}} + \bar{\bar{\Omega}}) \bar{\bar{\Omega}}^{-1} (\bar{\bar{U}} + \bar{\bar{\Omega}}) = (\bar{\bar{L}} + \bar{\bar{\Omega}}) (\bar{\bar{\Omega}}^{-1} \bar{\bar{U}} + \bar{\bar{I}}), \quad (3-7)$$

where $\bar{\bar{L}}$ and $\bar{\bar{U}}$ are the strictly lower and upper triangular part of $\bar{\bar{\Delta}}^{(3)}$, and $\bar{\bar{\Omega}}$ is a block diagonal matrix consisting of $\bar{\bar{\Omega}}_l$'s.

Similarly, for each vertical plane, the coefficient matrix $\bar{\bar{\Delta}}^{(2)}$, which has a block penta-diagonal structure, can be viewed as a block tri-diagonal matrix with an individual block size of $2\Lambda \times 2\Lambda$. Therefore, this coefficient matrix can be factorized in the same manner as described above. This process continues to the 1-D level where the coefficient matrix is a true block tri-diagonal matrix with an individual block size of 2×2 .

The main issue in performing the complete LU factorization is the computational cost of calculating the inverse of the diagonal components since they become fuller matrices as the dimensionality of the problem increases. An approximation is introduced [22] whereby some elements of the inverse of the diagonal component are eliminated so that the inverse has the same sparsity as the original diagonal component.

3.2. Porsching's Algorithm

The idea for this algorithm is introduced in a paper by Porsching *et. al.*[43] on hydraulic networks. Using this algorithm, the components of the coefficient matrix of a linear system, $\bar{\bar{A}}\bar{x} = \bar{b}$, are rearranged if possible into the following block form

$$\bar{\bar{A}} = \begin{bmatrix} \bar{\bar{A}}_{11} & \bar{\bar{A}}_{12} \\ \bar{\bar{A}}_{21} & 0 \end{bmatrix}, \quad (3-8)$$

where, despite the matrix notation, $\bar{\bar{A}}_{12}$ and $\bar{\bar{A}}_{21}$ are actually column and row vectors, respectively. In the discussion, note that matrix notation will always be used to denote the submatrix component even though sometimes the vector notation is more appropriate. The system of equations to be solved can then be written as

$$\begin{bmatrix} \bar{\bar{A}}_{11} & \bar{\bar{A}}_{12} \\ \bar{\bar{A}}_{21} & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \end{bmatrix} = \begin{bmatrix} \bar{b}_1 \\ \bar{b}_2 \end{bmatrix}. \quad (3-9)$$

The algorithm for solving this system of equations is straight-forward and is described in Figure 3.1. Porsching's algorithm is used as a part of the preconditioners introduced in the next three subsections.

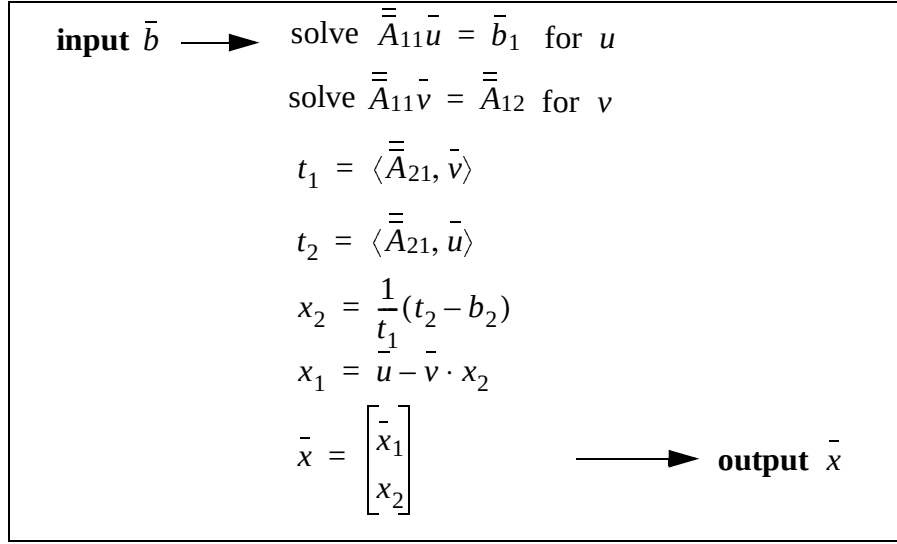


Figure 3.1: Porsching's algorithm.

3.3. Preconditioner 1

The first type of preconditioners examined is using a node-wise grouping of the unknowns, where all of the unknowns belonging to a particular node are grouped together. The preconditioner can be symbolically written as

$$\bar{\bar{P}}_1 = \begin{bmatrix} \bar{\bar{\hat{J}}}_{11} & \bar{\bar{\hat{J}}}_{12} \\ \bar{\bar{\hat{J}}}_{21} & 0 \end{bmatrix}, \quad (3-10)$$

where the “hat” indicates the re-ordered Jacobian matrix block from what was introduced earlier.

The structure of this preconditioner is depicted in Figure 3.2. From this figure, one can see clearly the seven banded structure of the $\bar{\bar{\hat{J}}}_{11}$ block which represents the underlying nodes coupling of the finite difference method applied to the neutron diffusion equation. Recall that we are using a nodal method but employing the non-linear iterative nodal method, which gives rise to the finite difference method spatial coupling. Also included in $\bar{\bar{\hat{J}}}_{11}$ are the thermal hydraulics equations. Each entry of the $\bar{\bar{\hat{J}}}_{11}$ block is a 6x6

matrix, while the entries of $\bar{\bar{J}}_{12}$ and $\bar{\bar{J}}_{21}$ blocks are 6x1 and 1x6 vectors, respectively, associated with the criticality constant.

This preconditioner is formulated based on the fact that the more components of the original Jacobian matrix are included in the preconditioner matrix, the faster the convergence rate of the Krylov solver will be. The algorithm for solving a preconditioner matrix system (*i.e.*, $\bar{\bar{P}}_1 \bar{w} = \bar{z}$) during Krylov iterations is given in Figure 3.3. In the outer shell of the preconditioner, the Porsching's algorithm is utilized to take advantage of the structure of the matrix depicted in Eq. (3-10). Within the Porsching's algorithm itself, we use a BILU approximation to determine the action of $\left(\bar{\bar{J}}_{11}\right)^{-1}$, which equates to solving

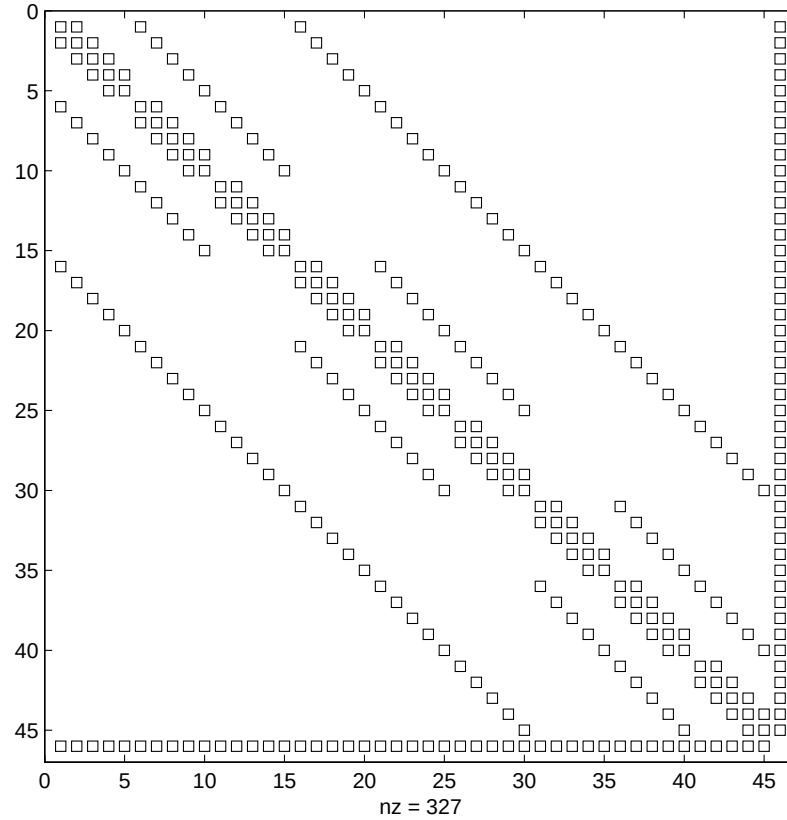


Figure 3.2: Structure of Preconditioner Type 1.

for $\bar{u}$ and $\bar{v}$ in Figure 3.3. As clearly indicated in the figure, the $\bar{v}$ vector and t_1 term are saved in order to reduce the overall execution time.

3.4. Preconditioner 2

A core-wise grouping of the unknowns is used in the second type of preconditioner. This grouping allows us to relate parts of the Jacobian to the underlying balance equations since they are distinctly separated from one another. The preconditioner can be symbolically written as

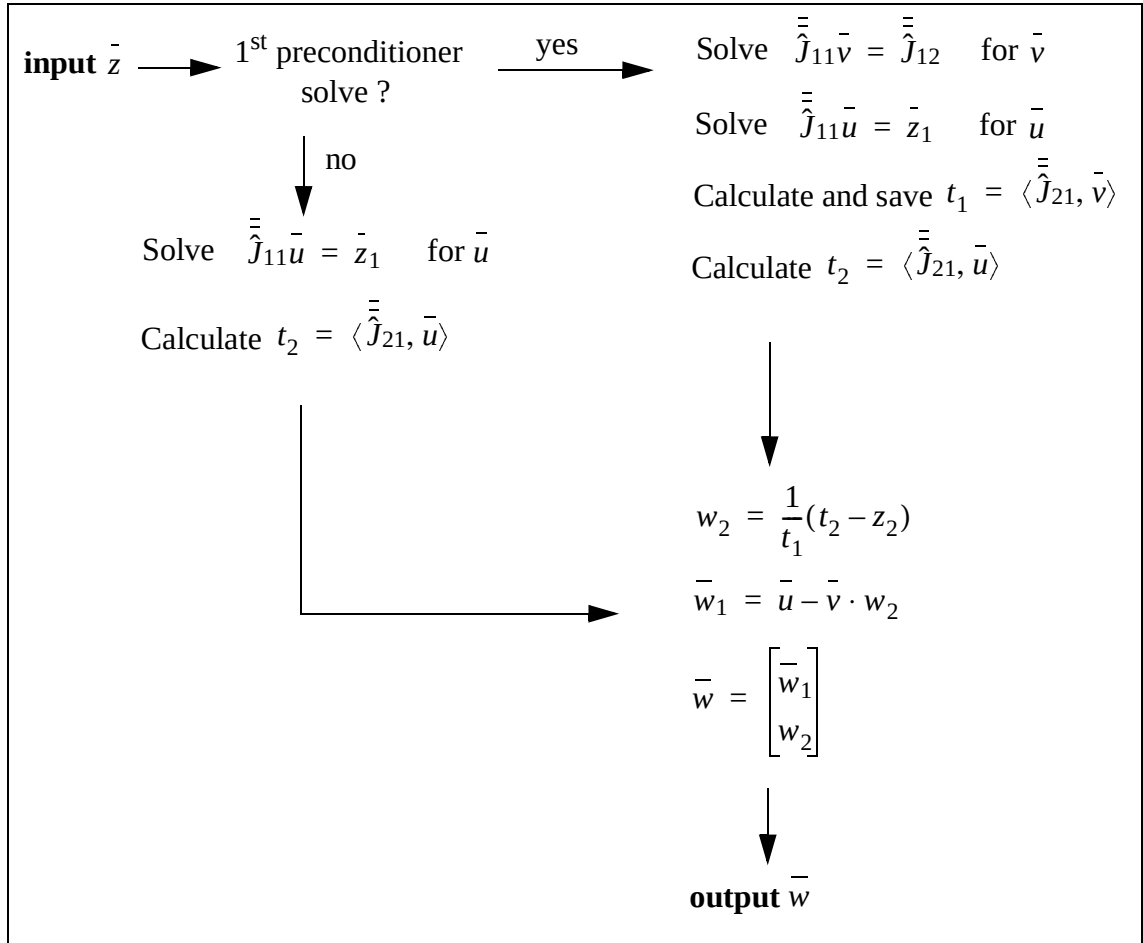


Figure 3.3: Preconditioner 1 algorithm.

$$\bar{\bar{P}}_2 = \begin{bmatrix} \bar{\bar{J}}_{11} & \bar{\bar{J}}_{12} & \bar{\bar{J}}_{13} \\ 0 & \bar{\bar{J}}_{22} & 0 \\ \bar{\bar{J}}_{31} & \bar{\bar{J}}_{32} & 0 \end{bmatrix}. \quad (3-11)$$

Seven-banded structures can be observed in blocks $\bar{\bar{J}}_{11}$ and $\bar{\bar{J}}_{12}$ depicting the underlying couplings of the finite difference method applied to the neutron diffusion equation. Block $\bar{\bar{J}}_{22}$ is a block diagonal matrix with each diagonal component being a lower triangular matrix, denoting the axial coupling of the thermal-hydraulics equations. In order to apply the Porsching's algorithm, this preconditioner can also be written in the same manner as Eq. (3-10), where by inspection we can write

$$\bar{\bar{J}}_{11} = \begin{bmatrix} \bar{\bar{J}}_{11} & \bar{\bar{J}}_{12} \\ 0 & \bar{\bar{J}}_{22} \end{bmatrix}, \quad (3-12)$$

$$\bar{\bar{J}}_{12} = \begin{bmatrix} \bar{\bar{J}}_{13} \\ 0 \end{bmatrix}, \quad (3-13)$$

and

$$\bar{\bar{J}}_{21} = \begin{bmatrix} \bar{\bar{J}}_{31} & \bar{\bar{J}}_{32} \end{bmatrix}. \quad (3-14)$$

The algorithm for solving a preconditioner matrix system (*i.e.*, $Pw = z$) is shown in Figure 3.4. Note that subscripts 1 to 3 used by the right hand side and the solution vectors refer to the original three row blocks of the Jacobian matrix. When calculating the action of $\bar{\bar{J}}_{11}^{-1}$, the BILU approximation is used to determine the action of $\bar{\bar{J}}_{11}^{-1}$. In addition, since each diagonal block of the $\bar{\bar{J}}_{22}$ submatrix is a lower triangular matrix associated with the coolant flow up a coolant channel, the action of $\bar{\bar{J}}_{22}^{-1}$ can be completed by performing a

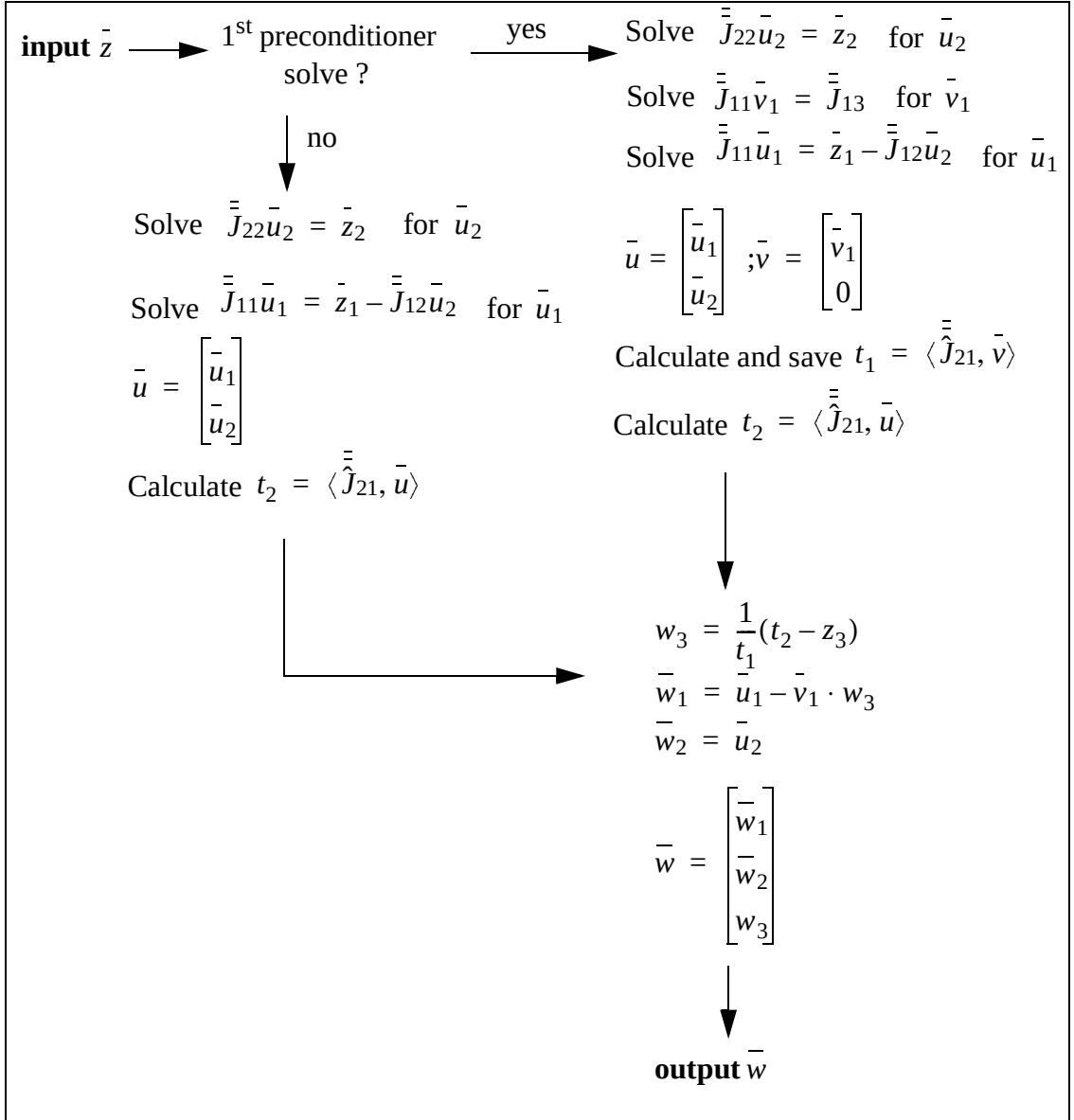


Figure 3.4: Preconditioner 2 algorithm.

forward sweep. Also note that some variables are saved during the first pass through the preconditioner to reduce the computational cost.

This preconditioner was formulated to reduce the cost of factorizing the $\bar{\bar{J}}_{11}$ block and subsequently the cost of performing the BILU approximation of Preconditioner 1. While the same block seven banded structure is possessed by both the $\bar{\bar{J}}_{11}$ block of

Preconditioner 1 and the $\bar{\bar{J}}_{11}$ block of Preconditioner 2, the size of each entry is significantly reduced from 6x6 in Preconditioner 1 to just 2x2 in Preconditioner 2. However, while in Preconditioner 1 all entries of the Jacobian matrix are used in the preconditioner, in Preconditioner 2 the $\bar{\bar{J}}_{21}$ block, which physically represent the changes in the thermal hydraulics properties as a result of changes in flux (power) distribution, is eliminated. The rest of the Krylov algorithm, where all entries of the Jacobian matrix are considered, will correct for this imperfection.

3.5. Preconditioner 3

Preconditioner 3 also employs the core-wise grouping of the unknowns. However, the original Jacobian matrix is slightly rearranged in this preconditioner. The preconditioner can be symbolically written as

$$\bar{\bar{P}}_3 = \begin{bmatrix} \bar{\bar{J}}_{11} & \bar{\bar{0}} \\ \bar{\bar{J}}_{21} & \bar{\bar{J}}_{22} \end{bmatrix}, \quad (3-15)$$

where

$$\bar{\bar{J}}_{11} = \begin{bmatrix} \bar{\bar{J}}_{11} & \bar{\bar{J}}_{13} \\ \bar{\bar{J}}_{31} & 0 \end{bmatrix}, \quad (3-16)$$

$$\bar{\bar{J}}_{21} = \begin{bmatrix} \bar{\bar{J}}_{21} & 0 \end{bmatrix}, \quad (3-17)$$

and

$$\bar{\bar{J}}_{22} = \begin{bmatrix} \bar{\bar{J}}_{22} \end{bmatrix}. \quad (3-18)$$

The algorithm for this preconditioner is outlined in Figure 3.5. Note that the subscripts of the right hand side and the solution vectors still follow the original Jacobian matrix block

ordering. When solving the preconditioner matrix system, due to the structure of the matrix, changes in the flux distribution and eigenvalue can now be calculated first and subsequently used for determining the changes in the thermal hydraulics properties. This is similar to the fission source iterations and thermal hydraulics update in the traditional iterative solver. The Porsching's algorithm is used when determining the action of $\bar{J}_{11}^{-1}$.

