Chapter 4

Development of iteration schemes and an associated higher-order module

4.1 Introduction

The previous chapter concluded with the definition of a so-called general-order Consistent Leakage Approximation (CLA). If the second order expansion of this approximation is used, the method is termed CQLA (Consistent Quadratic Leakage Approximation) and represents the proposed replacement for the Standard Quadratic Leakage Approximation (SQLA) utilized in almost all transversely-integrated nodal methods today. It was shown how the hierarchical construction of higher-order node-averaged flux moments could make the method scalable to calculate a Full Higher-Order (FHO) solution if needed.

The number of unknowns per node for a CQLA is 15 in 3D; if we consider that the ANM has four unknowns per node, it is foreseen that significant further improvement to this efficiency level is needed. This chapter describes a number of iteration schemes which aim at making the CQLA method practical. Prior to proposing these schemes, it is useful to overview the basic architecture of the proposed higher-order module, its interfaces with standard nodal codes and the algorithmic layout. Within this context, three independent, but synergistic solution schemes are presented. Finally, these schemes are integrated in a proposed solution strategy.

4.2 Basic Driver Code and Module Description

Within the scope of this work a 3D Cartesian geometry multigroup nodal code was developed within which various methods could be tested. The nodal code employs the Semi-Analytic Nodal Method (SANM), which implies that transversely-integrated

one-dimensional equations are solved analytically, but that fission and scattering sources are treated as inhomogeneous sources and expressed in terms of flux moment expansions up to a predefined source order I. The SANM equations were derived and discussed in Section 2.3.2. This standard nodal code is referred to as SANS (Standard Analytic Nodal Solver) and has the following characteristics:

- 3D multigroup solver with node-averaged fluxes as primary unknowns;
- fixed multigroup cross-sections are read for an arbitrary number of energy groups;
- outer fission source iterations are accelerated via fission source extrapolation and up-scatter iterations are accelerated via the Aitken δ^2 extrapolation method (δ^2 - process)(Abramowitz and Stegun, 1970);
- one-dimensional sources are expanded up to an arbitrary Legendre polynomial order;
- node-averaged fluxes are determined via a Gauss-Seidel iteration scheme, with a fixed SOR (Varga, 1965) parameter (set to 1.3 as the typical value); and
- multiple leakage approximations are implemented, including constant, SQLA, CQLA (via higher-order module) and full higher-order approximations , via the higher-order module.

The developed higher-order nodal module primarily integrates with any zero-order nodal code at the transverse leakage update level and solves the higher-order one-dimensional equations at every interface in the system. The module is referred to as HOTR (Higher-Order Transverse leakage and Reconstruction) and has the following characteristics:

- primarily exchanges information with the low order solver at the transverse leakage level;
- requires a description of the system geometry as well as the macroscopic nodal cross-sections;
- applies the Semi-Analytic Nodal Method to solve two-node problems at every interface in the system, for each required higher-order equation; and

- solves various subsets of the higher-order equations, with the primary options being CQLA and FHO. Flat and SQLA are also implemented in the module which implies that the driver code no longer requires any transverse leakage option.

4.2.1 Module interfaces and layout

The communication between any transversely-integrated nodal code and the developed higher-order module, is depicted in Figure 4.1. In the figure, parallelograms indicate data sources and sinks and raised rectangles refer to code components. Arrows show data flows, with dotted lined arrows referring to optional flows.