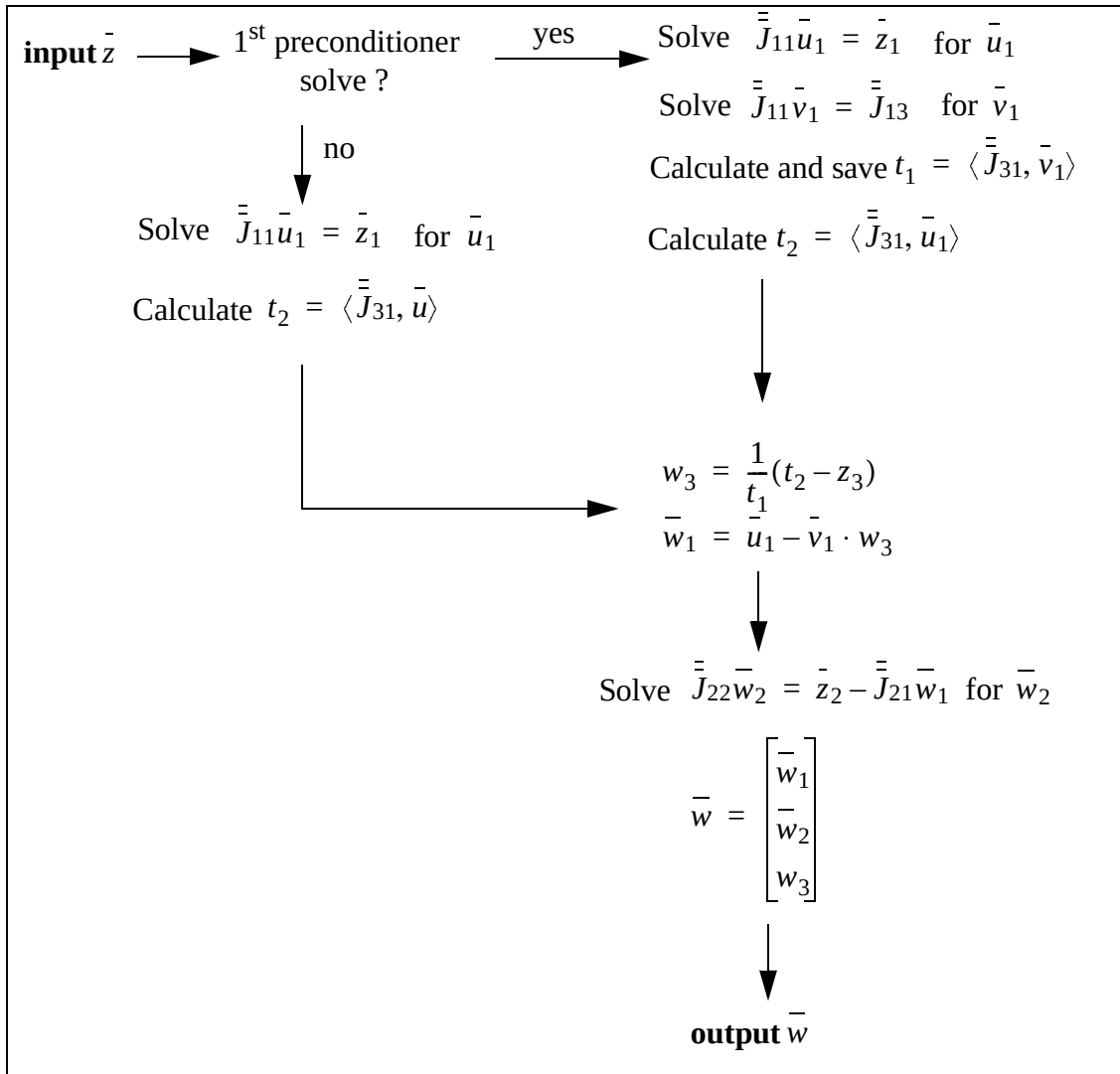


Figure 3.5: Preconditioner 3 algorithm.

The BILU approximation is used to determine the action of $\bar{J}_{11}^{-1}$ within the Porsching's algorithm.

This preconditioner is devised after considering the alternative of this preconditioner, in which the changes in thermal hydraulics properties are calculated first and then used in determining the changes in flux distribution and eigenvalue. The advantage of Preconditioner 3 is the fact that the costs for matrix-vector multiplications involving the $\bar{J}_{21}$ block (a diagonal matrix) are greatly reduced compared to the ones involving the $\bar{J}_{12}$ block (a seven-banded matrix).

3.6. Performance of the Preconditioners

Numerical experiments have been performed to examine the performance of these preconditioners. A preliminary study was completed using three Krylov methods, namely BICGSTAB, CGS, and restarted GMRES methods. The purpose of this preliminary study is to select a Krylov method to be employed in the final implementation of the Newton-Krylov solver and to observe the performance of the preconditioners. The chosen Krylov method will be examined further to check whether the conclusion from this preliminary study still holds.

For the preliminary study, a tight stopping criteria is used for the Krylov solvers to avoid Krylov iteration error contamination of the Newton iteration. As a result, the same number of Newton iterations is observed for all cases, enabling us to fairly compare the number of Krylov iterations from these cases. The results are summarized in Table 3.1 and Table 3.2. The stopping criteria for the Krylov iterations is $\epsilon_{Krylov} = 5 \times 10^{-5}$ and the stopping criteria for the Newton iteration is $\epsilon_{true} = 1 \times 10^{-3}$. The true error is defined as

$$\epsilon_{true} = \left\| \frac{\bar{\Phi}^{(m)} - \bar{\Phi}^{(ref)}}{\bar{\Phi}^{(ref)}} \right\|_2 = \sqrt{\frac{1}{2L} \sum_{l=1}^L \left\{ \left[\frac{\phi_{1,l}^{(m)} - \phi_{1,l}^{(ref)}}{\phi_{1,l}^{(ref)}} \right]^2 + \left[\frac{\phi_{2,l}^{(m)} - \phi_{2,l}^{(ref)}}{\phi_{2,l}^{(ref)}} \right]^2 \right\}}, \quad (3-19)$$

where L is the total number of spatial fuel nodes and m represents the current Newton iteration count. The reference flux solution is obtained using the traditional iterative algorithm with a set of extremely tight stopping criteria. Also note that the preliminary study was performed on a somewhat simplified 368 assembly GE BWR/4 core model. Although the geometry of the core is maintained, the core is loaded with only three types of fuel bundles and the core does not have radial reflectors. In addition, since the spacer-void model is not included in the derivation of the Jacobian matrix, algebraic variable reduction allows us to reduce the number of thermal hydraulics unknowns included in the Newton's method treatment to three, namely the node-wise moderator pressure, density, and internal energy.

Before analyzing the results in these tables, the author would like to explain variables being observed in Table 3.2. Initialization time covers everything involved in the process of setting up the initial system of equations, *e.g.* determination of initial guesses of flux distribution, nodal power, and thermal hydraulics variables distribution. Computational times spent for completing calculations of the cross section derivatives with respect to the moderator density and fuel temperature, calculations of elemental derivatives as described in the appendix, and setting up the Jacobian matrix from the elemental components are included in the "Jacobian Matrix Setup." Matrix factorization

Table 3.1: Preconditioners Performance for a Simplified 368 Assembly BWR/4 Core Model.

Variable	BICGSTAB			CGS			GMRES(5)		
	P_1	P_2	P_3	P_1	P_2	P_3	P_1	P_2	P_3
Krylov Iterations	44	47	51	57	61	63	93	130	127
Newton Iterations	5	5	5	5	5	5	5	5	5
Execution Time (sec ¹)	3.88	3.15	2.97	4.47	3.60	3.28	5.42	5.35	4.84

1: Timing was performed on a 440 MHz SUN Ultra10 machine.

Table 3.2: Detail Time Allocation (in seconds¹).

Variable	BICGSTAB			CGS			GMRES(5)		
	P_1	P_2	P_3	P_1	P_2	P_3	P_1	P_2	P_3
Initialization	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
Newton part of the solver									
Jacobian Matrix Setup	0.80	0.80	0.80	0.79	0.79	0.81	0.79	0.79	0.81
Matrix Factorization	0.28	0.07	0.07	0.28	0.07	0.07	0.28	0.07	0.07
Variable Updates	0.21	0.21	0.21	0.21	0.21	0.21	0.21	0.21	0.21
Setup FDM Matrix and RHS vector	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10	0.10
Total Newton	1.39	1.18	1.18	1.38	1.17	1.19	1.38	1.17	1.19
Krylov Part of the solver									
Preconditioner Solve	1.61	1.03	0.85	2.08	1.37	1.09	2.24	1.82	1.39
Matrix-vector Operation	0.43	0.49	0.47	0.53	0.59	0.53	0.93	1.29	1.20
Inner Product	0.06	0.07	0.07	0.04	0.04	0.04	0.03	0.04	0.04
Vector Updates	0.10	0.10	0.11	0.15	0.14	0.14	0.51	0.70	0.69
Other operations in Krylov	0.02	0.03	0.03	0.02	0.02	0.02	0.05	0.06	0.06
Total Krylov	2.22	1.72	1.52	2.82	2.16	1.82	3.76	3.91	3.38
Others	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02

1: Timing was performed on a 440 MHz SUN Ultra10 machine.

represents the cost of performing the BILU factorization. The times for setting up the right hand side vector for the Newton iteration and setting up the NEM/CMFD coefficient

matrix $(\bar{\bar{M}}(\bar{\xi}, \bar{\Psi}))$ is recorded under “Setup FDM Matrix and RHS vector.” The “Variable Updates” category encompasses all calculations involved in determining the new state point for the Newton iteration. This process involves what can be considered as direct and indirect updates. For the direct updates, the values of the flux distribution, pressure, density, internal energy, void fraction, and the core eigenvalue are directly calculated using the old values and the changes just calculated from solving the linearized system of equations. For the indirect updates, the remaining thermal hydraulics properties are calculated using the updated variables from the direct update. It is not necessary to explain the categories for the Krylov part of the table since they are fairly self-explanatory.

From Table 3.1 one can see that the number of Newton iterations does not change since the Krylov stopping criteria is tight enough. From this table, one can observe that Preconditioner 1, which carries all elements of the Jacobian matrix, needs the fewest number of iterations to converge. However, since the cost per Krylov iteration is adequately reduced in Preconditioners 2 and 3, the overall execution times decrease for these preconditioners despite a slight increase in total Krylov iterations. From Table 3.2, one can calculate the average times per Krylov iteration as 0.047 sec, 0.034 sec, and 0.029 sec for Preconditioner 1, 2, and 3, respectively.

The advantage of performing the BILU factorization on a smaller matrix as employed by Preconditioners 2 and 3 is numerically confirmed by Table 3.2. For the Newton part of the solver, while other components stay relatively constant, the cost for Jacobian matrix factorization is reduced by a factor of four. Subsequently when applying the preconditioner in the Krylov part of the solver, one can see that the costs for the solution of the preconditioner matrix system for Preconditioners 2 and 3 are also smaller than the cost for Preconditioner 1. Finally, for the Krylov solver part of the preliminary study, observing the results shown in Table 3.1, one can see that the BICGSTAB method outperforms the other Krylov methods and, therefore, will be employed in the final implementation of the Newton-Krylov solver along with Preconditioner 3.

Figure 3.6 presents an example of more detailed information about the convergence behavior of the preconditioners within a Newton iteration. The BICGSTAB

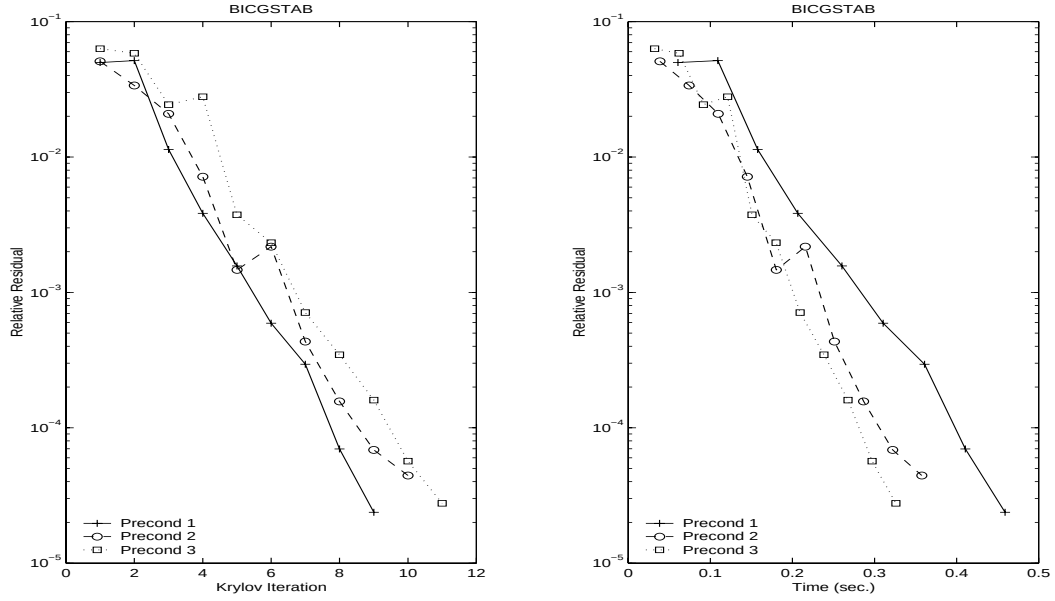


Figure 3.6: Convergence Behavior of the Preconditioners.

method is used for illustrative purpose. The plot on the left shows that Preconditioner 1 provides the highest convergence rate. However, the plot on the right depicts how more computationally intensive this preconditioner is compared to the other two.

Additional studies were performed to examine the performance of the preconditioners. For this study, a more realistic 368 assembly GE BWR/4 core is utilized. Also note that for this study the spacer-void model has been explicitly included as a part of the Jacobian matrix system. This modification adds the void fraction as an additional unknown. While this change does not impact the LU factorization cost for Preconditioners 2 and 3 since the submatrix to be factorized for these preconditioners only has the neutronics part of the problem, *i.e.* the derivative of the neutron diffusion equation, it changes the LU factorization cost of Preconditioner 1 dramatically since now each entry of the $\hat{J}_{11}$ block becomes a 6x6 versus 5x5 matrix.

The results from this study are presented in Table 3.3. While the cost for performing other parts of the Newton iterations remains unchanged, the cost of the Jacobian matrix factorization for Preconditioners 2 and 3 is now reduced more significantly, by a factor of ~ 20 , with respect to Preconditioner 1. Subsequently, compared

to Preconditioner 1, the cost of the preconditioner solve is reduced by 45% and 70% for Preconditioners 2 and 3, respectively. Regardless of the BICGSTAB stopping criteria, the

Table 3.3: Preconditioner Performance (in seconds¹) for a Realistic 368 Assembly GE BWR/4 Core Model.

Variable	$\epsilon_{Krylov} = 1 \times 10^{-3}$			$\epsilon_{Krylov} = 1 \times 10^{-4}$			$\epsilon_{Krylov} = 1 \times 10^{-5}$		
	P_1	P_2	P_3	P_1	P_2	P_3	P_1	P_2	P_3
Initialization	0.048	0.048	0.048	0.048	0.048	0.049	0.049	0.048	0.048
Newton part of the solver									
Jacobian Matrix Setup	1.381	1.378	1.377	1.380	1.383	1.379	1.379	1.375	1.383
Jacobian Factorization	0.634	0.029	0.030	0.622	0.030	0.030	0.620	0.030	0.030
Variable Updates	0.514	0.514	0.514	0.513	0.514	0.514	0.513	0.514	0.515
Setup FDM Matrix and RHS Vector	0.088	0.088	0.088	0.088	0.088	0.088	0.088	0.088	0.087
Krylov part of the solver									
Preconditioner Solve	4.625	2.487	1.418	5.287	2.768	1.578	5.847	3.134	1.809
Matrix-Vector Operation	2.909	3.136	3.106	3.349	3.498	3.511	3.789	4.024	4.002
Inner Product	0.117	0.129	0.130	0.138	0.144	0.146	0.156	0.165	0.162
Vector Updates	0.217	0.237	0.233	0.250	0.267	0.262	0.281	0.310	0.297
Iterations									
Newton Iteration	8	8	8	8	8	8	8	8	8
Krylov Iteration	46	51	51	53	57	57	60	65	65
Avg. Krylov/Newton	5.75	6.38	6.38	6.63	7.13	7.13	7.50	8.13	8.13

1: Timing was performed on a 440 MHz SUN Ultra-10 machine.

results presented in this table exhibit a consistent trend with the results from the preliminary study; hence, we will utilize the BICGSTAB solver along with Preconditioner 3 in the final implementation of the Newton-Krylov solver.

4. Implementation of BICGSTAB as the Inner Iteration Solver.

4.1. Introduction

Another means to provide speedup to the current FORMOSA-B core simulator solver is to implement a preconditioned Krylov solver in the FORMOSA-B code as the inner iteration solver replacing the red/black line successive over relaxation (R/B LSOR) solver. The implementation of the BICGSTAB solver, with both block incomplete LU factorization and domain decomposition preconditioners, into the NESTLE code [52] had shown that the BICGSTAB solver outperforms the R/B LSOR method for completing the inner iteration. Therefore, prior to implementing the Newton-Krylov solver, we would like to know how much speedup should be expected from just the implementation of the Krylov method, *i.e.* the BICGSTAB method, in FORMOSA-B.

Following this introduction section, the idea of the inner iteration is introduced. Then in the ensuing section, the traditional R/B LSOR solver currently implemented in the FORMOSA-B code is presented. The next sub-section briefly discusses the BICGSTAB solver implementation and, finally, Section 4.5 summarizes the results from this study.

4.2. General Overview of the Inner Iteration.

The discretized form of the two-group neutron diffusion equation assuming no upscattering term and no fission yield in the thermal group, as shown in Eq. (2-1), can be expressed as

$$\bar{\bar{M}}\bar{\phi} = \lambda\bar{\bar{F}}\bar{\phi}, \quad (4-1)$$

where

$$\overline{\overline{M}} = \begin{bmatrix} \overline{\overline{M}}_1 & 0 \\ -\overline{\overline{\Sigma}}_{12} & \overline{\overline{M}}_2 \end{bmatrix}, \quad (4-2)$$

$$\overline{\overline{F}} = \begin{bmatrix} \overline{\overline{F}}_1 & \overline{\overline{F}}_2 \\ 0 & 0 \end{bmatrix}, \quad (4-3)$$

and

$$\overline{\phi} = \begin{bmatrix} \overline{\phi}_1 \\ \overline{\phi}_2 \end{bmatrix}. \quad (4-4)$$

Note that the $\overline{\overline{M}}$ and $\overline{\overline{F}}$ are square matrices of order $L \times 2$, where L is the total number of nodes in the model. The elements of the unknown vector, $\overline{\phi}$, are ordered axially from bottom to top in each fuel bundle, fuel bundle by fuel bundle, and finally group by group.

As a result of this ordering, the $\overline{\overline{M}}_g$ sub-matrix for group g has a dimension of $L \times L$ and has a tri-diagonal, penta-diagonal, or septa-diagonal structure depending upon whether a one-, two- or three-dimensional problem is being solved.

The power method [58] is utilized to solve this generalized eigenvalue problem. As part of the iterations in the power method implementation, we have to solve a linear system of equations of the form

$$\overline{\overline{M}}\overline{\phi} = \overline{S}, \quad (4-5)$$

where $\overline{S}$ is the current fission source vector. This solution is called the inner iteration since one will need another iterative solver to solve this linear system of equations.

Since there is no upscattering term, the coefficient matrix $\overline{\overline{M}}$ is a triangular matrix and, therefore, the unknown vector, $\overline{\phi}$, can be solved group-by-group starting at the highest energy group and then marching through the lower energy groups. This is the approach used when performing the inner iteration using the R/B LSOR method. When

the BICGSTAB method is used, the ordering of the unknowns is slightly changed and will be discussed in Section 4.4.

4.3. The Traditional R/B LSOR Solver

The R/B LSOR method is a variant of the SOR method. This method colors the z-lines red and black such that every red z-line is surrounded by four black z-lines and vice versa. Note that a z-line defines a group of axial nodes corresponding to a single fuel bundle in FORMOSA-B. Grouping the z-lines this way allows us to solve for the unknowns corresponding to a red z-line, independent of other red z-lines, using just the information from the black z-lines surrounding the bundle of interest, enabling a two-cyclic iterative approach.

The R/B LSOR approach is done for each energy group. We can partition the $\overline{\overline{M}}_g$ sub-matrix and write the group g linear system of equations, suppressing the g notation for brevity, as follow

$$\begin{bmatrix} \overline{\overline{M}}_R & \overline{\overline{H}} \\ \overline{\overline{H}}^T & \overline{\overline{M}}_B \end{bmatrix} \begin{bmatrix} \overline{\phi}_R \\ \overline{\phi}_B \end{bmatrix} = \begin{bmatrix} \overline{S}_R \\ \overline{S}_B \end{bmatrix}. \quad (4-6)$$

The $\overline{\overline{M}}_R$ sub-matrix represents the coupling of the red axial nodes along a z-line and, similarly, the $\overline{\overline{M}}_B$ sub-matrix represents the coupling of the black axial nodes along a z-line. The couplings between red lines to the black lines and vice versa are represented by $\overline{\overline{H}}$ sub-matrix and its transpose $\overline{\overline{H}}^T$, respectively.

The SOR method associated with Eq. (4-6), can be written as

$$\overline{\overline{M}}_R \overline{\phi}_R^{(n+1)} = \omega [\overline{S}_R - \overline{\overline{H}} \overline{\phi}_B^{(n)}] + (1 - \omega) \overline{\overline{M}}_R \overline{\phi}_R^{(n)} \quad (4-7)$$

$$\overline{\overline{M_B \phi_B}}^{-(n+1)} = \omega \left[\overline{\overline{S_B}} - \overline{\overline{H}}^T \overline{\overline{\phi_R}}^{-(n+1)} \right] + (1 - \omega) \overline{\overline{M_B \phi_B}}^{-(n)} \quad (4-8)$$

where ω is a relaxation parameter and n is the inner iteration count. Eqs. (4-7) and (4-8) are solved in sequence and the process is repeated until a certain stopping criteria on the flux values, $\bar{\phi}$, has been satisfied.

4.4. The BICGSTAB Solver

When utilizing the preconditioned BICGSTAB solver, instead of solving group-by-group at the core-wide level (solving first for the fast group flux followed by the thermal group flux), we solve for both groups simultaneously. In order to do this, a slight reordering of the unknowns is necessary as illustrated in Figure 4.1 for a z-line with three axial nodes. After the reordering, instead of the $\overline{\overline{M}}_g$ sub-matrices, the overall $\overline{\overline{M}}$ matrix in Eq. (4-1) now has a block tri-diagonal, block penta-diagonal, or block septa-diagonal structure depending on the spatial dimension of the problem. Note that the “nuclear”

Fast Group	$\left[\begin{array}{ccc ccc} a_{11}^{(1)} & a_{71}^{(1)} & 0 & a_{21}^{(1)} & 0 & 0 \\ a_{81}^{(2)} & a_{11}^{(2)} & a_{71}^{(2)} & 0 & a_{21}^{(2)} & 0 \\ 0 & a_{81}^{(3)} & a_{11}^{(3)} & 0 & 0 & a_{21}^{(3)} \end{array} \right]$						Fast	$\left[\begin{array}{ccc ccc} a_{11}^{(1)} & a_{21}^{(1)} & a_{71}^{(1)} & 0 & 0 & 0 \\ a_{12}^{(1)} & a_{22}^{(1)} & 0 & a_{72}^{(1)} & 0 & 0 \\ a_{81}^{(2)} & 0 & a_{11}^{(2)} & a_{21}^{(2)} & a_{71}^{(2)} & 0 \end{array} \right]$							
	Thermal Group	$\left[\begin{array}{ccc ccc} a_{12}^{(1)} & 0 & 0 & a_{22}^{(1)} & a_{72}^{(2)} & 0 \\ 0 & a_{12}^{(2)} & 0 & a_{82}^{(2)} & a_{22}^{(2)} & a_{72}^{(2)} \\ 0 & 0 & a_{12}^{(3)} & 0 & a_{82}^{(3)} & a_{22}^{(3)} \end{array} \right]$						Thermal	$\left[\begin{array}{ccc ccc} 0 & a_{82}^{(2)} & a_{12}^{(2)} & a_{22}^{(2)} & 0 & a_{72}^{(2)} \\ 0 & 0 & a_{81}^{(3)} & 0 & a_{11}^{(3)} & a_{21}^{(3)} \\ 0 & 0 & 0 & a_{82}^{(3)} & a_{12}^{(3)} & a_{22}^{(3)} \end{array} \right]$						
OLD						NEW									

Figure 4.1: Change in unknowns ordering.

element of the blocks is a 2x2 matrix, denoting the two-group neutron energy discretization.

The new banded structure of the $\overline{M}$ matrix suggests the utilization of one of the LU factorization techniques as the preconditioner for the BICGSTAB solver. The preconditioner implemented in FORMOSA-B is a block incomplete LU factorization preconditioner similar to the one implemented in the NESTLE code. The derivation for the BILU3D preconditioner implemented in NESTLE is available in [22]. An overview of the idea behind this preconditioner is presented in Section 3.1.

One advantage of solving both energy groups simultaneously, as in the BICGSTAB solver, is a reduction in the number of outer iterations which may reduce the overall execution time. Table 4.1 illustrates this phenomena. A Block R/B LSOR solver, where now the two groups are updated simultaneously, was developed to ascertain that this phenomena is not an artificial effect from the BICGSTAB solver. Table 4.1 also confirms that the Block R/B LSOR and BICGSTAB solvers need the same number of

Table 4.1: R/B LSOR versus BICGSTAB Solvers.

		Group-wise Solution Procedure		All Groups Solved for Simultaneously	
		R/B LSOR		Block R/B LSOR	BICGSTAB
Outer Iteration		85		43	43
Inner Iteration	Fast	836	423	465	43
	Thermal		413		
Inner/Outer	Fast	9.835	4.976	10.814	1.000
	Thermal		4.859		
Solution time (sec ¹)		5.73		6.63	3.53

1: Timing was performed on a 440 MHz SUN Ultra-10 machine.

outer iteration to reach convergence when using appropriate inner iteration stopping criteria.

4.5. Results

We have performed parametric studies to determine the optimum stopping criteria for the preconditioned BICGSTAB solver for the inner iteration, the frequency of thermal hydraulics (T/H) and NEM updates, and the Wielandt shift parameter such that the overall CPU time is minimized. When performing a calculation without a critical flow search, only one BICGSTAB inner iteration is allowed. Two outer iterations are performed for each T/H update and three T/H updates are done for each NEM update. The optimum Wielandt shift parameter is 0.065 and is different from the value for the traditional R/B LSOR solver (0.050). We arrived at this set of parameters after performing parametric studies on load follow depletions using several core types. The speedups observed using the BICGSTAB method as the inner iteration solver for completing the load follow calculations are summarized in Table 4.2. Overall speedups of ~ 1.5 to ~ 2 are observed.

Table 4.2: BICGSTAB Performance for Load Follow Calculations.

Core Type	Number of Depletion Steps	BICGSTAB CPU Time (sec ¹)	Speedup ²
368 assembly BWR/4	138	103.8	1.536
560 assembly BWR/4	74	84.2	1.553
724 assembly BWR/3 unit 2	27	41.2	1.468
724 assembly BWR/3 unit 3 cycle N	55	83.6	1.651
724 assembly BWR/3 unit 3 cycle N+1	83	129.3	1.583
724 assembly BWR/3 unit 3 cycle N+2	71	114.8	1.589
800 assembly BWR/6	71	110.2	1.959

1: Timing was performed on a 2 GHz DELL 530 workstation.

2: Speedup = [R/B LSOR CPU time]/[BICGSTAB CPU time]

The overall stopping criteria utilized for both the R/B LSOR and BICGSTAB solvers are $1e-4$, $5e-4$, and $5e-2$ for ϵ_k , ϵ_F , and ϵ_p , respectively, as defined in Eqs. (2-46) through (2-48).

When performing a calculation with the critical flow search option on, a different set of parameters is utilized. For the BICGSTAB inner iteration stopping criteria, we require the reduction in the L_2 -norm of the residual to be less than 0.5 or

$$\frac{\|\bar{r}^{(i)}\|_2}{\|\bar{r}^{(0)}\|_2} < 5 \times 10^{-1}, \quad (4-9)$$

where i is the current BICGSTAB iteration count and, suppressing the outer iteration count and the dependencies of the coefficient matrix on the thermal hydraulics parameters, the residual, $\bar{r}$, can be written as:

$$\bar{r} = \left(\bar{\bar{M}} - \frac{1}{k_s} \bar{\bar{F}} \right) \bar{\phi}^{(i)} - \left(\frac{1}{k} - \frac{1}{k_s} \right) \bar{\bar{F}} \bar{\phi}^{(0)}, \quad (4-10)$$

where k_s denotes the Wielandt shift. Four outer iterations are done before the T/H update is performed and three T/H updates are completed for each NEM update. The critical flow search update is performed right after a T/H update. The summary of speedups observed using the BICGSTAB inner iteration solver in critical flow search calculations for several core types is shown in Table 4.3. The same set of stopping criteria as before is utilized for terminating the overall solver. For the critical flow search, the k_{eff} window is set to 25 pcm.

Observing Table 4.2 and Table 4.3, one can see that the speedups for the 800 assembly core are significantly higher than the speedups for the other core types. We believe that this can be related to the fact that when solving the problem using the traditional solver, this core type requires a slightly higher number of iterations for both fast and thermal inner iterations compared to the other core types. This core type requires, on average, 5 fast and 5 thermal iterations during inner iteration calculations while the other core types require, on average, 4 and 3 iterations for the fast and thermal groups,

Table 4.3: BICGSTAB Performance for Critical Flow Search Calculations.

Core Type	Number of Depletion Steps	BICGSTAB CPU time (sec ¹)	Speedup ²
368 assembly BWR/4	12	14.3	1.36
560 assembly BWR/4	19	36.4	1.48
724 assembly BWR/3	14	39.7	1.29
800 assembly BWR/6	18	28.9	2.20

1: Timing was performed on a 2 GHz DELL 530 workstation.

2: Speedup = [R/B LSOR CPU time]/[BICGSTAB CPU time]

respectively. Assuming the same number of outer iterations for all core types, the increase in the number of inner iterations alone can explain the difference in the speedup observed.

A typical FORMOSA-B run utilizes a quarter core geometry model, as done in the previous examples, to reduce the overall CPU time. However, we still need to test the BICGSTAB solver performance for completing calculations using a full core model. A problem which requires a full core calculation is the cold shutdown margin calculation (CSDM). For a full core CSDM calculation, the BICGSTAB iterations are performed until the following stopping criteria is satisfied:

$$\frac{\|\bar{r}^{(i)}\|_2}{\|\bar{r}^{(0)}\|_2} < 1 \times 10^{-1}, \quad (4-11)$$

where $\bar{r}$ is defined in Eq. (4-10). A CSDM calculation at the beginning of cycle (BOC) using a 368 assembly BWR/4 core was examined. The number of control rods on which an N-1 SDM calculation is requested is set to 20. The speedup observed without CSDM calculation is 1.64 while the speedup observed with CSDM calculation is 1.57.

A combined LP-CRP optimization with critical flow search and CSDM constraint, similar to the problem reported in [25], was performed using both R/B LSOR and

Table 4.4: FORMOSA-B LP-CRP Optimization Results for SDM Constraint for a 368 assembly GE BWR/4 core.

	Constraint	Reference LP	R/B LSOR	BICGSTAB
EOC ARO flow	<1.05	1.068	1.050	1.046
MFLPD	<0.99	1.056	0.978	0.969
MAPRAT	<0.99	1.100	0.974	0.989
MFLCPR	<0.99	1.025	0.982	0.974
SDM	>0.0120	0.0058	0.0119	0.0119
LP Examined			4130	4055
CPU time (hours ^{1,2})			27.08	14.23
Speedup			1.90	

1: Timing was performed on a 2 GHz DELL 530 workstation.

2: Early rejection option is turned on.

BICGSTAB solvers. The number of LPs sampled during the optimization is 4000 LPs and the same overall stopping criteria as before is utilized. Note that during the optimization search, a quick approximate CSDM calculation which does not involve a full core calculation is completed. The summary of the optimization results is shown in Table 4.4. The BICGSTAB solver approximately cuts the CPU time in half. The resulting optimized LP-CRP obtained using the R/B LSOR versus BICGSTAB solver is different as expected. However, the quality of the optimized LP-CRP is comparable.

A CRP optimization only was also performed using both R/B LSOR and BICGSTAB as the inner iteration solver. An 800 assembly BWR/6 core is utilized. Again, the same overall stopping criteria is employed. The CPU time for the CRP sampling using R/B LSOR is 100.0 minutes while the CPU time using the BICGSTAB solver is 59.8 minutes (speedup of 1.672). The optimum CRPs obtained using R/B LSOR and

Table 4.5: FORMOSA-B CRP Sampling Results for an 800 Assembly BWR/6 Core.

	Constraint	Reference LP (ARO)	R/B LSOR	BICGSTAB
EOC ARO flow	<1.05	<0.6	0.929	0.923
MFLPD	<0.94	1.28	0.937	0.934
MAPRAT	<0.94	1.24	0.863	0.869
MFLCPR	<0.94	1.29	0.924	0.917
CPU time (hours ¹)			1.67	1.00
Speedup			1.67	

1: Timing was performed on a 2 GHz DELL 530 workstation.

BICGSTAB solvers are slightly different (see Figures 4.2 and 4.3); hence, the quality of the optimum CRPs also differs very slightly (see Table 4.5).

We conclude that the implementation of BICGSTAB as the inner iteration solver in the FORMOSA-B code has been shown to be robust and computationally more efficient than the R/B LSOR solver previously utilized by the FORMOSA-B code. Depending on the problem examined, the overall speedup provided by the new solver can range from 1.3 to 2.2.

Depletion step 1	Depletion step 2	Depletion step 3
14 -- 24 -- 34 -- -- --	12 -- 24 -- 34 -- -- --	8 -- 24 -- 34 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
24 -- 14 -- 24 -- 38 --	24 -- 14 -- 24 -- 44 --	24 -- 14 -- 24 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
34 -- 24 -- 38 -- -- --	34 -- 24 -- 40 -- -- --	34 -- 24 -- 42 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- 38 -- -- -- --	-- -- 44 -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Depletion step 4	Depletion step 5	Depletion step 6
8 -- 24 -- 38 -- -- --	-- 12 -- 38 -- 14 -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	10 -- 30 -- 16 -- -- --
24 -- 10 -- 24 -- -- --	-- 36 -- 14 -- 36 -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	38 -- 8 -- 38 -- -- --
38 -- 24 -- -- -- --	-- 14 -- 36 -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	16 -- 38 -- 34 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Depletion step 7	Depletion step 8	Depletion step 9
-- -- 8 -- 12 -- 30 -- --	-- 8 -- 12 -- 34 -- --	-- -- 8 -- 12 -- 34 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 12 -- 36 -- 20 -- --	-- 12 -- 16 -- 34 -- --	-- 12 -- 12 -- 36 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 32 -- -- -- -- --	-- 34 -- -- -- -- --	-- 34 -- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Depletion step 10	Depletion step 11	Depletion step 12
16 -- 22 -- 12 -- -- --	-- 12 -- 14 -- 12 -- --	-- 12 -- 16 -- 12 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
22 -- 12 -- 12 -- 40 --	-- 12 -- 12 -- 22 -- --	-- 12 -- 12 -- 22 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 12 -- 38 -- -- --	-- 12 -- 22 -- -- --	-- 12 -- 22 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- 40 -- -- -- --	-- 44 -- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Depletion step 13	Depletion step 14	Depletion step 15
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
14 -- 12 -- 12 -- -- --	14 -- 12 -- 22 -- -- --	-- 12 -- 12 -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 12 -- 42 -- -- --	12 -- 12 -- -- -- --	-- 12 -- -- -- 18 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 42 -- -- -- --	22 -- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
Depletion step 16	Depletion step 17	Depletion step 18
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 12 -- 12 -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- 28 -- -- -- --	-- -- -- -- --
-- 14 -- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --

Figure 4.2: Optimum CRPs obtained using R/B LSOR inner iteration solver for an 800 assembly BWR/6 core.

Depletion step 1	Depletion step 2	Depletion step 3
18 -- 26 -- 36 -- -- --	12 -- 26 -- 36 -- -- --	12 -- 26 -- 36 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
26 -- 12 -- 24 -- 38 --	26 -- 14 -- 24 -- 42 --	26 -- 12 -- 24 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
36 -- 24 -- 38 -- --	36 -- 24 -- 40 -- --	36 -- 24 -- 42 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 38 -- -- --	-- 42 -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
Depletion step 4	Depletion step 5	Depletion step 6
8 -- 26 -- 38 -- -- --	-- 14 -- 38 -- 14 -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	10 -- 30 -- 16 -- -- --
26 -- 8 -- 24 -- -- --	-- 36 -- 14 -- 36 -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	36 -- 8 -- 38 -- -- --
38 -- 24 -- -- -- --	-- 14 -- 36 -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	16 -- 38 -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
Depletion step 7	Depletion step 8	Depletion step 9
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 8 -- 12 -- 30 -- --	-- 8 -- 12 -- 34 -- --	-- 8 -- 12 -- 34 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 12 -- 36 -- 20 -- --	-- 12 -- 16 -- 34 -- --	-- 12 -- 14 -- 38 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 32 -- -- -- -- --	-- 34 -- -- -- -- --	-- 34 -- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
Depletion step 10	Depletion step 11	Depletion step 12
12 -- 24 -- 12 -- -- --	-- 12 -- 12 -- 12 -- --	-- 12 -- 12 -- 12 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
24 -- 12 -- 12 -- 40 --	-- 12 -- 12 -- 24 -- --	-- 12 -- 12 -- 24 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 12 -- 38 -- --	-- 12 -- 24 -- -- --	-- 12 -- 24 -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 40 -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
Depletion step 13	Depletion step 14	Depletion step 15
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
16 -- 12 -- 12 -- -- --	16 -- 12 -- 22 -- -- --	-- 12 -- 12 -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 12 -- 42 -- -- --	12 -- 12 -- -- -- --	-- 12 -- -- -- 18 -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
12 -- 42 -- -- -- --	22 -- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --
Depletion step 16	Depletion step 17	Depletion step 18
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 12 -- 12 -- -- -- --	-- -- 26 -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- 14 -- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- -- --	-- -- -- -- --	-- -- -- -- --
-- -- -- --	-- -- -- --	-- -- -- --

Figure 4.3: Optimum CRPs obtained using BICGSTAB inner iteration solver for an 800 assembly BWR/6 core.

5. Results from the Implementation of the Newton-BICGSTAB Solver.

Presented in this section are the results from evaluating the Newton-BICGSTAB solver utilizing a single loading pattern (LP) as well as multiple LPs. The results from a control rod pattern optimization run are also included to show the performance of the solver for an optimization run. We use this approach to test the robustness of the solver against an increasing number of problems starting from analyzing a single configuration, *e.g.* when performing preconditioner studies, to evaluating numerous core configurations during an optimization run. We concluded this section with a brief discussion on the performance of inexact Newton's method.

5.1. Evaluation of a Single Loading Pattern

After deciding on utilizing the preconditioned BICGSTAB (with Preconditioner 3) as the solver for the linearized system of equations, we tested the solver on a single loading pattern to observe the average behavior of the solver over several burnup steps. This exercise is also intended to find a set of stopping criteria which provides the highest speedup while still maintaining the robustness of the solver.

The model used for this study is an 800 assembly GE BWR/6 core. The cycle modeled has 17 burnup steps. The nodal method is employed to solve the three-dimensional, two-group neutron diffusion equation. The three-equation mixture drift flux model is used to treat the two-phase flow in the core. An inlet flow redistribution is also performed to obtain uniform inlet and exit pressures across the core. In addition, the critical flow search is also executed.

The test is actually divided into two parts. In the first part, the true errors in the neutron flux distribution and the moderator density distributions are used to terminate the overall Newton-BICGSTAB solver. The true error for the neutron flux distribution is defined as follow:

$$\epsilon_{true}^{\phi} = \left\| \frac{\bar{\phi}^{(m)} - \bar{\phi}^{(ref)}}{\bar{\phi}^{(ref)}} \right\|_2 = \sqrt{\frac{1}{2L} \sum_{l=1}^L \left\{ \left[\frac{\bar{\phi}_{1,l}^{(m)} - \bar{\phi}_{1,l}^{(ref)}}{\bar{\phi}_{1,l}^{(ref)}} \right]^2 + \left[\frac{\bar{\phi}_{2,l}^{(m)} - \bar{\phi}_{2,l}^{(ref)}}{\bar{\phi}_{2,l}^{(ref)}} \right]^2 \right\}}, \quad (5-1)$$

where L denotes the total number of spatial fuel nodes and m represents the current Newton iteration count. Similarly, the true error for the moderator density distribution is defined as

$$\epsilon_{true}^{\rho} = \left\| \frac{\bar{\rho}^{(m)} - \bar{\rho}^{(ref)}}{\bar{\rho}^{(ref)}} \right\|_2 = \sqrt{\frac{1}{L} \sum_{l=1}^L \left[\frac{\rho_l^{(m)} - \rho_l^{(ref)}}{\rho_l^{(ref)}} \right]^2}. \quad (5-2)$$

The reference quantities are obtained using the traditional iterative algorithm with a set of extremely tight stopping criteria. The results from this part of the test are presented in Table 5.1. Speedup in Table 5.1 denotes the ratio of the CPU time for the traditional solver to Newton-BICGSTAB solver, both utilizing the same true error stopping criteria. We have chosen the true stopping criteria to be 1×10^{-2} for both the neutron flux and moderator density distributions. This implies that the results from these solvers are, on average, within 1% of the true solutions. There are two things to point out from this table. Firstly, the number of outer iterations are reduced by a factor of 12 as a result of the Newton-BICGSTAB implementation. This result numerically confirm the assertion that the Newton's method will provide a higher convergence rate compared to the traditional approach. Secondly, this test shows that the Newton-BICGSTAB solver can provide a speedup of 1.67 with respect to the traditional solver. However, in practice the true solutions are not available and, hence, the measure of true error cannot be utilized as the stopping criteria. Therefore, we still need to evaluate the performance of the solver when the real stopping criteria are utilized.

The second part of the test is aimed at answering this question. The stopping criteria utilized for the overall solver are $1e-4$, $5e-4$, and $5e-2$ for ϵ_k , ϵ_F , and ϵ_ρ (as defined in Eqs. (2-46) through (2-48)), respectively. The results from this part of the test are summarized in Table 5.2 and Figure 5.1. Speedup in Table 5.2 again denotes the ratio

Table 5.1: The Newton-BICGSTAB Performance on a Single LP Using True Error Stopping Criteria [Utilizing an 800 Assembly BWR/6 Core].

	Cases	
	Case 1	Case 2
Numerical Information		
Newton S.C.	1.00e-2	5.00e-3
Krylov S.C.	1.00e-2	5.00e-3
ϵ_{true}^{ϕ}	1.00e-2	1.00e-2
ϵ_{true}^{ρ}	1.00e-2	1.00e-2
Newton/BU Step	11.53	12.94
Newton Iterations	196	220
Traditional Outer Iterations	2417	
Speedup	1.67	1.55
Results Comparison (average differences)		
k_{eff} (in pcm)	14.84	12.77
Flow Fraction	0.0041	0.0032
MFLPD	5.8235e-4	5.7059e-4
MAPRAT	6.8824e-4	6.2355e-4
MFLCPR	3.9882e-3	3.1765e-3

of the CPU time for the traditional solver to Newton-BICGSTAB solver, both utilizing the same overall stopping criteria. The purpose of examining multiple pairs of Newton and Krylov stopping criteria is three-fold. First, we would like to observe the variation in the number of Newton iterations as a function of stopping criteria. Figure 5.1 shows that for a

Table 5.2: The Newton-BICGSTAB Performance on a Single LP [Utilizing an 800 Assembly BWR/6 Core].