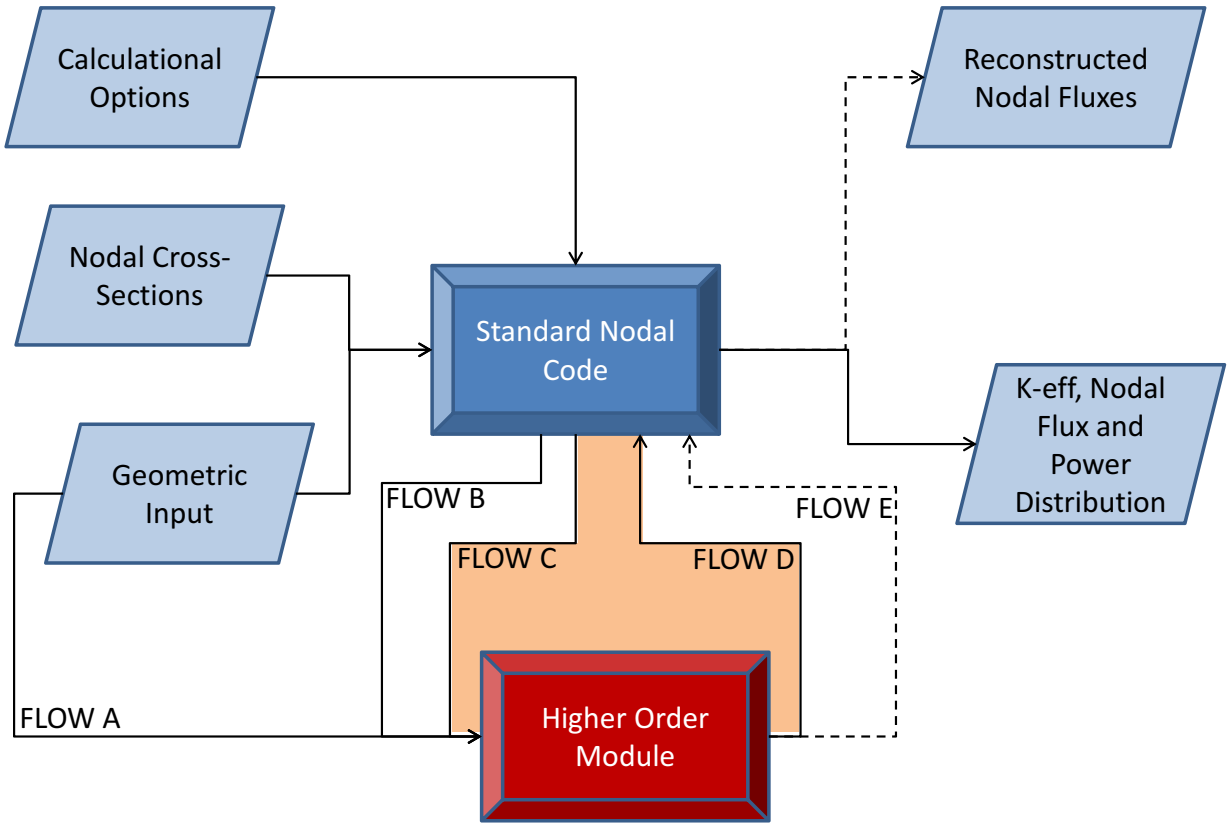

Figure 4.1: Schematic layout of the higher-order code interface with lower-order codes.

In Figure 4.1, the shaded area between the higher-order module and the standard nodal code refers to the within-iteration communication loop. The most important data flows are labeled and Table 4.1 discusses each of these flows in detail.

Table 4.1: Description of important data flows between a standard nodal code and the higher-order module.

Data flow	**Iteration level**	**Description**
FLOW A	Initialization	System geometry as well as selected calculational options is passed to HOTR.
FLOW B	Cross-section feedback level	If macroscopic cross-sections are updated in the nodal code, an update should be provided to HOTR.
FLOW C	Transverse leakage update level	Node-averaged fluxes and side-averaged currents are passed as a minimum. If side-averaged fluxes and flux moments are available, they are also passed.
FLOW D	Transverse leakage update level	Transverse leakage moments for every direction and every group, up to the chosen order, are returned.
FLOW E	Upon convergence	If required, reconstructed fluxes are returned.

The HOTR module is designed to operate independently of the solution strategy of the driver code. Thus it is, in principle, independent of the implementation of the driver code regarding:

- whether node-averaged flux or side-averaged currents are primary unknowns, as long as both may be recovered from one-another in the driver code. Zero-order side-averaged fluxes and one-dimensional flux moments are required by HOTR, but if these are not available, they are calculated by HOTR as needed. This incurs a small unwanted computational penalty, since these are actually standard nodal code quantities, but the option is added since not all the standard nodal codes necessarily collect these terms;

- whether a group-by-group or full multigroup solution is used, since HOTR can optionally generate transverse leakage moments only for the group under consideration, or for all energy groups, depending on the driver code requirements; and

- which acceleration and iteration schemes are used, be it via standard fission source iterative methods or more efficient schemes such as the so-called transverse leakage source iterative method (Vogel (1993)). This is possible since HOTR receives, iteration per iteration, the full set of system-wide node-averaged fluxes and side-averaged currents and thus the higher-order moments constructed

from these quantities yield a higher-order iteration scheme which conserves the low order solution at all times.

4.3 Algorithmic Layout

The solution scheme of SANS is laid out in Algorithm 1. Note that the code is developed with a flexible, recursive based iteration system, which means that iteration levels and thus the order of nesting of different iterative levels are configurable via code input. The algorithms presented here thus show a typical order of iterative layers, as used in the reported analysis.