	Cases						
	Case1	Case2	Case3	Case4	Case5	Case6	Case7
Numerical Information							
Newton S.C.	1.00e-2	1.00e-3	5.00e-4	1.00e-2	1.00e-3	5.00e-4	Var ⁽¹⁾
Krylov S.C	1.00e-2	1.00e-2	1.00e-2	5.00e-3	5.00e-3	5.00e-3	Var ⁽²⁾
Newton/BU Step	10.88	14.24	16.18	10.82	14.18	16.18	9.71
Newton Iterations	185	242	275	184	241	275	165
Traditional Outer Iterations	2377						
Reduction in Outer Iteration ⁽³⁾	12.85	9.82	8.64	12.92	9.86	8.64	14.41
Speedup	1.59	1.29	1.17	1.59	1.31	1.18	1.76
Results Comparison (average differences)							
k_{eff} (in pcm)	15.91	16.39	17.31	15.49	18.14	17.17	15.46
Flow fraction	0.0044	0.0042	0.0044	0.0041	0.0045	0.0043	0.0042
MFLPD	6.00e-4	6.29e-4	6.65e-4	6.29e-4	7.06e-4	6.64e-4	5.71e-4
MAPRAT	7.82e-4	8.71e-4	8.94e-4	8.35e-4	9.29e-4	8.94e-4	7.94e-4
MFLCPR	4.25e-3	4.17e-3	4.35e-3	4.08e-3	4.52e-3	4.33e-3	4.01e-3

- (1): Variable stopping criteria: 1E-1 for Newton iteration count less than 6
1E-2 for Newton iteration count between 6 and 10
1E-3 for Newton iteration count above 10
- (2): Variable stopping criteria: 1E-1 for Newton iteration count less than 4
1E-2 for Newton iteration count greater than and equal to 4
- (3): Reduction in Outer iteration is defined as [# traditional outer iteration]/[# Newton iteration]

particular set of stopping criteria, the number of Newton iterations does not change much throughout the depletion steps. Furthermore, as expected the total number of Newton iterations increases as the stopping criteria is reduced. Second, we would like to understand the trade-off between how much we gain in terms of improved agreement in

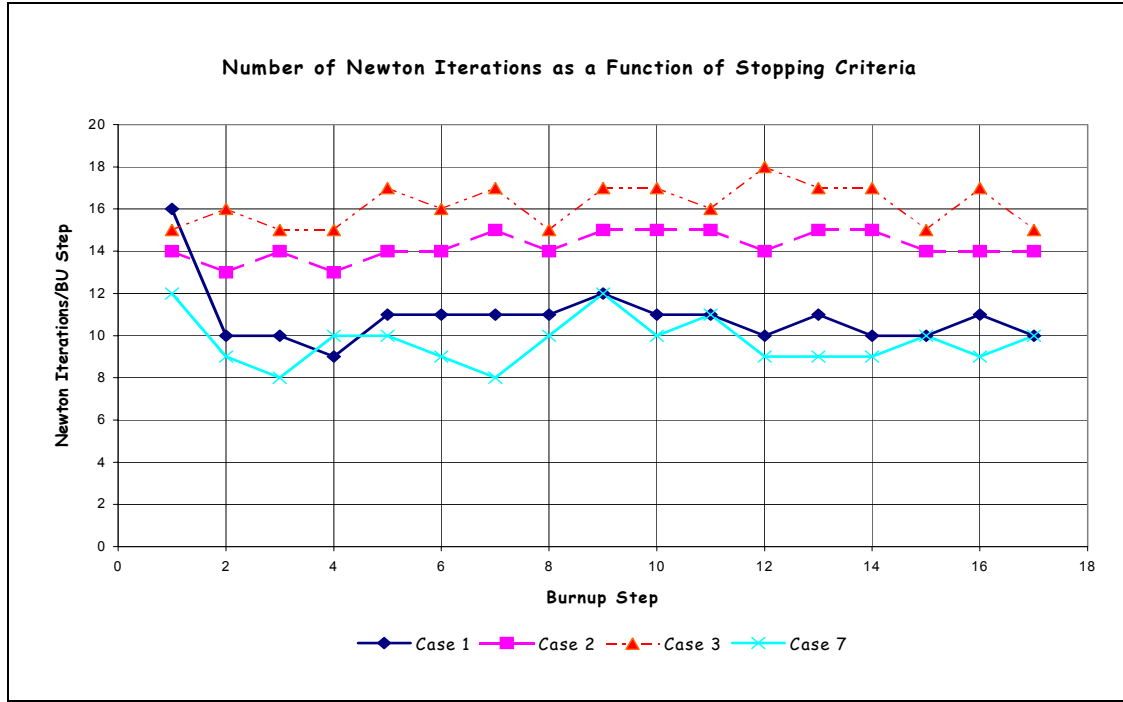


Figure 5.1: Newton iterations versus stopping criteria.

the results against how much we lose in terms of potential speedup. Results in Table 5.2 indicate that while there is no apparent improvement in the quality of the results as tighter stopping criteria are employed, significant decreases in the speedup factor are observed. These results suggest that a set of relatively relaxed stopping criteria should be employed by the solver. Lastly, we would like to examine the possibility of employing a set of dynamic stopping criteria, *i.e.* changing stopping criteria as the iteration progresses. This is what Case 7 in Table 5.2 represents. The philosophy used in dynamically changing the stopping criteria is that a more relaxed stopping criteria can be employed early in the iteration, while a tighter stopping criteria is needed when we are close to a converged solution. For example, employing a relaxed BICGSTAB stopping criteria in the first couple of Newton iterations will surely introduce error contaminations to the Newton iteration. However, at this stage since we are still far away from the solution, we believe that these contaminations are much less significant than the error introduced by the first-order approximation of the Newton's method. Later on when we are close to the solution,

since the Jacobian matrix is now “pointing” in the right direction and we would like to reach the solution as fast as possible, ideally in one iteration, we have to solve for the changes in the variables as accurately as we can; therefore, a tighter BICGSTAB stopping criteria is employed. The results shown in the last column of Table 5.2 indicates that this approach can produce a higher speedup than the statically determined set of stopping criteria.

5.2. Evaluation of Multiple Loading Patterns

The robustness of the solver utilizing the chosen set of stopping criteria from the single LP study, *i.e.* the dynamically determined stopping criteria, is examined further by testing the solver using multiple LPs. Two core types are evaluated during this study. The first core is an 800 assembly GE BWR/6 core. The cycle modeled has 17 burnup steps. The two-phase thermal hydraulics model with inlet flow redistribution is employed. In addition, the critical flow search is also performed. The second core is a 368 assembly GE BWR/4 core. The cycle has 10 depletion steps. The same thermal hydraulics model as the first core type is employed; however, no critical flow search is completed. For both core types, the nodal method is applied for solving the three-dimensional, two-group neutron diffusion equation.

The results from the multiple loading patterns evaluation are presented in Figures 5.2 and 5.3 for the first and second core types, respectively. The top part of Figure 5.2 depicts the agreement in the end of cycle (EOC) flow fraction between the traditional and the Newton-BICGSTAB solvers for the first core type. The stopping criteria for the critical flow search, *i.e.* the maximum difference allowed between the predicted and target k_{eff} value, is set to 25 pcm for this problem. Therefore, the difference between the two converged k_{eff} values can be as large as 50 pcm, which clearly can deteriorate the agreement in the EOC flow fraction as well. The top part of Figure 5.3 presents the agreement in the k_{eff} value between the two solvers for the second core type. The bottom part of these two figures show the agreement in the core limiting thermal

margins, *e.g.* MFLPD, MAPRAT, and MFLCPR. Based on these plots, one can see that the agreement between the two solvers is excellent. The average speedups for the 25 LPs evaluated are 1.708 and 1.697 for the 800 and 368 assembly cores, respectively, consistent with the result from the single LP evaluation. The speedups using BICGSTAB solver

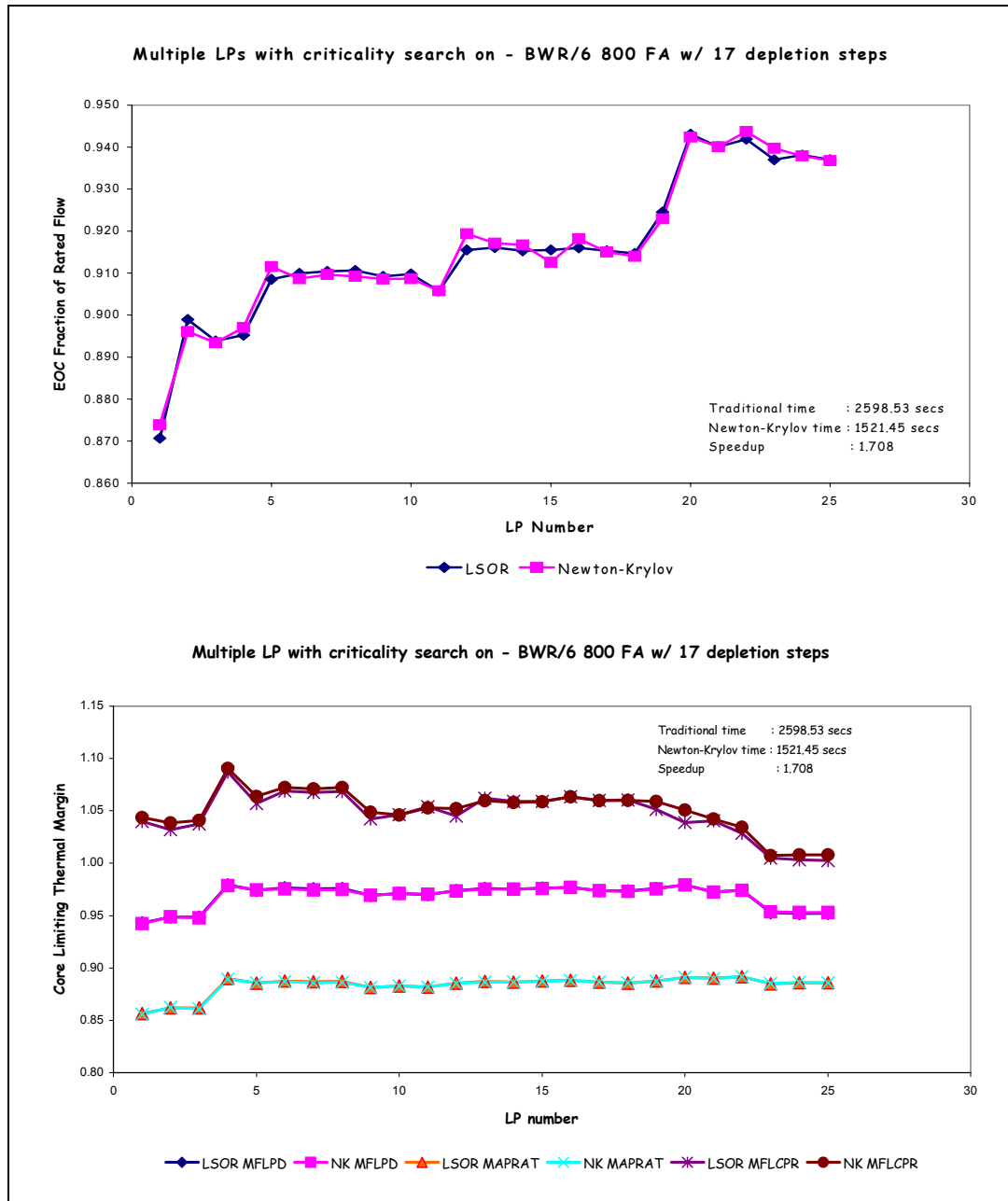


Figure 5.2: Multiple LPs results - 800 F/A GE BWR/6 core.

alone are 1.375 and 1.309 for the 800 and 368 assembly cores, respectively. Therefore, based on these results one can see that the Newton-BICGSTAB solver can provide additional 24%-30% increase in the overall speedup over the BICGSTAB solver alone.

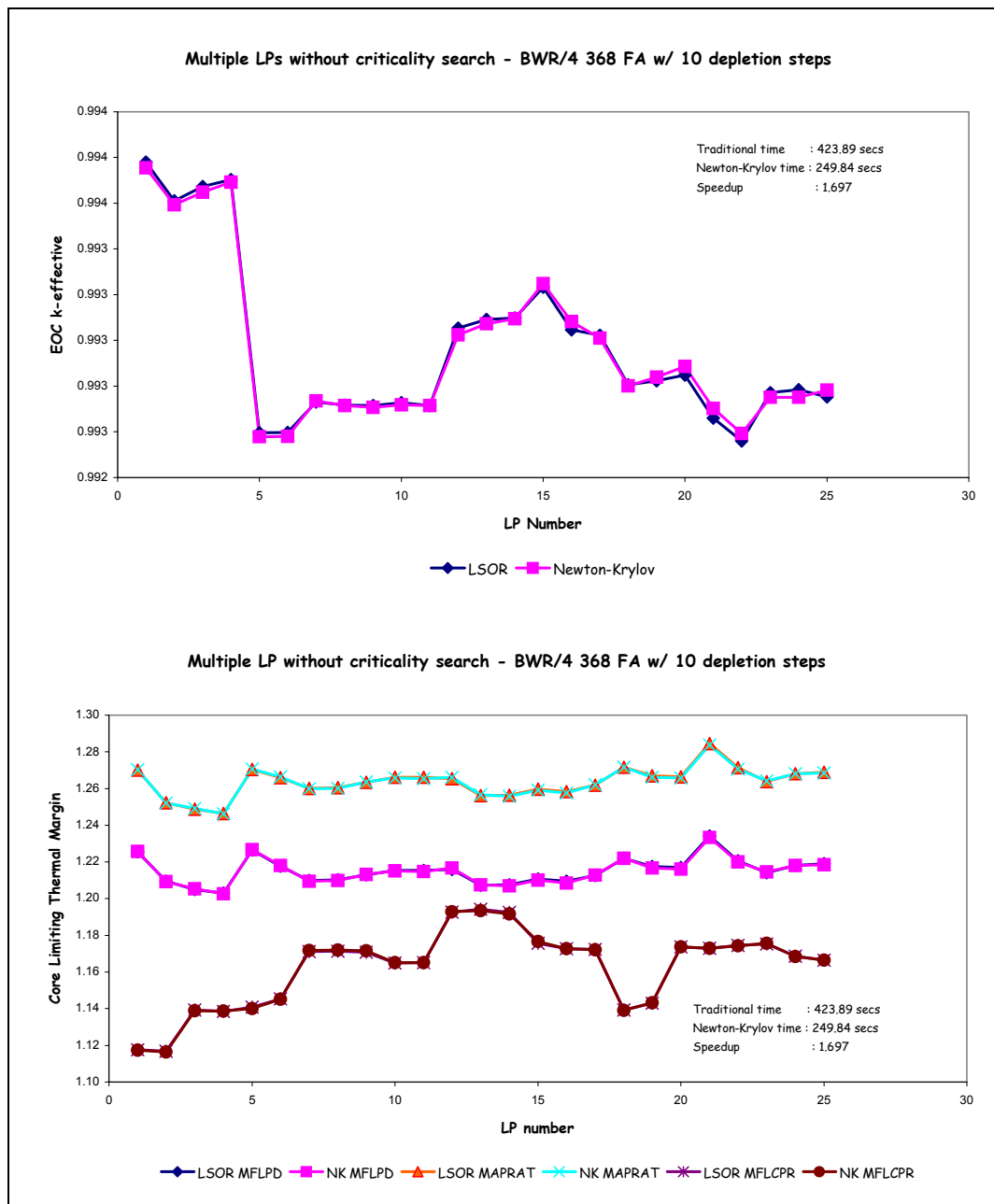


Figure 5.3: Multiple LPs results - 368 F/A GE BWR/4 core.

The speedups observed for the BICGSTAB solver as the inner iteration solver are lower than the average behavior observed earlier. The author believes that this anomalous behavior is due to the fact that only 25 LPs were evaluated. Therefore, to remove some statistical uncertainties from these results, we decided to examine the solver utilizing more LPs. We tested the Newton-BICGSTAB solver against 200 LPs of the 800 assembly core model and observed a speedup of 2.074 to the core simulator, with reference to the traditional approach. When solving the same problem using the BICGSTAB solver alone, as the inner iteration solver, we observed a speedup of 1.848. This means that the Newton-BICGSTAB solver provides an additional 12% increase in the overall speedup over the BICGSTAB solver alone.

5.3. Results from a Control Rod Pattern Optimization Run

Since the principal objective of this project is to reduce the CPU execution time of an optimization run, it is appropriate to use an optimization run as the ultimate test for both measuring speedup and assuring robustness of the Newton-BICGSTAB solver. The core model used for the optimization is an 800 assembly GE BWR/6 core. For the thermal hydraulics model, in addition to the model used in the previous studies, the spacer-void model which corrects the void fraction for the effects of spacer grids is also incorporated. The cycle analyzed has 10 depletion steps.

The results from this study are summarized in Table 5.3. Both optimization runs satisfy the thermal margin constraints imposed and produce similar quality results. With regard to the overall execution time, the Newton-BICGSTAB solver proves a speedup of 1.896. However, after taking into account that a different number of CRPs are evaluated in each run the average speedup gained by the new solver is around 1.61, which is still consistent with what was observed in the previous studies.

Table 5.3: CRP Optimization Results Using an 800 Assembly BWR/6 Core.

Variable	Constraint	Reference Pattern	Optimized Pattern	
			LSOR	Newton-BICGSTAB
MFLPD	<0.940	0.9540	0.9417	0.9396
MAPRAT	<0.940	0.8598	0.8848	0.8733
MFLCPR	<0.940	0.9456	0.9298	0.9235
Number of CRPs evaluated			209	178
Time (sec ¹)			1261	665

1: Timing was performed on a 2 GHz DELL 530 workstation.

5.4. Results from the Inexact Newton's Method Studies

Some studies on the feasibility of employing an inexact Newton's method to provide an additional speedup to the Newton-BICGSTAB solver have been initiated. The objective of this preliminary study is two-fold. Firstly, we would like to show that a proper utilization of the inexact Newton's method can indeed provide additional speedup to the exact Newton's method. Secondly, we would like to show some promising results from this preliminary study to motivate further research in this area. The results associated with a single LP evaluation presented in this section were obtained using the same core model employed in the single LP evaluation discussed in Section 5.1. The same core model, with the same thermal hydraulics option, is also utilized for the multiple LPs evaluation. However, note that the 25 LPs evaluated here are different from the ones evaluated in Section 5.2. This variation is introduced in order to examine the solver against a different set of problems. Each of the three sub-sections will discuss a unique approach for introducing the inexactness to the exact Newton's method.

5.4.1. Chord and Shamanskii's Methods

The performance of the Chord and Shamanskii's methods introduced earlier in Section 2.7 are evaluated. The results from this study are presented in Table 5.4 for a single LP depletion case using 17 burnup steps. From this table, one can see that initially a less frequent update of the Jacobian matrix, *i.e.* higher n value in the Shamanskii[n] method, provides additional speedup. This follows since the CPU time savings associated with less frequent Jacobian matrix updates more than offsets the penalty of increasing number of Newton iterations. However, if the frequency of the Jacobian matrix update is relaxed further, speedup is degraded since the additional computational time due the increase in Newton iterations surpasses the CPU time savings associated with less frequent Jacobian matrix updates. This trade-off results in an optimum performance by the

Table 5.4: Inexact Newton's Method Results [Utilizing an 800 Assembly BWR/6 Core].

Method	Time ⁽¹⁾ (seconds)	Newton Iteration	BICGSTAB Iteration	Weak Feedback Iteration
Newton	62.88	166	302	154
Chord	63.49	194	385	173
Shamanskii[2] ⁽²⁾	59.19	170	327	153
Shamanskii[3]	58.14	173	327	156
Shamanskii[4]	57.41	174	324	158
Shamanskii[5]	56.41	173	318	160
Shamanskii[6]	56.84	174	338	156
Shamanskii[7]	57.78	177	348	159
Shamanskii[8]	60.58	185	358	164

(1) : Timing was performed on a 2GHz DELL 530 workstation.

(2) : The n in Shamanskii[n] denotes the Jacobian matrix update frequency.

Shamanskii[5]-BICGSTAB solver, which provides an additional 10% reduction in the execution time.

When implementing the Shamanskii's method, we expected that the number of Newton iterations will grow with respect to the exact Newton's method due to the inexact nature of the method, but should not exceed the number of Newton iterations associated with the Chord method. Table 5.4 exhibits this expected behavior, where the numbers of Newton iterations associated with the exact Newton's method and the Chord method provide the lower and upper limits for the Shamanskii's method.

The Shamanskii-BICGSTAB solver has been further examined utilizing multiple LPs (25 LPs) as well. The timing results from this study are presented in Table 5.5. From this table, one can see that the Shamanskii[5]-BICGSTAB solver provides an additional 5.7% increase in speedup over the Newton-BICGSTAB solver.

Table 5.5: Multiple LP Results from Testing the Shamanskii's Method [Utilizing an 800 Assembly BWR/6 Core].

Method	Time (sec ⁽¹⁾)	Speedup
Traditional	2208.59	N/A
Newton	1330.35	1.660
Shamanskii[4] ⁽²⁾	1280.34	1.725
Shamanskii[5]	1259.12	1.754
Shamanskii[6]	1273.06	1.734

(1) : Timing was performed on a 2 GHz DELL 530 workstation.

(2) : The n in Shamanskii[n] denotes the Jacobian matrix update frequency.

5.4.2. A Hybrid Chord-Shamanskii Method

The Chord and Shamanskii's methods are both developed such that the computational time associated with performing the Jacobian matrix update is minimized. Unfortunately, as indicated in Table 5.4, the computational time for the Chord method is comparable to that of the exact Newton's method, due to the significant increase in the number of Newton iterations which more than offsets the saving in the computational time of Jacobian matrix updates. We believe that the number of Newton iterations for the Chord method can be reduced if a better Jacobian matrix, in a sense that it "points" in a better direction toward the solution, is introduced during the Newton iterations. This is essentially what is done in the Shamanskii's method. However, for what we call "the hybrid Chord-Shamanskii method," the additional Jacobian matrix setup is only performed once at the n^{th} Newton iteration, resulting in a total of two Jacobian matrix updates for solving the problem.

Table 5.6 summarizes the results from this study for a single LP depletion case using 17 burnup steps. There are several things to note from this table. First, initially when the second Jacobian matrix update is not too far from the initial determination, the number of total Newton iteration decreases. However, it hits an optimum "distance" for the second Jacobian matrix update at the 6th Newton iteration and henceforth the number of Newton iterations begins to climb. This result shows that the implementation of a hybrid Chord-Shamanskii method can provide an additional 10% reduction in the execution time. However, these results suggest that we need to know the optimum point to perform the second update to minimize the overall execution time and, therefore, some parametric studies are needed to determine the optimum point for variety of problems before any conclusion can be made. Also there is a question of robustness, since performing the update just one iteration less than the optimum value, *i.e.* 5th iteration, results in the iteration failing (see footnote on Table 5.6).

Table 5.6: The Hybrid Chord-Shamanskii Method Performance [Utilizing an 800 Assembly BWR/6 Core].

Newton iteration where the second Jacobian matrix update is performed							
	1 ⁽¹⁾	2	3	6 ⁽²⁾	7	8	9
Numerical Information							
Newton Iteration	194	182	183	171	174	180	188
BICSTAB Iteration	385	333	335	316	336	354	366
NEM Iteration	173	166	167	158	157	159	166
Time (sec ^{(3),(4)})	60.17	53.77	53.83	52.20	53.00	54.17	55.47
Results Comparison (average differences)⁽⁵⁾							
k_{eff} (in pcm)	14.271	9.195	10.242	11.188 (-27.050)	12.521	12.428	10.447
Flow fraction	0.0047	0.0036	0.0038	0.0040 (-0.0096)	0.0043	0.0043	0.0042
MFLPD	8.55e-4	7.29e-4	7.43e-4	7.12e-4 (-2.00e-3)	8.88e-4	8.29e-4	8.29e-4
MAPRAT	9.17e-4	8.24e-4	8.19e-4	8.35e-4 (-2.1e-3)	1.02e-3	9.65e-4	9.53e-4
MFLCPR	3.96e-3	3.43e-3	3.52e-3	3.79e-3 (9.80e-3)	4.03e-3	4.03e-3	3.96e-3

(1) : This is essentially the Chord Method.

(2) : Updates at the 4th and 5th Newton iterations failed at depletion step 13 due to entering a non-physical region.

(3) : Timing was performed on a 2 GHz DELL 530 workstation.

(4) : The exact Newton-BICGSTAB execution time is 58.14 seconds. The traditional solver execution time is 104.23. Speedup is 1.793.

(5) : The number in parentheses for the shaded column represents the maximum difference for each category.

The second thing to observe from this table is the fact that the number of Newton iterations for the hybrid Chord-Shamanskii method is still bounded below by the number of Newton iteration associated with the exact Newton's method. Thirdly, the maximum difference in the k_{eff} value of ~ 27 pcm, for the optimum case, is still within the $2 \times k_{window}$ band for the critical flow search.

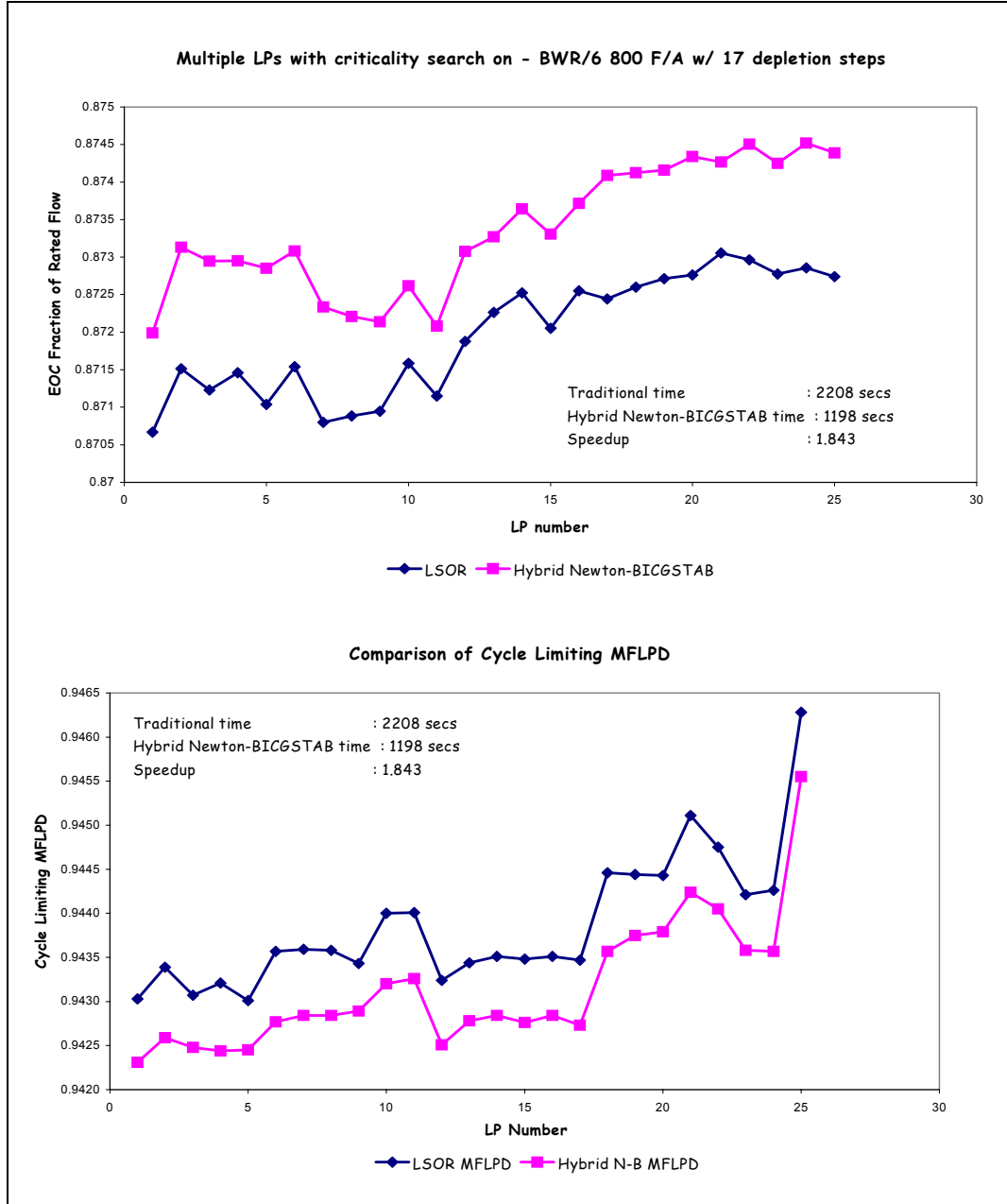


Figure 5.4: Multiple LP results from testing the hybrid Chord-Shamanskii Method.

We have also tested the optimum solver from this study, *i.e.* performing the second update at the 6th Newton iteration, utilizing the same 25 LPs as in Section 5.4.1. The results from this numerical testing are graphically presented in Figure 5.4. The top part of

this figure shows the comparison of the EOC fraction of rated flow. Both solvers show similar trends and the difference between the two curves is acceptable since it is well within the average difference reported in Table 5.6. The bottom part of the plot shows the agreement in the cycle limiting MFLPD values. Again the difference between the two curves is still within the average difference observed in the single LP evaluation. The average speedup for the 25 LPs evaluated is 1.843, an increase in speedup of 11% over what is obtained using the Newton-BICGSTAB method (speedup of 1.660, see Table 5.5).

5.4.3. An Incomplete Jacobian Matrix Approach

Observing Table 3.3, one can see that a significant portion of the CPU time is spent performing matrix-vector multiplications. One way to reduce the FLOP count for performing this task is to set to zero value the elements of the Jacobian matrix which are below a certain cut-off value. The advantage of doing this not only comes from the reduced number of matrix-vector multiplication operations but also comes from the Jacobian matrix updates portion of the solver. Since the zero value elements will not be subsequently used, there is no sense in calculating them during the Jacobian matrix updates introducing additional computational time savings.

We have performed a preliminary study where we set the cut-off value, to set Jacobian matrix elements to zero value, to 1×10^{-5} . This cut-off value is chosen since a larger value results in a divergent solver. In order to determine which elements whose values are below this cut-off value, we examined each element of the Jacobian matrix for two core types (800 and 368 assembly cores) over several burnup steps. The elements which consistently satisfy the test are subsequently set to zero permanently *via* modifying the source code. Alternatively, the test can be performed on the initial Jacobian matrix elements' values, *i.e.* all matrix elements are evaluated initially but henceforth matrix elements satisfying the test are set to zero. The disadvantage of this approach is that we have to “flag” the elements to be set to zero, which implies that we need an additional array for the flag, and we have to check for elements with “flags” each time we update the

Jacobian matrix. The check, which involves “if” statements inside the loop over all nodes, can become computationally costly.

The test was performed in conjunction with the optimum hybrid Chord-Shamanskii method from the previous section. The core examined is the same as the one described in Section 5.1 and one LP depletion case is studied. The speedup gained from this test is 2.104, which is a 19.5% increase in speedup over the Newton-BICGSTAB solver (speedup of 1.760). The average differences in k_{eff} value and EOC fraction of rated flow are 10.108 pcm and 0.0039, respectively. The average differences in thermal margin are 7.12e-4, 8.47e-4, and 3.80e-3 for MFLPD, MAPRAT, and MFLCPR, respectively.

The combination of hybrid Chord-Shamanskii method and the incomplete Jacobian matrix approach has been tested against multiple LPs (25 LPs) as well. The timing results from the evaluation are summarized in Table 5.7. This table shows that the result from the single LP evaluation still holds. However, the increase in the speedup over the Newton-BICGSTAB solver of ~14% is not as large as in the single LP case.

Table 5.8 shows the time allocation for performing several major functions in the Newton and Krylov parts of the solver. The test was performed on a single LP of the 800

Table 5.7: Results from Testing the Hybrid Chord-Shamanskii and Incomplete Jacobian Matrix Approach [Utilizing an 800 Assembly BWR/6 Core].

Method	Time (sec ⁽¹⁾)	Speedup
Traditional	2208.59	N/A
Newton	1330.35	1.660
Hybrid[6] ⁽²⁾	1165.72	1.895
Hybrid[7]	1310.50	1.685

(1) : Timing was performed on a 2 GHz DELL 530 workstation.

(2) : The n in Hybrid[n] denotes the Newton iteration where the second Jacobian matrix update is performed.

Table 5.8: Detail Time Allocation from Testing the Hybrid Chord-Shamanskii and Incomplete Jacobian Matrix Approach on a Single LP of 800 Assembly BWR/6 Core.

Variable	Time (sec ¹)	
	Exact Newton-BICGSTAB	Inexact Newton-BICGSTAB
Newton part of the solver		
Jacobian Matrix Setup	9.7032	2.6717
Jacobian Factorization	0.4219	0.0938
Variable Update	11.6092	11.9531
Setup FDM Matrix and RHS vector	1.3438	1.5470
Krylov part of the solver		
Preconditioner Solve	5.1096	6.0783
Matrix-Vector Operation	3.5780	2.4062
Inner Product	0.1563	0.2344
Vector Updates	0.2657	0.2501
Newton Iteration	163	169
Krylov Iteration	290	317

(1) : Timing was performed on a 2 GHz DELL 530 workstation.

assembly BWR/6 core model, with 17 burnup steps. There are three things to be pointed out from this table. Firstly, since there are only two Jacobian matrix updates performed for each burnup step, the computational time associated with completing this task is reduced significantly (by about 3.6 times). Secondly, since the Jacobian matrix is factorized every time it is updated, a less frequent Jacobian matrix update reduces the computational time for the factorization as well. Finally, since some of the Jacobian matrix elements are

eliminated (by setting them to zero values), the cost for matrix-vector multiplication is reduced as well.

6. Conclusions and Recommendations for Future Projects.

The main objective of this research, which is to reduce the CPU execution time of the FORMOSA-B code BWR core simulator, has been accomplished by virtue of the implementation of a robust and computationally efficient Newton-BICGSTAB solver. An average overall speedup of 1.7 has been observed for several set of problems. During this research, the best exact Newton-BICGSTAB solver performance shows an overall speedup of 2.07 to the core simulator, with reference to the traditional approach. This solver provides an additional 12% increase in the overall speedup over the traditional, *i.e.* outer (fission source)- inner (red/black line SOR) iterations, approach but now using BICGSTAB as the inner iterative solver [traditional (BICGSTAB)].

An investigation of the feasibility of implementing an inexact Newton-BICGSTAB solver has also been initiated in this research. Several approximations were examined in regard to introducing some inexactness to the solver. The robustness of several types of inexact Newton's method has been assessed by testing the solvers against multiple LPs. The results from the tests show that the inexact Newton's method can provide a speedup of 1.75 to 1.90, which is a 5.7% to 14.2% increase over the exact Newton-BICGSTAB solver. In particular, an inexact Newton-BICGSTAB solver, in which an approximation is made to the Jacobian coefficient matrix by setting to zero value some elements of the matrix which are less than a predetermined cut-off value in conjunction with utilizing a hybrid Chord-Shamanskii scheme to fix the frequency of the Jacobian matrix update, provides the highest speedup of 1.90.

A key to a successful implementation of the new solver is the development of a preconditioner for the BICGSTAB solver. The performance of the current Newton-BICGSTAB solver can likely be further improved by formulating a more sophisticated preconditioner. For example, a preconditioner which is suitable for parallel computations could substantially further reduce the computational time.

Inclusion of more non-linear variables, *e.g.* the weak non-linear feedbacks, in the Newton's method treatment is another topic worthy of examination. Associated with this work is the development of a new preconditioner since the structure of the Jacobian matrix changes as more variables are included.

The implementation of an appropriate inexact Newton's method is another way to improve the performance of the Newton-BICGSTAB solver. The results reported in Section 5 are encouraging and should motivate more research in this area. There are several topics related to the implementation of an inexact Newton-BICGSTAB solver that can be further investigated.

Firstly, there were several assumptions made when analyzing the results in Section 5.4.2 and Section 5.4.3. When evaluating the solver against multiple LPs in Section 5.4.2, it was assumed that the optimum time for performing the second Jacobian matrix update for a single LP holds for multiple LPs as well. One needs to actually perform a study to determine an optimum time for the Jacobian matrix update based on averaged performance of multiple LPs. Results from testing the solver against multiple LPs presented in Section 5.4.3 are obtained by making the same assumption. This also needs to be examined further to ascertain that the combination evaluated is truly the optimum value.

Secondly, keeping other parts of the exact Newton-BICGSTAB algorithm unchanged, we can elect to perform the LU factorization of the preconditioner matrix less frequently to reduce the execution time. This approach was not pursued during this research since the computational cost of the LU factorization is insignificant when compared to the overall CPU time. However, in conjunction to developing a new preconditioner this approximation might prove to be more beneficial.

Thirdly, observing Table 5.8, one can see that the CPU time for performing variable updates is a significant portion of the overall solver time. Therefore, more studies are needed to perform this task more efficiently.

Fourthly, instead of developing a new preconditioner for the Krylov solver, since the Jacobian matrix has a nice banded structure, one can elect to utilize an optimized preconditioned Krylov package. If this approach were to be pursued, one needs to be

careful in arranging the data structure of the variables within the code so that passing arguments to the “black-box” preconditioner can be done efficiently so as to prevent degradation in the overall performance.

Lastly, since in many preconditioned Krylov solvers the Jacobian matrix operation appears only in a form of matrix-vector multiplication, there are ways to approximate this by utilizing a finite difference approximation using the original non-linear system of equations. This approximation leads to a matrix-free implementation of the Newton’s method [38]. For the implementation of the exact Newton’s method, this approach can likely reduce the execution time since the computational time for setting up the Jacobian matrix is a significant portion of the overall execution time. For the inexact Newton’s method, such as Chord and Shamanskii’s methods, the reduction in the overall execution time might not be too significant since a fewer number of Jacobian matrix updates is needed.

7. References

- [1] O. Axelsson, "Incomplete Block Matrix Factorization Preconditioning Methods. The Ultimate Answer?," *J. Comp. App. Math.*, Vol. 12&13, pp. 3-18 (1985).
- [2] B. R. Bandini, *A Three-Dimensional Transient Neutronics Routine for the TRAC-PF1 Reactor Thermal Hydraulic Computer Code*, Ph.D. Dissertation, Pennsylvania State University (1990).
- [3] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM, Philadelphia, P.A. (1993).
- [4] F. Bennewitz, H. Finnmann, and H. Moldaschl, "Solution of the Multidimensional Neutron Diffusion Equation by Nodal Expansion," *Proc. Conf. Computational Methods in Nuclear Engineering*, p. 1-99, CONF-750413, Charleston, S.C., 15-17 April, American Nuclear Society (1975).
- [5] Michele Benzi, Carl D. Meyer, Miroslav Tuma, "A Sparse Approximate Inverse Preconditioner for the Conjugate Gradient Method," *SIAM J. Sci. Comput.*, Vol. 17, No. 5, pp. 1135-1149 (1996).
- [6] P. N. Brown and Y. Saad, "Convergence Theory of Nonlinear Newton-Krylov Algorithms," *SIAM J. Optimization*, Vol. 4, No. 2, pp. 297-330 (1994).
- [7] P. N. Brown and H. F. Walker, "GMRES on (Nearly) Singular Systems," *SIAM J. Matrix Anal. Appl.*, Vol. 18, No. 1, pp. 37-51 (1997).
- [8] T. F. Chan, E. Chow, Y. Saad, and M. C. Yeung, "Preserving Symmetry in Preconditioned Krylov Subspace Methods," *SIAM J. Sci. Comput.*, Vol. 20, No. 2, pp. 568-581 (1998).
- [9] T. F. Chan, S. D. Margenov, and P. S. Vassilevski, "Performance of Block-ILU Factorization Preconditioners Based on Block-Size Reduction of 2D Elasticity Systems," *SIAM J. Sci. Comput.*, Vol. 18, No. 5, pp. 1355-1366 (1997).
- [10] B. Chexal and G. Lellouche, *A Full Range Drift-Flux Correlation for Vertical Flows*, EPRI NP-3989-SR (1986).

- [11] P. Concus, G. H. Golub, and G. Meurant, "Block Preconditioning for the Conjugate Gradient Method," *SIAM J. Sci. Stat. Comput.*, Vol. 6, No. 1, pp. 220-252 (1985).
- [12] M. DeChaine and M. Feltus, "Fuel Management Optimization Using Genetic Algorithms and Expert Knowledge," *Nucl. Sci. Eng.*, **124**, 188-196 (1996).
- [13] M. DeChaine and M. Feltus, "Nuclear Fuel Management Optimization Using Genetic Algorithms," *Nucl. Technol.*, **111**, 109 (1995).
- [14] D. W. Decker and C. T. Kelley, "Broyden's Method for a Class of Problems Having Singular Jacobian at the Root," *SIAM J. Numer. Anal.*, Vol. 22, No. 3, pp. 566-574 (1985).
- [15] D. W. Decker and C. T. Kelley, "Convergence Acceleration for Newton's Method at Singular Points," *SIAM J. Numer. Anal.*, Vol. 19, No. 1, pp. 219-229 (1981).
- [16] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, "Inexact Newton Methods," *SIAM J. Numer. Anal.*, Vol. 19, No. 2, 400-408 (1982).
- [17] T. Downar, J. Y. Wu, and R. Janardhan, "Parallel and Serial Applications of the RETRAN-03 Power Plant Simulation Code Using Domain Decomposition and Krylov Subspace Methods," *Nucl. Technol.*, **117**, pp. 133-150 (1997).
- [18] H. Finnemann, F. Bennewitz, and M. R. Wagner, "Interface Current Techniques for Multidimensional Reactor Calculations," *Atomkernenergie*, **30**, 123 (1977).
- [19] R. W. Freund and N. M. Nachtigal, "QMR: A Quasi-Minimal Residual Method for Non-Hermitian Linear System," *Numer. Math. Journal*, Vol. 60 (1991).
- [20] D. Janaki Ram, T. H. Sreenivas, K. G. Subramaniam, "Parallel Simulated Annealing Algorithms," *Journal of Parallel and Distributed Computing*, **37**, 207-212 (1996).
- [21] R. Janardhan and T. Downar, "A Nested FGMRES Method for Parallel Calculation of Nuclear Reactor Transients," *Journal of Scientific Computing*, Vol. 13, No. 1, pp. 65-93 (1998).
- [22] H. G. Joo, *Domain Decomposition Methods for Nonlinear Nodal Spatial Kinetics*, Ph.D. Dissertation, Purdue University (1996).

- [23] H. G. Joo and T. Downar, "Incomplete Domain Decomposition Preconditioning for Coarse Mesh Neutron Diffusion Problems," *Nucl. Sci. Eng.*, **123**, pp. 403-414 (1996).
- [24] A. A. Karve and P. J. Turinsky, "FORMOSA-B: A Boiling Water Reactor In-Core Fuel Management Optimization Package II," *Nucl. Technol.*, **131**, 48-68 (2000).
- [25] A. A. Karve and P. J. Turinsky, "FORMOSA-B: A Boiling Water Reactor In-Core Fuel Management Optimization Package III," *Nucl. Technol.*, **135**, 241-251 (2001).
- [26] P. Keller and P. J. Turinsky, "FORMOSA-P Three Dimensional/Two Dimensional Geometry Collapse Methodology," *Nucl. Sci. Eng.*, **139**, pp. 235-247 (2001).
- [27] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, SIAM, Philadelphia (1995).
- [28] C. T. Kelley, *Iterative Methods for Optimization*, SIAM, Philadelphia (1999).
- [29] T. Kiguchi, *et al.*, "Optimization of Boiling Water Reactor Control Rod Patterns Using Linear Search," *Nucl. Technol.*, **67**, 38 (1984).
- [30] S. Kirkpatrick, C. D. Gellat, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, **220**, 671 (1983).
- [31] D. Kropaczek and P. J. Turinsky, "In-Core Nuclear Fuel Management Optimization for Pressurized Water Reactors Utilizing Simulated Annealing," *Nucl. Technol.*, **95**, 9 (1991).
- [32] D. Kropaczek, *In-Core Nuclear Fuel Management Optimization Utilizing Simulated Annealing*, Ph.D. Dissertation, N.C. State University (1990).
- [33] R. Lahey and F. Moody, *The Thermal Hydraulics of a Boiling Water Nuclear Reactor*, American Nuclear Society, La Grange Park, IL (1977).
- [34] R. D. Lawrence, "The DIF3D Nodal Neutronics Option for One-, Two-, and Three-Dimensional Diffusion Theory Calculations in Hexagonal Geometry," ANL-83-1, Argonne National Laboratory (1983).
- [35] G. Lellouche and B. Zolotar, *Mechanistic Model for Predicting Two-Phase Void Fraction in Vertical Tubes, Channels, and Rod Bundles*, EPRI NP-2246-SR (1982).

- [36] C. Lin, "An Automatic Control Rod Programming Method for a Boiling Water Reactor," *Nucl. Technol.*, **92**, 118 (1990).
- [37] G. Maldonado and P. J. Turinsky, "Application of Nonlinear Nodal Diffusion Generalized Perturbation Theory to Nuclear Reload Optimization," *Nucl. Technol.*, **110**, 198 (1995).
- [38] Paul R. McHugh and Dana A. Knoll, "Comparison of Standard and Matrix-Free Implementations of Several Newton-Krylov Solvers," *AIAA Journal*, Vol. 32, No. 12, 2394-2400 (1994).
- [39] N. Metropolis, *et al.*, "Equation of State Calculations by Fast Computing Machines," *J. Chemical Physics*, **21**, 1087 (1953).
- [40] B. R. Moore, P. J. Turinsky, and A. A. Karve, "FORMOSA-B: A Boiling Water Reactor Incore Fuel Management Optimization Package," *Nucl. Technol.*, **126**, 153 (1999).
- [41] B. R. Moore, *Higher Order Generalized Perturbation Theory for BWR In-Core Nuclear Fuel Management Optimization*, Ph.D. Dissertation, N.C. State University (1996).
- [42] P. W. Poon and G. T. Park, "Application of Genetic Algorithms to In-Core Nuclear Fuel Management Optimization," *Proc. Joint Int. Conf. on Mathematical Methods and Supercomputing in Nuclear Applications*, Karlsruhe, Germany, April 19-23, 1993, Vol. I, p. 777, Kernforschungszentrum Karlsruhe GmbH, Karlsruhe (1993).
- [43] T. A. Porsching, J. H. Murphy and J. A. Redfield, "Stable Numerical Integration of Conservation Equations for Hydraulic Networks," *Nucl. Sci. Eng.*, **43**, pp. 218-225 (1971).
- [44] D. Reddy, S. Sreepada, and A. Nahavandi, *Two-Phase Friction Multiplier Correction for High-Pressure Steam-Water Flow*, EPRI NP-2522 (1982).
- [45] Y. Saad and M. H. Schultz, "GMRES: A Generalized Minimal Residual Algorithm for Solving Non-Symmetric Linear Systems," *SIAM J. Sci. Stat. Comput.*, Vol. 7, No. 3, pp. 856-869 (1986).