Algorithm 1: SANS solution algorithm

```
      Input: Nodal cross-sections and problem geometry
      Result: Eigenvalue, nodal flux and power distribution
      initialization
 1    while Outer fission source iterations not converged do
 2        while Upscatter iterations not converged do
 3            for g ← 1 to numgroups do
                  Calculate nodal scattering source in group g
                  Calculate nodal scattering source moments in group g
 4                while Transverse leakage iterations not converged do
HOTR call 1           Calculate transverse leakage source
                      Calculate nodal source
                      Calculate nodal source moments
                      Calculate one-dimensional particular solution
                      Calculate tensorial sources
 5                    while Spatial flux distribution not converged do
                          Perform Gauss Seidell iterations
                      end
                      Update side-averaged fluxes and side-averaged currents
                      Update one-dimensional flux moments
                  end
              end
              if asymptotic up-scatter source behaviour then
                  perform up-scatter source acceleration
              end
          end
          Calculate nodal fission source
          Calculate nodal fission source moments
          Calculate k-eff
          if asymptotic fission source behaviour then
              perform fission source extrapolation
          end
      end
      Perform power normalization
HOTR call 2 Perform flux reconstruction
```

Algorithm 1 describes the overview order of calculations in SANS, with the numbering indicating the nested iteration levels. The point at which the external transverse leakage module is to be called from is labeled **HOTR call 1** and the reconstruction option is called from the point labeled **HOTR call 2**.

Prior to presenting the algorithmic layout of HOTR, a number of specific iteration schemes are discussed in the upcoming section. These schemes aim to:

- retain the design requirements that HOTR remains largely independent of the solution method of the driver code;
- retain the accuracy of the underlying higher-order solution (be it CQLA or FHO); and
- improve the efficiency of the solution by at least a factor of 2.

After discussion of these schemes, the chapter is concluded with the algorithmic layout of HOTR.

4.4 Proposed Iteration Schemes

In order to achieve the target efficiency and maintain sufficient accuracy, a number of iteration schemes are conceived for the described module. These approaches generally rely on hybrid low and higher-order strategies and are classified as:

- Partial convergence of the higher-order leakage shape (PLC);
- Higher-order Model Reduction (MR); and
- SQLA Correction (QLAC).

In the following subsections, each of these approaches are discussed, followed by a description of an integrated strategy, where these schemes are implemented synergistically. This integrated scheme is termed the Reduced Leakage Correction Scheme, or RLCS and represents the preferred iteration scheme proposed in this work. RLCS, when combined with CQLA (RLCS is applicable to any higher-order solution), would then be the default operational mode of the higher-order module, when utilizing it for the sake of providing leakage coefficients to any given transversely-integrated nodal code.

4.4.1 Partial convergence of the leakage shape

In the PLC iteration scheme, standard SQLA is solved until a set convergence is reached, at which point CQLA (or an alternative higher-order mode) is activated. CQLA is solved until the level of convergence (denoted by ε) reaches a second set point (e.g. a factor 10 greater than target convergence), after which the leakage shapes are no longer updated. Higher-order equations are then no longer solved

and a significant number of outer iterations may proceed via the standard lower-order scheme. In this phase no leakage method is needed. This approach, although simple and somewhat crude, has value since the higher-order moments often take a large number of outer iterations to fully converge, although a significant accuracy improvement, as compared to the SQLA approach, is already achieved much earlier in the iteration process. In PLC, the relevant parameters are given in Table 4.2.

Table 4.2: PLC specific iteration parameters.

Parameter	**Typical value**	**Description**
ε	N/A	Current iteration node-averaged flux convergence criteria
$\varepsilon_{\max}$	10^{-6}	Target maximum node-averaged flux convergence criteria
ε_{ho}	1000	Activate higher-order calculation when $\varepsilon < \varepsilon_{\max} \times \varepsilon_{\text{ho}}$
ε_{plc}	50	De-activate higher-order calculation when $\varepsilon < \varepsilon_{\max} \times \varepsilon_{\text{plc}}$

4.4.2 Model reduction

It is clear that not all the higher-order moments would be equally important in each node and each group in the system. A technique known as the Analysis of Variance (or ANOVA analysis) (An and Owen, 2001; Lemieux and Owen, 2002) may be applied to gauge the importance of the various modes in each node at various steps during the iteration process. If a given mode is found to be unimportant (as compared to some set criteria) in a given node and group, it may be excluded from the calculation, and thus effectively produces a local node and group wise reduction of higher-order equations.