- [46] K. S. Smith, *An Analytic Nodal Method for Solving the Two-Group, Multidimensional, Static and Transient Neutron Diffusion Equations*, M.S. Thesis, M.I.T. (1979).
- [47] K. S. Smith, "QPANDA: An Advanced Nodal Method for LWR Analyses," *Trans. Am. Nucl. Soc.*, **50**, 532 (1985).
- [48] J. So, R. Janardhan, T. Downar, and H. J. Siegel, "Mapping the Preconditioned Conjugate Gradient Algorithm for Neutron Diffusion Applications onto SIMD, MIMD, and Mixed-Mode Parallel Computers," *International Journal of Parallel Programming*, Vol. 26, No. 1, pp. 183-207 (1998).
- [49] T. M. Sutton, "NODEX: A High-Order NEM-Based Multigroup Nodal Code," *Advances in Nuclear Engineering Computation and Radiation Shielding*, Vol. 1, LaGrange Park, IL, American Nuclear Society (1989).
- [50] T. M. Sutton, "Wiendt Iteration as Applied to the Nodal Expansion Method," *Nucl. Sci. Eng.*, **98**, 169-173 (1988).
- [51] C. Tsallis, D. A. Stariolo, "Generalized Simulated Annealing," *Physica A*, **233**, 395-406 (1996).
- [52] P. J. Turinsky, *et al.*, "NESTLE: A Few-Group Neutron Diffusion Equation Solver Utilizing the Nodal expansion Method for Eigenvalue, Adjoint, Fixed-Source Steady-State and Transient Problems," EGG-NRE-11406 (1994).
- [53] H. A. Van der Vorst, "Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems," *SIAM J. Sci. Stat. Comput.*, Vol. 13, No. 2, pp. 631-644 (1992).
- [54] P. J. M. Van Laarhoven, *Theoretical and Computational Aspects of Simulated Annealing*, Centrum voor Wiskunde en Informatica, Amsterdam (1988).
- [55] M. R. Wagner and K. Koebke, "Progress in Nodal Reactor Analysis," *Proc. Topl. Mtg. Advances in Reactor Computations*, Vol. II, pp. 941, Salt Lake City, March, American Nuclear Society (1983).
- [56] M. R. Wagner, K. Koebke, and H. J. Winter, "A Non-Linear Extension of the Nodal Expansion Method," *Proc. Conf. Advances in Mathematical Methods for*

- the Solution of Nuclear Engineering Problems*, p. 43, Vol. II, Munich, 27-29 April, American Nuclear Society (1981).
- [57] J. F. Whidborne, D. W. Gu, I. Pstlethwaite, "Simulated Annealing for Multi-Objective Control System Design," UKACC International Conference on CONTROL'96: Conference Publication No. 427, 376-381 (1996).
 - [58] E. L. Wachspress, *Iterative Solution of Elliptic System and Applications to the Neutron Diffusion Equations of Reactor Physics*, Prentice-Hall, Englewood Cliffs, N.J. (1966).
 - [59] J. Wu, *The Application of Domain Decomposition and Krylov Subspace Methods to Reactor Safety Analysis*, Ph.D. Dissertation, Purdue University (1997).
 - [60] N. Zuber and J. Findlay, "Average Volumetric Concentration in Two-Phase Flow Systems," *J. Heat Transfer*, **87**, 453 (1965).

Appendix A: Derivation of the Jacobian matrix system for the EPRI void-quality correlation.

NOMENCLATURE

α	void fraction
A_x	cross sectional flow area
C_0	mixture void-quality parameter
D_e	equivalent hydraulic diameter
ϕ_{fric}^2	frictional loss two-phase multiplier
$f_{1\phi}$	single phase frictional loss
G	coolant mass flux
g	gravitational constant
h	enthalpy
h_l	saturated liquid enthalpy
h_g	saturated vapor enthalpy
j_D^g	drift flux
K	local pressure loss factor
mod	notation for “moderator”
P	pressure
P_c	critical pressure
ρ	density
ρ_{avg}	average density
ρ_l	saturated liquid density
ρ_g	saturated vapor density
σ	surface tension
u	internal energy
u_{avg}	average internal energy
u_l	saturated liquid internal energy
u_g	saturated vapor internal energy

V_{gj}	mixture void-quality parameter
x_{eq}	equilibrium quality
x_f	flow quality
x_{nvg}	net-vapor-generation quality
Δz	height of axial node
Z	square root of fuel temperature

Other Notation

To denote the spatial dependencies between axial nodes, some components of the Jacobian matrix are written with a bar on top to indicate the coupling to the node below the node of interest.

Derivations

The Jacobian of the thermal hydraulics equations involves many unknowns. We would like to reduce the number of unknowns to reduce the computational effort to solve for them. The power constraint part of the Jacobian matrix system will be dealt with at the end of the appendix. In deriving the Jacobian matrix system, the derivation for each component is first determined, and then utilizing the chain rule, the variable reduction process will be performed. The derivations given here are for the cases with the following specifications:

- The iterative EPRI model of Lellouche-Zolotar is used as the void-quality correlation.
- The EPRI subcooled void formation model is used in the subcooled boiling region.

- The spacer-void model is used to correct the predicted void value which is subsequently used to modify the moderator density. This corrected moderator density is then used for cross section updates.

The Jacobian of the V_{gj} mixture parameter

$$V_{gj} = 1.41 \left[\frac{(\rho_l - \rho_g)}{\rho_l^2} \sigma g \right]^{1/4} (1 - \alpha)^{3/2} \quad (\text{A-1})$$

$$\delta V_{gj} - \frac{\partial V_{gj}}{\partial \alpha} \delta \alpha - \left(\sum_{\theta_j = \rho_l, \rho_g, \sigma} \frac{\partial V_{gj}}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \delta P = 0, \quad (\text{A-2})$$

where

$$\frac{\partial V_{gj}}{\partial \alpha} = -2.115 \sqrt{1 - \alpha} \left[\frac{(\rho_l - \rho_g)}{\rho_l^2} \sigma g \right]^{1/4}, \quad (\text{A-3})$$

$$\frac{\partial V_{gj}}{\partial \rho_l} = \frac{-0.3525 (1 - \alpha)^{3/2} [\sigma g]^{1/4} (\rho_l - 2\rho_g)}{[\rho_l - \rho_g]^{3/4} \rho_l^{3/2}}, \quad (\text{A-4})$$

$$\frac{\partial V_{gj}}{\partial \rho_g} = \frac{-0.3525 (1 - \alpha)^{3/2} [\sigma g]^{1/4}}{[\rho_l - \rho_g]^{3/4} \rho_l^{1/2}}, \quad (\text{A-5})$$

and

$$\frac{\partial V_{gj}}{\partial \sigma} = \frac{0.3525 (1 - \alpha)^{3/2} [(\rho_l - \rho_g) g]^{1/4}}{\sigma^{3/4} \rho_l^{1/2}}. \quad (\text{A-6})$$

The Jacobian of the void (α)

The EPRI model of Lellouche-Zolotar is utilized for determining the void. There is an iteration between α , V_{gj} , and C_0 . In deriving the Jacobian of the void, there are several conditions which should be checked in order to obtain the proper derivation.

If $L_N \geq 0.2$:

$$\alpha = \frac{1}{C_0 \left(1 + \frac{Y}{x_f}\right)} . \quad (\text{A-7})$$

$$\delta\alpha - \frac{\partial\alpha}{\partial C_0} \delta C_0 - \frac{\partial\alpha}{\partial Y} \delta Y - \frac{\partial\alpha}{\partial x_f} \delta x_f = 0 , \quad (\text{A-8})$$

where

$$\frac{\partial\alpha}{\partial C_0} = -\frac{1}{C_0^2 \left(1 + \frac{Y}{x_f}\right)} , \quad (\text{A-9})$$

$$\frac{\partial\alpha}{\partial Y} = -\frac{1}{C_0 \left(1 + \frac{Y}{x_f}\right)^2 x_f} , \quad (\text{A-10})$$

$$\frac{\partial\alpha}{\partial x_f} = \frac{Y}{C_0 \left(1 + \frac{Y}{x_f}\right)^2 x_f^2} . \quad (\text{A-11})$$

If $L_N < 0.2$:

$$\alpha = \sqrt{\frac{A}{\left(1 + \frac{Y}{x_f}\right) C_1}} \quad (\text{A-12})$$

$$\delta\alpha - \frac{\partial\alpha}{\partial A}\delta A - \frac{\partial\alpha}{\partial Y}\delta Y - \frac{\partial\alpha}{\partial x_f}\delta x_f - \frac{\partial\alpha}{\partial C_1}\delta C_1 = 0 , \quad (\text{A-13})$$

where:

$$\frac{\partial\alpha}{\partial A} = \frac{x_f}{2\sqrt{Ax_f(x_f + Y)C_1}}, \quad (\text{A-14})$$

$$\frac{\partial\alpha}{\partial Y} = -\frac{Ax_f}{2\sqrt{Ax_f(x_f + Y)^3C_1}}, \quad (\text{A-15})$$

$$\frac{\partial\alpha}{\partial x_f} = \frac{AY}{2\sqrt{Ax_f(x_f + Y)^3C_1}}, \quad (\text{A-16})$$

and

$$\frac{\partial\alpha}{\partial C_1} = -\frac{Ax_f}{2\sqrt{Ax_f(x_f + Y)C_1^3}}. \quad (\text{A-17})$$

The Jacobian of the Y parameter

$$Y = (1 - x_f) \frac{\rho_g}{\rho_l} \left[1 + \frac{\rho_l AV_{gj}}{GL_N(1 - x_f)} \right]. \quad (\text{A-18})$$

$$\begin{aligned} & \delta Y - \frac{\partial Y}{\partial x_f} \delta x_f - \frac{\partial Y}{\partial G} \delta G - \frac{\partial Y}{\partial L_N} \delta L_N - \frac{\partial Y}{\partial A} \delta A - \frac{\partial Y}{\partial V_{gj}} \delta V_{gj} - \\ & \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial Y}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \delta P = 0 \end{aligned} \quad , \quad (\text{A-19})$$

where:

$$\frac{\partial Y}{\partial x_f} = -\frac{\rho_g}{\rho_l}, \quad (\text{A-20})$$

$$\frac{\partial Y}{\partial G} = -\frac{\rho_g A V_{gj}}{G^2 L_N}, \quad (\text{A-21})$$

$$\frac{\partial Y}{\partial L_N} = -\frac{\rho_g A V_{gj}}{G L_N^2}, \quad (\text{A-22})$$

$$\frac{\partial Y}{\partial A} = \frac{\rho_g V_{gj}}{G L_N}, \quad (\text{A-23})$$

$$\frac{\partial Y}{\partial V_{gj}} = \frac{\rho_g A}{G L_N}, \quad (\text{A-24})$$

$$\frac{\partial Y}{\partial \rho_l} = \frac{\rho_g (x_f - 1)}{\rho_l^2}, \quad (\text{A-25})$$

and

$$\frac{\partial Y}{\partial \rho_g} = \frac{G L_N (1 - x_f) + \rho_l A V_{gj}}{L_N G \rho_l}. \quad (\text{A-26})$$

The Jacobian of the A parameter

$$A = K_0 + (1 - K_0) \alpha^r \quad (\text{A-27})$$

$$\delta A - \frac{\partial A}{\partial K_0} \delta K_0 - \frac{\partial A}{\partial \alpha} \delta \alpha - \frac{\partial A}{\partial r} \delta r = 0, \quad (\text{A-28})$$

where:

$$\frac{\partial A}{\partial K_0} = 1 - \alpha^r, \quad (\text{A-29})$$

$$\frac{\partial A}{\partial \alpha} = -(K_0 - 1)\alpha^{r-1}r, \quad (\text{A-30})$$

and

$$\frac{\partial A}{\partial r} = -(K_0 - 1)\alpha^r \ln(\alpha). \quad (\text{A-31})$$

The Jacobian of the L_N parameter

If $\alpha = 0$, then $\delta L_N = 0$; otherwise

$$L_N = 1 - \exp(-C_1 \alpha). \quad (\text{A-32})$$

$$\delta L_N - \frac{\partial L_N}{\partial C_1} \delta C_1 - \frac{\partial L_N}{\partial \alpha} \delta \alpha = 0, \quad (\text{A-33})$$

where

$$\frac{\partial L_N}{\partial C_1} = \alpha \exp(-C_1 \alpha), \quad (\text{A-34})$$

and

$$\frac{\partial L_N}{\partial \alpha} = C_1 \exp(-C_1 \alpha). \quad (\text{A-35})$$

The Jacobian of the K_0 parameter

$$K_0 = K_1 + (1 - K_1) \left(\frac{\rho_g}{\rho_l} \right)^{1/4} \quad (\text{A-36})$$

$$\delta K_0 - \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \delta P = 0, \quad (\text{A-37})$$

where

$$\frac{\partial K_0}{\partial \rho_l} = \frac{(K_1 - 1) \rho_g^{1/4}}{4 \sqrt{\rho_l}}, \quad (\text{A-38})$$

and

$$\frac{\partial K_0}{\partial \rho_g} = - \frac{K_1 - 1}{\rho_g^{3/4} \rho_l^{1/4}}. \quad (\text{A-39})$$

The Jacobian of the r parameter

$$r = \frac{\left(1 + 1.57 \frac{\rho_g}{\rho_l} \right)}{1 - K_1} \quad (\text{A-40})$$

$$\delta r - \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \delta P = 0, \quad (\text{A-41})$$

where

$$\frac{\partial r}{\partial \rho_l} = 1.57 \frac{\rho_g}{\rho_l^2 (K_1 - 1)}, \quad (\text{A-42})$$

and

$$\frac{\partial r}{\partial \rho_g} = -1.57 \frac{1}{\rho_l(K_1 - 1)}. \quad (\text{A-43})$$

The Jacobian of the C_1 parameter

$$C_1 = \frac{4P_c^2}{P(P_c - P)} \quad (\text{A-44})$$

$$\delta C_1 - \frac{\partial C_1}{\partial P} \delta P = 0, \quad (\text{A-45})$$

where

$$\frac{\partial C_1}{\partial P} = \frac{4P_c^2(2P - P_c)}{P^2(P - P_c)^2}. \quad (\text{A-46})$$

The Jacobian of the C_0 parameter

$$C_0 = \frac{L_N}{A} \quad (\text{A-47})$$

$$\delta C_0 - \frac{\partial C_0}{\partial L_N} \delta L_N - \frac{\partial C_0}{\partial A} \delta A = 0, \quad (\text{A-48})$$

where

$$\frac{\partial C_0}{\partial L_N} = \frac{1}{A}, \quad (\text{A-49})$$

and

$$\frac{\partial C_0}{\partial A} = -\frac{L_N}{A^2}. \quad (\text{A-50})$$

The Jacobian of the equilibrium quality (x_{eq})

$$x_{eq} = \frac{h - h_l}{h_g - h_l} = \frac{\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)}{\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)} \quad (\text{A-51})$$

$$\delta x_{eq} - \frac{\partial x_{eq}}{\partial \rho} \delta \rho - \frac{\partial x_{eq}}{\partial P} \delta P - \frac{\partial x_{eq}}{\partial u} \delta u = 0 \quad (\text{A-52})$$

$$\frac{\partial x_{eq}}{\partial \rho} = -\frac{P}{\rho^2 \left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right) \right]} \quad (\text{A-53})$$

$$\frac{\partial x_{eq}}{\partial P} = \left(\frac{\frac{1}{\rho} - \frac{1}{\rho_l}}{\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)} - \frac{\left[\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right) \right] \left(\frac{1}{\rho_g} - \frac{1}{\rho_l} \right)}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right) \right]^2} \right) +$$

$$\frac{\partial x_{eq}}{\partial \rho_l} \cdot \frac{\partial \rho_l}{\partial P} + \frac{\partial x_{eq}}{\partial \rho_g} \cdot \frac{\partial \rho_g}{\partial P} + \frac{\partial x_{eq}}{\partial u_l} \cdot \frac{\partial u_l}{\partial P} + \frac{\partial x_{eq}}{\partial u_g} \cdot \frac{\partial u_g}{\partial P} \quad (\text{A-54})$$

$$\begin{aligned}
\frac{\partial x_{eq}}{\partial P} = & \left(\frac{\frac{1}{\rho} - \frac{1}{\rho_l}}{\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)} - \frac{\left[\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]\left(\frac{1}{\rho_g} - \frac{1}{\rho_l}\right)}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]^2} \right) + \\
& \left(\frac{P}{\rho_l^2 \left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]} - \frac{\left[\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]P}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]^2 \rho_l^2} \right) \cdot \frac{\partial \rho_l}{\partial P} + \\
& \left(\frac{\left[\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]P}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]^2 \rho_g^2} \right) \cdot \frac{\partial \rho_g}{\partial P} + \\
& \left(-\frac{1}{\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)} + \frac{\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]^2} \right) \cdot \frac{\partial u_l}{\partial P} + \\
& \left(-\frac{\left(u + \frac{P}{\rho}\right) - \left(u_l + \frac{P}{\rho_l}\right)}{\left[\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)\right]^2} \right) \cdot \frac{\partial u_g}{\partial P}
\end{aligned} \tag{A-55}$$

$$\frac{\partial x_{eq}}{\partial u} = \frac{1}{\left(u_g + \frac{P}{\rho_g}\right) - \left(u_l + \frac{P}{\rho_l}\right)} \tag{A-56}$$

The Jacobian of the flow quality (x_f)

The equation determining the flow quality prior to reaching thermal equilibrium ($x_f < x_{eq}$ region) is

$$x_f = \frac{x_{eq} - x_{nvg} \left(1 - \tanh \left(1 - \frac{x_{eq}}{x_{nvg}} \right) \right)}{1 - x_{nvg} \left(1 - \tanh \left(1 - \frac{x_{eq}}{x_{nvg}} \right) \right)} \quad (\text{A-57})$$

$$\delta x_f - \frac{\partial x_f}{\partial x_{eq}} \delta x_{eq} - \frac{\partial x_f}{\partial x_{nvg}} \delta x_{nvg} = 0 \quad (\text{A-58})$$

$$\frac{\partial x_f}{\partial x_{eq}} = \left(\frac{A_{x,1}^2}{A_{x,3}} \right) - \frac{A_{x,2}(-1 + A_{x,1}^2)}{A_{x,3}^2} \quad (\text{A-59})$$

$$\frac{\partial x_f}{\partial x_{nvg}} = \left(\frac{1}{A_{x,3}} - \frac{A_{x,2}}{A_{x,3}^2} \right) \left(A_{x,1} - 1 + \frac{x_{eq}}{x_{nvg}} (1 - A_{x,1}^2) \right) \quad (\text{A-60})$$

where

$$A_{x,1} = \tanh \left(1 - \frac{x_{eq}}{x_{nvg}} \right), \quad A_{x,2} = x_{eq} - x_{nvg} (1 - A_{x,1}), \quad A_{x,3} = 1 - x_{nvg} (1 - A_{x,1}).$$

Once thermal equilibrium is reached:

$$\frac{\partial x_f}{\partial x_{eq}} = 1.0 \quad (\text{A-61})$$

$$\frac{\partial x_f}{\partial x_{nvg}} = 0.0 \quad (\text{A-62})$$

The Jacobian of the net vapor generation quality (x_{nvq})

The x_{nvq} can be determined by the following formula:

$$x_{nvq} = \frac{c_p z}{h_{fg}}, \quad (\text{A-63})$$

where

$$z = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}, \quad (\text{A-64})$$

$$A = 4H_B(H_{DB} + H_{HN})^2 \quad (\text{A-65})$$

$$B = 2H_{DB}^2 \left(H_{HN} + \frac{H_{DB}}{2} \right) + 8q''H_B(H_{DB} + H_{HN}) \quad (\text{A-66})$$

$$C = q''(4H_Bq'' + H_{DB}^2) \quad (\text{A-67})$$

given

q'' wall heat flux at point of departure [BTU/hr-ft²]

P pressure [psia]

Re single-phase Reynolds number

D_e equivalent hydraulic diameter [ft]

Pr Prandtl number

k thermal conductivity [BTU/hr-ft-°F]

Define

$$H_{HN} = C_{HN} Re^{0.662} Pr \frac{k}{D_e} \quad (\text{A-68})$$

$$H_{DB} = C_{DB} Re^{0.8} Pr^{0.4} \frac{k}{D_e} \quad (A-69)$$

$$H_B = 193 \exp\left(\frac{P}{630}\right) \quad (A-70)$$

where

$$C_{DB} = 0.033\varepsilon + 0.013 \text{ for rod bundles}$$

$$C_{HN} = \frac{0.02D_e}{4D_{rod}} \text{ for rod bundles}$$

$$\varepsilon = \frac{A_{flow}}{A_{flow} + A_{rods}}$$

D_{rod} is heated rod radius [ft]

A_{flow} is the active flow area [ft²]

$$A_{rods} = N_{rod} \pi D_{rod}^2$$

$$\begin{aligned} \delta x_{nvg} - \frac{\partial x_{nvg}}{\partial \phi_1} \delta \phi_1 - \frac{\partial x_{nvg}}{\partial \phi_2} \delta \phi_2 - \frac{\partial x_{nvg}}{\partial P} \delta P - \frac{\partial x_{nvg}}{\partial Z} \delta Z - \\ \frac{\partial x_{nvg}}{\partial G} \delta G - \frac{\partial x_{nvg}}{\partial \rho} \delta \rho - \overline{\frac{\partial x_{nvg}}{\partial \rho} \delta \bar{\rho}} = 0 \end{aligned} \quad (A-71)$$

$$\begin{aligned}
\frac{\partial x_{nvq}}{\partial \rho} &= \frac{\partial x_{nvq}}{\partial B} \cdot \frac{\partial B}{\partial q} \cdot \frac{\partial q}{\partial \kappa \Sigma_f} \cdot \frac{\partial}{\partial \rho} \kappa \Sigma_f + \frac{\partial x_{nvq}}{\partial C} \cdot \frac{\partial C}{\partial q} \cdot \frac{\partial q}{\partial \kappa \Sigma_f} \cdot \frac{\partial}{\partial \rho} \kappa \Sigma_f \\
&= \left(\frac{c_{pf}}{h_{fg}} \cdot \left(\frac{-B + \sqrt{B^2 - 4AC}}{2A\sqrt{B^2 - 4AC}} \right) \right) (8H_B(H_{DB} + H_{HN})) \frac{\phi}{2} \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right) + \\
&\quad \left(\frac{c_{pf}}{h_{fg}} \cdot \frac{1}{\sqrt{B^2 - 4AC}} \right) (8H_B q + H_{DB}^2) \frac{\phi}{2} \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right)
\end{aligned} \tag{A-72}$$

$$\begin{aligned}
\frac{\partial x_{nvq}}{\partial Z} &= \left(\frac{c_{pf}}{h_{fg}} \cdot \left(\frac{-B + \sqrt{B^2 - 4AC}}{2A\sqrt{B^2 - 4AC}} \right) \right) (8H_B(H_{DB} + H_{HN})) \phi \left(\frac{\partial}{\partial Z} \kappa \Sigma_f \right) + \\
&\quad \left(\frac{c_{pf}}{h_{fg}} \cdot \frac{1}{\sqrt{B^2 - 4AC}} \right) (8H_B q + H_{DB}^2) \phi \left(\frac{\partial}{\partial Z} \kappa \Sigma_f \right)
\end{aligned} \tag{A-73}$$

$$\begin{aligned}
\frac{\partial x_{nvq}}{\partial \phi_g} &= \left(\frac{c_{pf}}{h_{fg}} \cdot \left(\frac{-B + \sqrt{B^2 - 4AC}}{2A\sqrt{B^2 - 4AC}} \right) \right) (8H_B(H_{DB} + H_{HN})) \kappa \Sigma_{f_g} + \\
&\quad \left(\frac{c_{pf}}{h_{fg}} \cdot \frac{1}{\sqrt{B^2 - 4AC}} \right) (8H_B q + H_{DB}^2) \kappa \Sigma_{f_g}
\end{aligned} \tag{A-74}$$

$$\begin{aligned}
\frac{\partial x_{nvq}}{\partial P} &= \left(\frac{c_{pf}}{h_{fg}} \right) \left(\frac{-2AC - B\sqrt{B^2 - 4AC} + B^2}{2A^2\sqrt{B^2 - 4AC}} \right) \cdot \frac{\partial A}{\partial P} + \\
&\quad \left(\frac{c_{pf}}{h_{fg}} \right) \left(\frac{-B + \sqrt{B^2 - 4AC}}{2A\sqrt{B^2 - 4AC}} \right) \cdot \frac{\partial B}{\partial P} + \left(\frac{c_{pf}}{h_{fg}} \cdot \frac{1}{\sqrt{B^2 - 4AC}} \right) \cdot \frac{\partial C}{\partial P}
\end{aligned} \tag{A-75}$$

where

$$\frac{\partial A}{\partial P} = 4(H_{DB} + H_{HN})^2 \cdot \frac{\partial H_B}{\partial P}, \quad \frac{\partial B}{\partial P} = 8q(H_{DB} + H_{HN}) \cdot \frac{\partial H_B}{\partial P}, \quad \frac{\partial C}{\partial P} = 4q^2 \cdot \frac{\partial H_B}{\partial P}, \text{ and}$$

$$\frac{\partial H_B}{\partial P} = \frac{1}{(630)(0.072)^2} \exp\left(\frac{P}{630}\right).$$

The Jacobian of the square root of the fuel temperature ($Z = \sqrt{T}$)

$$T = Z^2 - (T_{mod} - T_{mod,ref}) - P(q') - \Delta T_{eff}(q', BU)$$

$$\frac{\partial T}{\partial Z} \delta Z - \frac{\partial T}{\partial \phi_1} \delta \phi_1 - \frac{\partial T}{\partial \phi_2} \delta \phi_2 - \frac{\partial T}{\partial P} \delta P - \frac{\partial T}{\partial u} \delta u - \frac{\partial T}{\partial \bar{u}} \delta \bar{u} - \frac{\partial T}{\partial \rho} \delta \rho - \frac{\partial T}{\partial \bar{\rho}} \delta \bar{\rho} = 0 \quad (\text{A-76})$$

$$\text{where: } q' = \frac{1}{\Delta Z} (\kappa \Sigma_{f1} \phi_1 + \kappa \Sigma_{f2} \phi_2)$$

$$\begin{aligned} \frac{\partial T}{\partial Z} &= 2Z - \frac{\partial}{\partial Z} (P(\kappa \Sigma_f \phi) + \kappa \Sigma_f \phi \cdot BU_{corr}) = 2Z - \left(\frac{\partial P}{\partial q} + BU_{corr} \right) \frac{\partial}{\partial Z} (\kappa \Sigma_f) \phi \\ &= 2Z - \tau_1 \frac{\partial}{\partial Z} (\kappa \Sigma_f) \phi \end{aligned} \quad (\text{A-77})$$

$$\text{where } \tau_1 = \frac{\partial P}{\partial q} + \frac{\partial}{\partial q} (\Delta T_{eff}(q', BU)).$$

$$\frac{\partial T}{\partial \phi_g} = - \frac{\partial}{\partial \phi_g} (P(q') + \Delta T_{eff}(q', BU)) = -\tau_1 \frac{\partial}{\partial \phi_g} \kappa \Sigma_{fg} \phi_g = -\tau_1 \kappa \Sigma_{fg} \quad (\text{A-78})$$

$$\frac{\partial T}{\partial P} = - \frac{\partial T_{mod}}{\partial P} \quad (\text{A-79})$$

$$\frac{\partial T}{\partial u} = - \frac{\partial T_{mod}}{\partial u_{avg}} \cdot \frac{1}{2} \quad (\text{A-80})$$

$$\frac{\overline{\partial T}}{\partial u} = \frac{\partial T_{mod}}{\partial u_{avg}} \cdot \frac{1}{2} \quad (\text{A-81})$$

$$\frac{\partial T}{\partial \rho} = -\tau_1 \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right) \cdot \frac{\phi}{2} \quad (\text{A-82})$$

$$\frac{\overline{\partial T}}{\partial \rho} = -\tau_1 \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right) \cdot \frac{\phi}{2} \quad (\text{A-83})$$

The Jacobian of the momentum equation (pressure)

Note that the interphase momentum transfer is not modeled in FORMOSA-B. The Jacobian of the momentum equation can be written as

$$\frac{\partial M}{\partial P} \delta P - \frac{\overline{\partial M}}{\partial P} \delta \bar{P} - \frac{\partial M}{\partial x_f} \delta x_f - \frac{\partial M}{\partial \mathbf{G}} \delta \mathbf{G} - \frac{\partial M}{\partial \rho} \delta \rho - \frac{\overline{\partial M}}{\partial \rho} \delta \bar{\rho} = 0 \quad (\text{A-84})$$

$$\frac{\partial M}{\partial P} = 1 - \left(\frac{\Delta z f G^2}{2 D_e \rho_l^2} \right)_{k+1/2} - \left(\frac{K G^2}{2 \rho_l^2} \right)_{k+1/2} \quad (\text{A-85})$$

$$\frac{\overline{\partial M}}{\partial P} = 1.0 \quad (\text{A-86})$$

$$\frac{\partial M}{\partial x_f} = \frac{f_{1\phi} G^2}{2(D_e \rho_l)} \cdot \frac{\partial \phi_{fric}^2}{\partial x_f} \quad (\text{A-87})$$

$$\frac{\partial M}{\partial \rho} = \left(\frac{G^2}{\rho^2} \right)_{k+1/2} - \Delta z_{k+1/2} g \quad (\text{A-88})$$

$$\frac{\overline{\partial M}}{\partial \rho} = - \frac{(G^2 A x)_{k-1/2}}{\rho_{k-1/2}^2 A x_{k+1/2}} \quad (\text{A-89})$$

The Jacobian of the mass flux

Assuming constant mass flux during Newton iteration:

$$\delta G = 0 \quad (\text{A-90})$$

The Jacobian of the internal energy

Note that when the spacer void model is used, although the remaining of the thermal hydraulics calculation use the uncorrected density, the cross section term is determined using the corrected density which is defined as:

$$\rho = \rho_l + \widehat{\alpha}(\rho_g - \rho_l), \quad (\text{A-91})$$

where $\widehat{\alpha}$ is the corrected void fraction and is defined as:

$$\widehat{\alpha} = 2\left(\frac{\alpha + \bar{\alpha}}{2} + cGA\left[\beta_0 + \beta_1\left(\frac{\alpha + \bar{\alpha}}{2}\right) + \beta_2\left(\frac{\alpha + \bar{\alpha}}{2}\right)^2\right]\right) - \bar{\alpha}. \quad (\text{A-92})$$

$$\begin{aligned} \delta u - \frac{\partial u}{\partial u} \delta \bar{u} - \frac{\partial u}{\partial \phi_1} \delta \phi_1 - \frac{\partial u}{\partial \phi_2} \delta \phi_2 - \frac{\partial u}{\partial V_{gj}} \delta V_{gj} - \frac{\partial u}{\partial \bar{V}_{gj}} \delta \bar{V}_{gj} - \frac{\partial u}{\partial P} \delta P \\ - \frac{\partial u}{\partial Z} \delta Z - \frac{\partial u}{\partial G} \delta G - \frac{\partial u}{\partial \rho} \delta \rho - \frac{\partial u}{\partial \bar{\rho}} \delta \bar{\rho} - \frac{\partial u}{\partial \alpha} \delta \alpha = 0 \end{aligned} \quad (\text{A-93})$$

$$\frac{\partial u}{\partial u} = \frac{1}{(GAx)_{k+1/2}} ((GAx)_{k-1/2} - \dot{m}_{BY} - \dot{m}_{WR}) \quad (\text{A-94})$$

$$\frac{\partial u}{\partial \phi_g} = \frac{\kappa \Sigma_{fg} \bar{A}_x \Delta z_k}{(GAx)_{k+1/2}} \quad (\text{A-95})$$

$$\frac{\partial u}{\partial Z} = \frac{\bar{A}_x \Delta z_k}{(GAx)_{k+1/2}} \left(\frac{\partial}{\partial Z} \kappa \Sigma_f \right) \phi \quad (\text{A-96})$$

$$\begin{aligned} \frac{\partial u}{\partial \rho} = & \frac{P_k}{\rho_{k+1/2}^2} + \frac{P_k}{(GAx)_{k+1/2}} \left(\frac{(\rho_l - \rho_g)}{\rho^2} \alpha V_{gj} Ax \right)_{k+1/2} + \\ & \frac{1}{(GAx)_{k+1/2}} \left(\frac{\rho_l \rho_g}{\rho^2} (u_g - u_l) \alpha V_{gj} Ax \right)_{k+1/2} \end{aligned} \quad (A-97)$$

$$\begin{aligned} \overline{\frac{\partial u}{\partial \rho}} = & -\frac{P_k}{\rho_{k-1/2}^2} \frac{(GAx)_{k-1/2}}{(GAx)_{k+1/2}} - \frac{P_k}{(GAx)_{k+1/2}} \left(\frac{(\rho_l - \rho_g)}{\rho^2} \alpha V_{gj} Ax \right)_{k-1/2} \\ & - \frac{1}{(GAx)_{k+1/2}} \left(\frac{\rho_l \rho_g}{\rho^2} (u_g - u_l) \alpha V_{gj} Ax \right)_{k-1/2} \end{aligned} \quad (A-98)$$

$$\frac{\partial u}{\partial V_{gj}} = -\frac{P_k}{(GAx)_{k+1/2}} \left(\frac{(\rho_l - \rho_g)}{\rho} \alpha Ax \right)_{k+1/2} - \frac{1}{(GAx)_{k+1/2}} \left(\frac{\rho_l \rho_g}{\rho} (u_g - u_l) \alpha Ax \right)_{k+1/2} \quad (A-99)$$

$$\overline{\frac{\partial u}{\partial V_{gj}}} = \frac{P_k}{(GAx)_{k+1/2}} \left(\frac{(\rho_l - \rho_g)}{\rho} \alpha Ax \right)_{k-1/2} + \frac{1}{(GAx)_{k+1/2}} \left(\frac{\rho_l \rho_g}{\rho} (u_g - u_l) \alpha Ax \right)_{k-1/2} \quad (A-100)$$

$$\begin{aligned} \frac{\partial u}{\partial P} = & - \left(\frac{\partial u}{\partial \rho_l} \cdot \frac{\partial \rho_l}{\partial P} + \frac{\partial u}{\partial \rho_g} \cdot \frac{\partial \rho_g}{\partial P} + \frac{\partial u}{\partial u_l} \cdot \frac{\partial u_l}{\partial P} + \frac{\partial u}{\partial u_g} \cdot \frac{\partial u_g}{\partial P} \right) + \\ & \left\{ \frac{1}{\rho_{k+1/2}} - \frac{(GAx)_{k-1/2}}{(GAx)_{k+1/2}} \frac{1}{\rho_{k-1/2}} + \frac{1}{(GAx)_{k+1/2}} \left(\left[\frac{(\rho_l - \rho_g)}{\rho} j_D^g A_x \right]_{k-1/2}^{k+1/2} \right) \right\} \end{aligned} \quad (A-101)$$

$$\begin{aligned} \frac{\partial u}{\partial \alpha} = & -\frac{P_k}{(GA)_{k+1/2}} \left(\left[\frac{(\rho_l - \rho_g)}{\rho} V_{gj} A_x \right]_{k+1/2} \right) - \frac{1}{(GA)_{k+1/2}} \left(\left[\frac{\rho_l \rho_g}{\rho} V_{gj} A_x \right]_{k+1/2} \right) + \\ & \frac{\bar{Ax} \Delta z_k}{(GA_x)_{k+1/2}} \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right) \frac{\phi}{2} (\rho_g - \rho_l) [1 + cGA(\beta_1 + \beta_2(\alpha + \bar{\alpha}))] \end{aligned} \quad (A-102)$$

$$\begin{aligned}
\frac{\partial \bar{u}}{\partial \alpha} = & \frac{P_k}{(GA)_{k+1/2}} \left(\left[\frac{(\rho_l - \rho_g)}{\rho} V_{gj} A_x \right]_{k-1/2} \right) + \frac{1}{(GA)_{k+1/2}} \left(\left[\frac{\rho_l \rho_g}{\rho} V_{gj} A_x \right]_{k-1/2} \right) + \\
& \frac{\bar{A}_x \Delta z_k}{(GA_x)_{k+1/2}} \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \right) \frac{\phi}{2} \left\{ (\rho_g - \rho_l) c GA (\beta_1 + \beta_2 (\alpha + \bar{\alpha})) + \right. \\
& \left. (\bar{\rho}_g - \bar{\rho}_l) [1 + c GA (\beta_1 + \beta_2 (\bar{\alpha} + \bar{\bar{\alpha}}))] \right\}
\end{aligned} \tag{A-103}$$

The Jacobian of the density

$$\delta \rho - \frac{\partial \rho}{\partial \alpha} \delta \alpha - \frac{\partial \rho}{\partial P} \delta P - \frac{\partial \rho}{\partial u} \delta u = 0 \tag{A-104}$$

The equation for density depends on whether the water is single or two phase.

If $(\alpha > 1 \times 10^{-5} \text{ and } x_f > 0)$

$$\rho = \rho_l + \alpha(\rho_g - \rho_l)$$

otherwise

$$\rho = f(u, P)$$

Single Phase

$$\rho = f(u, P) \tag{A-105}$$

$$\frac{\partial \rho}{\partial u} = \frac{\partial}{\partial u} f(u, P) \tag{A-106}$$

$$\frac{\partial \rho}{\partial P} = \frac{\partial}{\partial P} f(u, P) \tag{A-107}$$

$$\frac{\partial \rho}{\partial \alpha} = 0 \tag{A-108}$$

Two Phase

$$\rho = \rho_l + \alpha(\rho_g - \rho_l) \tag{A-109}$$

$$\delta\rho - \left(\frac{\partial\rho}{\partial\alpha}\right)\delta\alpha - \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial\rho}{\partial\theta_j} \cdot \frac{\partial\theta_j}{\partial P}\right)\delta P = 0, \quad (\text{A-110})$$

where

$$\frac{\partial\rho}{\partial\alpha} = (\rho_g - \rho_l), \quad (\text{A-111})$$

$$\frac{\partial\rho}{\partial\rho_l} = 1 - \alpha, \quad (\text{A-112})$$

and

$$\frac{\partial\rho}{\partial\rho_g} = \alpha. \quad (\text{A-113})$$

From Eq. (A-90), it is clear that the mass flux will not play any role in solving the Jacobian. Therefore, the δG component can be immediately eliminated from the system of equations. The next variable to eliminate is x_{eq} . Solving Eq. (A-52) for δx_{eq} and substituting the result into Eq. (A-58) gives us

$$-\left(\frac{\partial x_f}{\partial x_{eq}}\right)\left\{\left(\frac{\partial x_{eq}}{\partial P}\right)\delta P + \left(\frac{\partial x_{eq}}{\partial u}\right)\delta u + \left(\frac{\partial x_{eq}}{\partial \rho}\right)\delta\rho\right\} + \delta x_f \left(\frac{\partial x_f}{\partial x_{nvq}}\right)\delta x_{nvq} = 0. \quad (\text{A-114})$$

Next, solve Eq. (A-45) and plug in the result to Eq. (A-33) to get the following expression

$$\delta L_N - \left(\frac{\partial L_N}{\partial C_1}\right)\left(\frac{\partial C_1}{\partial P}\right)\delta P - \left(\frac{\partial L_N}{\partial \alpha}\right)\delta\alpha = 0. \quad (\text{A-115})$$

Solving Eqs. (A-37) and (A-41) and plug in the results into Eq. (A-28), we will get

$$\delta A - \left\{ \left(\frac{\partial A}{\partial K_0} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) + \left(\frac{\partial A}{\partial r} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \right\} \delta P - \left(\frac{\partial A}{\partial \alpha} \right) \delta \alpha = 0$$

(A-116)

Now, we can solve Eq. (A-115) for δL_N and Eq. (A-116) for δA and substitute the corresponding variable in Eq. (A-19) and Eq. (A-48) appropriately to get the following expressions

$$\begin{aligned} & \delta Y - \frac{\partial Y}{\partial x_f} \delta x_f - \frac{\partial Y}{\partial V_{gj}} \delta V_{gj} - \left\{ \frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial Y \partial A}{\partial A \partial \alpha} \right\} \delta \alpha - \\ & \left\{ \frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \frac{\partial Y}{\partial A} \left[\left(\frac{\partial A}{\partial K_0} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) + \left(\frac{\partial A}{\partial r} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \right] + \right. \\ & \quad \left. \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial Y}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right\} \delta P = 0 \end{aligned}$$

(A-117)

and

$$\begin{aligned} & \delta C_0 - \left[\frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial C_0}{\partial A} \frac{\partial A}{\partial \alpha} \right] \delta \alpha - \\ & \left\{ \frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \frac{\partial C_0}{\partial A} \left[\frac{\partial A}{\partial K_0} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} + \frac{\partial A}{\partial r} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right] \right\} \delta P = 0 \end{aligned}$$

(A-118)

Next, solve Eqs. (A-117) and (A-118) for δY and δC_0 , respectively, and then plug in the results into Eq. (A-8) to obtain the following expression

$$\begin{aligned}
& \left\{ 1 - \frac{\partial \alpha}{\partial Y} \left[\frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial Y \partial A}{\partial A \partial \alpha} \right] - \frac{\partial \alpha}{\partial C_0} \left[\frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial C_0 \partial A}{\partial A \partial \alpha} \right] \right\} \delta \alpha - \\
& \left[\left(\frac{\partial \alpha}{\partial x_f} \right) + \frac{\partial \alpha \partial Y}{\partial Y \partial x_f} \right] \delta x_f - \frac{\partial \alpha \partial Y}{\partial Y \partial V_{gj}} \delta V_{gj} - \\
& \left\{ \frac{\partial \alpha}{\partial C_0} \left[\frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \frac{\partial C_0}{\partial A} \left[\frac{\partial A}{\partial K_0} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} + \frac{\partial A}{\partial r} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right] \right] + \right. \\
& \quad \left. \frac{\partial \alpha}{\partial Y} \left[\frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial Y}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} + \right. \right. \\
& \quad \left. \left. \frac{\partial Y}{\partial A} \left[\left(\frac{\partial A}{\partial K_0} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) + \left(\frac{\partial A}{\partial r} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \right] \right] \right\} \delta P = 0.
\end{aligned} \tag{A-119}$$

Solve Eq. (A-114) for δx_f and plug in the result into Eqs. (A-84) and (A-119) to get the following expressions

$$\begin{aligned}
& \left[\frac{\partial M}{\partial P} - \frac{\partial M}{\partial x_f} \frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial P} \right] \delta P - \frac{\overline{\partial M}}{\partial P} \delta \bar{P} + \left(\frac{\partial M}{\partial \rho} + \frac{\partial M}{\partial x_f} \frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial \rho} \right) \delta \rho - \frac{\overline{\partial M}}{\partial \rho} \delta \bar{\rho} - \\
& \frac{\partial M}{\partial x_f} \frac{\partial x_f}{\partial x_{nvq}} \delta x_{nvq} - \frac{\partial M}{\partial x_f} \frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial u} \delta u = 0
\end{aligned} \tag{A-120}$$

and

$$\begin{aligned}
& \left\{ 1 - \frac{\partial \alpha}{\partial Y} \left[\frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial Y \partial A}{\partial A \partial \alpha} \right] - \frac{\partial \alpha}{\partial C_0} \left[\frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial \alpha} + \frac{\partial C_0 \partial A}{\partial A \partial \alpha} \right] \right\} \delta \alpha - \\
& \left[\left(\frac{\partial \alpha}{\partial x_f} \right) + \frac{\partial \alpha \partial Y}{\partial Y \partial x_f} \right] \left(\frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial u} \right) \delta u - \frac{\partial \alpha \partial Y}{\partial Y \partial V_{gj}} \delta V_{gj} - \\
& \left\{ \frac{\partial \alpha}{\partial C_0} \left[\frac{\partial C_0}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \frac{\partial C_0}{\partial A} \left[\frac{\partial A}{\partial K_0} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} + \frac{\partial A}{\partial r} \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right] \right] + \right. \\
& \left[\left(\frac{\partial \alpha}{\partial x_f} \right) + \frac{\partial \alpha \partial Y}{\partial Y \partial x_f} \right] \frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial P} + \frac{\partial \alpha}{\partial Y} \left[\frac{\partial Y}{\partial L_N} \frac{\partial L_N}{\partial C_1} \frac{\partial C_1}{\partial P} + \sum_{\theta_j = \rho_l, \rho_g} \frac{\partial Y}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} + \right. \\
& \left. \frac{\partial Y}{\partial A} \left[\left(\frac{\partial A}{\partial K_0} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial K_0}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) + \left(\frac{\partial A}{\partial r} \right) \left(\sum_{\theta_j = \rho_l, \rho_g} \frac{\partial r}{\partial \theta_j} \cdot \frac{\partial \theta_j}{\partial P} \right) \right] \right] \left. \right\} \delta P - \\
& \left[\left(\frac{\partial \alpha}{\partial x_f} \right) + \frac{\partial \alpha \partial Y}{\partial Y \partial x_f} \right] \frac{\partial x_f}{\partial x_{nvq}} \delta x_{nvq} - \left[\left(\frac{\partial \alpha}{\partial x_f} \right) + \frac{\partial \alpha \partial Y}{\partial Y \partial x_f} \right] \frac{\partial x_f}{\partial x_{eq}} \frac{\partial x_{eq}}{\partial \rho} \delta \rho = 0
\end{aligned}
\tag{A-121}$$

There are seven remaining variables, namely δV_{gj} , $\delta \alpha$, δx_{nvq} , δZ , δP , δu , and $\delta \rho$. There are also two other variables ($\delta \phi_1$ and $\delta \phi_2$) which exist due to the coupling of the thermal hydraulics equations to the neutron diffusion equation through power density. The inter-dependencies among these variables are summarized in Table A.1. Note that the shaded boxes indicates axial coupling to the node below node of interest.

Table A.1: Variable Dependencies.

Eq.	Var.	V_{gj}	α	x_{nvg}	Z	P	u	ρ	ϕ_1	ϕ_2
(A-2)	V_{gj}	X	X			X				
(A-121)	α	X	X	X		X	X	X		
(A-71)	x_{nvg}			X	X	X		X	X	X
(A-76)	Z				X	X	X	X	X	X
(A-120)	P			X		X	X	X		
(A-93)	u	X	X		X	X	X	X	X	X
(A-104)/ (A-110)	ρ		X			X	X	X		

Since the equations are relatively long, they can be written in a more compact fashion using the following convention. $J_{x,y}$ denotes the derivative of variable x with respect to variable y , both of the same node. $\overline{J_{x,y}}$ denotes the derivative of variable x on node z with respect to variable y on node $z - 1$, where z is the axial node index. The components constructing a particular $J_{x,y}$ can be obtained by inspection from the equations listed in column 1 of Table A.1.

$$\delta V_{gj} + J_{V,\alpha} \delta \alpha + J_{V,P} \delta P = 0 \quad (\text{A-122})$$

$$J_{\alpha,V} \delta V_{gj} + J_{\alpha,\alpha} \delta \alpha + J_{\alpha,x_{nvg}} \delta x_{nvg} + J_{\alpha,P} \delta P + J_{\alpha,u} \delta u + J_{\alpha,\rho} \delta \rho = 0 \quad (\text{A-123})$$

$$\begin{aligned} & \delta x_{nvg} + J_{x_{nvg}, Z} \delta Z + J_{x_{nvg}, P} \delta P + J_{x_{nvg}, \rho} \delta \rho + \\ & \overline{J_{x_{nvg}, \rho}} \delta \bar{\rho} + J_{x_{nvg}, \phi_1} \delta \phi_1 + J_{x_{nvg}, \phi_2} \delta \phi_2 = 0 \end{aligned} \quad (\text{A-124})$$

$$\begin{aligned} & J_{Z, Z} \delta Z + J_{Z, P} \delta P + J_{Z, u} \delta u + \overline{J_{Z, u}} \delta \bar{u} + J_{Z, \rho} \delta \rho + \overline{J_{Z, \rho}} \delta \bar{\rho} + \\ & J_{Z, \phi_1} \delta \phi_1 + J_{Z, \phi_2} \delta \phi_2 = 0 \end{aligned} \quad (\text{A-125})$$

$$J_{P, P} \delta P - \delta \bar{P} + J_{P, x_{nvg}} \delta x_{nvg} + J_{P, u} \delta u + J_{P, \rho} \delta \rho + \overline{J_{P, \rho}} \delta \bar{\rho} = 0 \quad (\text{A-126})$$

$$\begin{aligned} & \delta u + \overline{J_{u, u}} \delta \bar{u} + J_{u, v} \delta V_{gj} + J_{u, \alpha} \delta \alpha + \overline{J_{u, \alpha}} \delta \bar{\alpha} + J_{u, Z} \delta Z + J_{u, P} \delta P + \\ & J_{u, \rho} \delta \rho + \overline{J_{u, \rho}} \delta \bar{\rho} + J_{u, \phi_1} \delta \phi_1 + J_{u, \phi_2} \delta \phi_2 = 0 \end{aligned} \quad (\text{A-127})$$

$$\delta \rho + J_{\rho, \alpha} \delta \alpha + J_{\rho, P} \delta P + J_{\rho, u} \delta u = 0 \quad (\text{A-128})$$

Solve Eq. (A-125) for δZ :

$$\delta Z = -J_{Z, Z}^{-1} \left\{ J_{Z, P} \delta P + J_{Z, u} \delta u + \overline{J_{Z, u}} \delta \bar{u} + J_{Z, \rho} \delta \rho + \overline{J_{Z, \rho}} \delta \bar{\rho} + J_{Z, \phi_1} \delta \phi_1 + J_{Z, \phi_2} \delta \phi_2 \right\} \quad (\text{A-129})$$

Substitute Eq. (A-129) into Eqs. (A-124) and (A-127).

Into Eq. (A-124):

$$\begin{aligned} & \delta x_{nvg} + \left(J_{x_{nvg}, P} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, P} \right) \delta P + \left(J_{x_{nvg}, \rho} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \rho} \right) \delta \rho + \\ & \left(\overline{J_{x_{nvg}, \rho}} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}} \right) \delta \bar{\rho} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, u} \delta u - J_{x_{nvg}, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}} \delta \bar{u} + \\ & \left(J_{x_{nvg}, \phi_1} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right) \delta \phi_1 + \left(J_{x_{nvg}, \phi_2} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right) \delta \phi_2 = 0 \end{aligned} \quad (\text{A-130})$$

Into Eq. (A-127):

$$\begin{aligned}
& (1 - J_{u, Z} J_{Z, Z}^{-1} J_{Z, u}) \delta u + (\overline{J_{u, u}} - J_{u, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}}) \delta \bar{u} + J_{u, V} \delta V_{gj} + J_{u, \alpha} \delta \alpha + \overline{J_{u, \alpha}} \delta \bar{\alpha} + \\
& (J_{u, P} - J_{u, Z} J_{Z, Z}^{-1} J_{Z, P}) \delta P + (J_{u, \rho} - J_{u, Z} J_{Z, Z}^{-1} J_{Z, \rho}) \delta \rho + (\overline{J_{u, \rho}} - J_{u, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}}) \delta \bar{\rho} + \\
& \left(J_{u, \phi_1} - J_{u, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right) \delta \phi_1 + \left(J_{u, \phi_2} - J_{u, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right) \delta \phi_2 = 0
\end{aligned} \tag{A-131}$$

Solve Eq. (A-130) for δx_{nvq} :

$$\begin{aligned}
\delta x_{nvq} = & - \left\{ \left(J_{x_{nvq}, P} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, P} \right) \delta P + \left(J_{x_{nvq}, \rho} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \rho} \right) \delta \rho + \right. \\
& \left(\overline{J_{x_{nvq}, \rho}} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}} \right) \delta \bar{\rho} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, u} \delta u - J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}} \delta \bar{u} + \\
& \left. \left(J_{x_{nvq}, \phi_1} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right) \delta \phi_1 + \left(J_{x_{nvq}, \phi_2} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right) \delta \phi_2 \right\}
\end{aligned} \tag{A-132}$$

Substitute Eq. (A-132) into Eqs. (A-123) and (A-126).

Into Eq. (A-123):

$$\begin{aligned}
& J_{\alpha, \alpha} \delta \alpha + J_{\alpha, V} \delta V_{gj} + \left[J_{\alpha, P} - J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, P} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, P} \right) \right] \delta P + \\
& \left[J_{\alpha, \rho} - J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \rho} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \rho} \right) \right] \delta \rho - \\
& J_{\alpha, x_{nvq}} \left(\overline{J_{x_{nvq}, \rho}} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}} \right) \delta \bar{\rho} + \\
& \left[J_{\alpha, u} + J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, u} \right) \right] \delta u + J_{\alpha, x_{nvq}} J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}} \delta \bar{u} - \\
& J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \phi_1} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right) \delta \phi_1 - \\
& J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \phi_2} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right) \delta \phi_2 = 0
\end{aligned} \tag{A-133}$$

Into Eq. (A-126):

$$\begin{aligned}
& \left(J_{P,P} - J_{P,x_{nv}g} \left(J_{x_{nv}g,P} - J_{x_{nv}g,Z} J_{Z,Z}^{-1} J_{Z,P} \right) \right) \delta P - \delta \bar{P} + \\
& \left(J_{P,u} + J_{P,x_{nv}g} J_{x_{nv}g,Z} J_{Z,Z}^{-1} J_{Z,u} \right) \delta u + \left(J_{P,\rho} - J_{P,x_{nv}g} \left(J_{x_{nv}g,\rho} - J_{x_{nv}g,Z} J_{Z,Z}^{-1} J_{Z,\rho} \right) \right) \delta \rho + \\
& \left(\overline{J_{P,\rho}} - J_{P,x_{nv}g} \left(\overline{J_{x_{nv}g,\rho}} - J_{x_{nv}g,Z} J_{Z,Z}^{-1} \overline{J_{Z,\rho}} \right) \right) \delta \bar{\rho} + J_{P,x_{nv}g} J_{x_{nv}g,Z} J_{Z,Z}^{-1} \overline{J_{Z,u}} \delta \bar{u} - \\
& J_{P,x_{nv}g} \left(J_{x_{nv}g,\phi_1} - J_{x_{nv}g,Z} J_{Z,Z}^{-1} J_{Z,\phi_1} \right) \delta \phi_1 - \\
& J_{P,x_{nv}g} \left(J_{x_{nv}g,\phi_2} - J_{x_{nv}g,Z} J_{Z,Z}^{-1} J_{Z,\phi_2} \right) \delta \phi_2 = 0
\end{aligned} \tag{A-134}$$

Next, solve Eq. (A-122) for δV_{gj} :

$$\delta V_{gj} = -(J_{V,\alpha} \delta \alpha + J_{V,P} \delta P) . \tag{A-135}$$

Substitute Eq. (A-135) into Eqs. (A-131) and (A-133).

Into Eq. (A-131):

$$\begin{aligned}
& (1 - J_{u,Z} J_{Z,Z}^{-1} J_{Z,u}) \delta u + (\overline{J_{u,u}} - J_{u,Z} J_{Z,Z}^{-1} \overline{J_{Z,u}}) \delta \bar{u} + (J_{u,\alpha} - J_{u,V} J_{V,\alpha}) \delta \alpha + \\
& (J_{u,P} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,P} - J_{u,V} J_{V,P}) \delta P + (J_{u,\rho} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\rho}) \delta \rho + \\
& (\overline{J_{u,\rho}} - J_{u,Z} J_{Z,Z}^{-1} \overline{J_{Z,\rho}}) \delta \bar{\rho} + \left(J_{u,\phi_1} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\phi_1} \right) \delta \phi_1 + \\
& \left(J_{u,\phi_2} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\phi_2} \right) \delta \phi_2 + \overline{J_{u,\alpha}} \delta \bar{\alpha} = 0
\end{aligned} \tag{A-136}$$

Into Eq. (A-133):

$$\begin{aligned}
& (J_{\alpha, \alpha} - J_{\alpha, V} J_{V, \alpha}) \delta \alpha + \left[J_{\alpha, P} - J_{\alpha, x_{nvg}} \left(J_{x_{nvg}, P} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, P} \right) - J_{\alpha, V} J_{V, P} \right] \delta P + \\
& \left[J_{\alpha, \rho} - J_{\alpha, x_{nvg}} \left(J_{x_{nvg}, \rho} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \rho} \right) \right] \delta \rho - \\
& J_{\alpha, x_{nvg}} \left(\overline{J_{x_{nvg}, \rho}} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}} \right) \delta \bar{\rho} + \\
& \left[J_{\alpha, u} + J_{\alpha, x_{nvg}} \left(J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, u} \right) \right] \delta u + J_{\alpha, x_{nvg}} J_{x_{nvg}, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}} \delta \bar{u} - \\
& J_{\alpha, x_{nvg}} \left(J_{x_{nvg}, \phi_1} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right) \delta \phi_1 - \\
& J_{\alpha, x_{nvg}} \left(J_{x_{nvg}, \phi_2} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right) \delta \phi_2 = 0
\end{aligned} \tag{A-137}$$

The axial coupling which exists in the remaining equations prevents us from reducing the equations further. The remaining system of equations can be written as follow.

The Pressure component:

$$\begin{aligned}
& J_{22}^{P, P} \delta P + \overline{J_{22}^{P, P}} \delta \bar{P} + J_{22}^{P, u} \delta u + \overline{J_{22}^{P, u}} \delta \bar{u} + J_{22}^{P, \rho} \delta \rho + \\
& \overline{J_{22}^{P, \rho}} \delta \bar{\rho} + J_{21}^{P, \phi_1} \delta \phi_1 + J_{21}^{P, \phi_2} \delta \phi_2 = 0
\end{aligned} , \tag{A-138}$$

where:

$$J_{22}^{P, P} = J_{P, P} - J_{P, x_{nvg}} \left(J_{x_{nvg}, P} - J_{x_{nvg}, Z} J_{Z, Z}^{-1} J_{Z, P} \right)$$

$$\overline{J_{22}^{P, P}} = -1.0$$

$$J_{22}^{P,u} = J_{P,u} + J_{P,x_{nvq}} J_{x_{nvq},Z} J_{Z,Z}^{-1} J_{Z,u}$$

$$\overline{J_{22}^{P,u}} = J_{P,x_{nvq}} J_{x_{nvq},Z} J_{Z,Z}^{-1} \overline{J_{Z,u}}$$

$$J_{22}^{P,\rho} = J_{P,\rho} - J_{P,x_{nvq}} \left(J_{x_{nvq},\rho} - J_{x_{nvq},Z} J_{Z,Z}^{-1} J_{Z,\rho} \right)$$

$$\overline{J_{22}^{P,\rho}} = \overline{J_{P,\rho}} - J_{P,x_{nvq}} \left(\overline{J_{x_{nvq},\rho}} - J_{x_{nvq},Z} J_{Z,Z}^{-1} \overline{J_{Z,\rho}} \right)$$

$$J_{21}^{P,\phi_1} = -J_{P,x_{nvq}} \left(J_{x_{nvq},\phi_1} - J_{x_{nvq},Z} J_{Z,Z}^{-1} J_{Z,\phi_1} \right)$$

$$J_{21}^{P,\phi_2} = -J_{P,x_{nvq}} \left(J_{x_{nvq},\phi_2} - J_{x_{nvq},Z} J_{Z,Z}^{-1} J_{Z,\phi_2} \right)$$

The Density component:

$$J_{22}^{\rho,P} \delta P + J_{22}^{\rho,\rho} \delta \rho + J_{22}^{\rho,u} \delta u + J_{22}^{\rho,\alpha} \delta \alpha = 0, \quad (\text{A-139})$$

where:

$$J_{22}^{\rho,P} = J_{\rho,P}$$

$$J_{22}^{\rho,\rho} = 1.0$$

$$J_{22}^{\rho,u} = J_{\rho,u}$$

$$J_{22}^{\rho,\alpha} = J_{\rho,\alpha}$$

The Internal Energy component:

$$\begin{aligned}
& J_{22}^{u,P} \delta P + J_{22}^{u,\rho} \delta \rho + \overline{J_{22}^{u,\rho}} \delta \bar{\rho} + J_{22}^{u,u} \delta u + \overline{J_{22}^{u,u}} \delta \bar{u} \\
& J_{22}^{u,\alpha} \delta \alpha + \overline{J_{22}^{u,\alpha}} \delta \bar{\alpha} + J_{21}^{u,\phi_1} \delta \phi_1 + J_{21}^{u,\phi_2} \delta \phi_2 = 0,
\end{aligned} \tag{A-140}$$

where:

$$J_{22}^{u,P} = J_{u,P} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,P} - J_{u,V} J_{V,P}$$

$$J_{22}^{u,\rho} = J_{u,\rho} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\rho}$$

$$\overline{J_{22}^{u,\rho}} = \overline{J_{u,\rho}} - J_{u,Z} J_{Z,Z}^{-1} \overline{J_{Z,\rho}}$$

$$J_{22}^{u,u} = 1 - J_{u,Z} J_{Z,Z}^{-1} J_{Z,u}$$

$$\overline{J_{22}^{u,u}} = \overline{J_{u,u}} - J_{u,Z} J_{Z,Z}^{-1} \overline{J_{Z,u}}$$

$$J_{22}^{u,\alpha} = J_{u,\alpha} - J_{u,V} J_{V,\alpha}$$

$$\overline{J_{22}^{u,\alpha}} = \overline{J_{u,\alpha}}$$

$$J_{21}^{u,\phi_1} = J_{u,\phi_1} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\phi_1}$$

$$J_{21}^{u,\phi_2} = J_{u,\phi_2} - J_{u,Z} J_{Z,Z}^{-1} J_{Z,\phi_2}$$

The Void Fraction component

$$J_{22}^{\alpha, P} \delta P + J_{22}^{\alpha, \rho} \delta \rho + \overline{J_{22}^{\alpha, \rho}} \delta \bar{\rho} + J_{22}^{\alpha, u} \delta u + \overline{J_{22}^{\alpha, u}} \delta \bar{u} \\ J_{22}^{\alpha, \alpha} \delta \alpha + J_{21}^{\alpha, \phi_1} \delta \phi_1 + J_{21}^{\alpha, \phi_2} \delta \phi_2 = 0 \quad , \quad (\text{A-141})$$

where:

$$J_{22}^{\alpha, P} = J_{\alpha, P} - J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, P} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, P} \right) - J_{\alpha, V} J_{V, P}$$

$$J_{22}^{\alpha, \rho} = J_{\alpha, \rho} - J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \rho} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \rho} \right)$$

$$\overline{J_{22}^{\alpha, \rho}} = -J_{\alpha, x_{nvq}} \left(\overline{J_{x_{nvq}, \rho}} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, \rho}} \right)$$

$$J_{22}^{\alpha, u} = J_{\alpha, u} + J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, u} \right)$$

$$\overline{J_{22}^{\alpha, u}} = J_{\alpha, x_{nvq}} J_{x_{nvq}, Z} J_{Z, Z}^{-1} \overline{J_{Z, u}}$$

$$J_{22}^{\alpha, \alpha} = J_{\alpha, \alpha} - J_{\alpha, V} J_{V, \alpha}$$

$$J_{21}^{\alpha, \phi_1} = -J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \phi_1} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_1} \right)$$

$$J_{21}^{\alpha, \phi_2} = -J_{\alpha, x_{nvq}} \left(J_{x_{nvq}, \phi_2} - J_{x_{nvq}, Z} J_{Z, Z}^{-1} J_{Z, \phi_2} \right)$$

The thermal hydraulics equations are now reduced to be just in terms of the pressure (Eq. (A-138)), moderator density (Eq. (A-139)), internal energy (Eq. (A-140)),

and void fraction (Eq. (A-141)). Note that the notations J_{21} and J_{22} indicate the block of the Jacobian matrix system where these sets of coefficients will be utilized. The components of the Jacobian matrix given in this derivation include both single phase and two phase regions. Therefore, one must be careful in assigning the appropriate components for a particular region (node).

The Jacobian of the power constraint

$$\begin{aligned} \sum_{n=1}^N \sum_{g=1}^G \kappa \Sigma_{f,g,n} \delta \phi_{g,n} + \sum_{n=1}^N \sum_{g=1}^G \frac{\partial}{\partial \rho_{avg}} (\kappa \Sigma_{f,g,n}) \phi_{g,n} \delta \rho_{avg} + \\ \sum_{n=1}^N \sum_{g=1}^G \frac{\partial}{\partial Z} (\kappa \Sigma_{f,g,n}) \phi_{g,n} \delta Z = 0 \end{aligned} \quad (A-142)$$

Substituting the expression for δZ from solving Eq. (A-76), we obtain:

$$\begin{aligned}
& \sum_{n=1}^N \sum_{g=1}^G \left[\kappa \Sigma_{f_{g,n}} + \frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial \phi_{g,n}} \right) \right] \delta \phi_{g,n} + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial P_n} \right) \right] \delta P + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial \rho_n} \right) \right] \delta \rho + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial \bar{\rho}_n} \right) \right] \delta \bar{\rho} + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial u_n} \right) \right] \delta u + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial Z} (\kappa \Sigma_{f_{g,n}}) \phi_{g,n} \left(\frac{\partial T}{\partial Z} \right)_n^{-1} \left(\frac{\partial T_n}{\partial \bar{u}_n} \right) \right] \delta \bar{u} + \\
& \sum_{n=1}^N \sum_{g=1}^G \left[\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \frac{\phi_{g,n}}{2} (\rho_g - \rho_l) [1 + cGA(\beta_1 + \beta_2[\alpha + \bar{\alpha}])] \right] \delta \alpha + \\
& \sum_{n=1}^N \sum_{g=1}^G \left(\frac{\partial}{\partial \rho_{avg}} \kappa \Sigma_f \frac{\phi_{g,n}}{2} \left\{ (\rho_g - \rho_l) [cGA(\beta_1 + \beta_2[\alpha + \bar{\alpha}])] + \right. \right. \\
& \left. \left. (\bar{\rho}_g - \bar{\rho}_l) [1 + cGA(\bar{\beta}_1 + \bar{\beta}_2(\bar{\alpha} + \bar{\bar{\alpha}}))] \right\} \right) \delta \bar{\alpha} = 0
\end{aligned}
\tag{A-143}$$