ANOVA theory states that if a multivariate function on a rectangular domain is represented as a tensor product of orthonormal uni-variate basis functions, then the contribution of each of the terms to the variation of the function can be represented by the square of its coefficient. Hence, if we rewrite the expression for the full intra-nodal flux expansion by expanding the semi-moments up to the method order, we have an expression for the intra-node flux distribution on which the ANOVA analysis may be directly applied, and thus we determine the relative contribution of each flux moment to the shape of the intra-nodal flux.

This powerful tool allows that a threshold criteria may be placed on any given expansion moment and that based on the set criteria a node- and group-wise model reduction approach may easily be applied. It is proposed that an analysis of all the contributions is performed a number of times during the iteration process and based

on the result, a node and group specific list of unimportant moments are assembled and subsequently no longer calculated. The efficiency of this approach is expected to be problem dependent, and by the nature of ANOVA, will be most efficient when the approximation space is large – hence the efficiency of the scheme will grow as the dimensionality, order of expansion and number of groups grow.

To proceed, we consider the higher-order intra-nodal trial function, given by eq. (3.2) and expand one-dimensional semi-moments of the flux into a Legendre polynomial expansion up to the method order M to yield:

$$\widetilde{\phi}(u,v,w) = \sum_{l=0}^{M}\sum_{k=0}^{M}\sum_{m=0}^{M} f_{lk,m}P_m(\frac{2u}{h_u})P_l(\frac{2v}{h_v})P_k(\frac{2w}{h_w})+ \tag{4.1}$$

$$\sum_{m=0}^{M}\sum_{k=0}^{M}\sum_{l=0}^{M} g_{km,l}P_l(\frac{2v}{h_v})P_k(\frac{2w}{h_w})P_m(\frac{2u}{h_u})+$$

$$\sum_{m=0}^{M}\sum_{l=0}^{M}\sum_{k=0}^{M} h_{ml,k}P_w(\frac{2w}{h_w})P_m(\frac{2u}{h_u})P_l(\frac{2v}{h_v}) - 2\sum_{k=0}^{M}\sum_{l=0}^{M}\sum_{m=0}^{M} c_{lkm}P_m(\frac{2u}{h_u})P_l(\frac{2v}{h_v})P_k(\frac{2w}{h_w}).$$

Note we refer here to the approximate intra-nodal flux $\widetilde{\phi}(u,v,w)$ since the analytic form of the semi-moments is additionally expanded in Legendre polynomials. We group like terms and write

$$\widetilde{\phi}(u,v,w) = \sum_{l=0}^{M}\sum_{k=0}^{M}\sum_{m=0}^{M} \beta_{klm}P_m(\frac{2u}{h_u})P_l(\frac{2v}{h_v})P_k(\frac{2w}{h_w}) \tag{4.2}$$

with

$$\beta_{klm} = f_{lk,m} + g_{km,l} + h_{ml,k} - 2c_{lkm}$$

in order to express the intra-nodal flux as a set of orthogonal basis functions with coefficients β_{klm}. ANOVA then provides us with the fact that that the total function variance is given by

$$\mathrm{Var}\left(\widetilde{\phi}(u,v,w)\right) = \sum_{l=0}^{M}\sum_{k=0}^{M}\sum_{m=0}^{M} \frac{1}{(2k+1)(2l+1)(2m+1)}\left(\beta_{klm}\right)^2 \tag{4.3}$$

with at least one of $\{k,l,m\} > 0$ (the constant term $0,0,0$ is excluded from the variance). The individual contribution for a flux moment with indices $\{k,l,m\}$ to the total function variance is then given by

$$C_{klm} = \left(\frac{1}{(2k+1)(2l+1)(2m+1)} \right) \frac{(\beta_{klm})^2}{\mathrm{Var}\left(\widetilde{\phi}(u,v,w)\right)}. \tag{4.4}$$

This knowledge of C_{klm} for every node and every energy group in the system, denoted by $C^g_{n,klm}$ provides a powerful mechanism for rejecting terms per node and per energy group, based on some defined importance criteria C_{crit}. We aim to retain all the terms for which

$$C^g_{n,klm} > C_{\mathrm{crit}} \tag{4.5}$$

and thus only these "important" moments need to be calculated. Given that, according to eq. (3.17), the value of higher-order flux moments in neighbouring nodes are required to determine the current moments on the shared surface, the situation could arise where a given moment is rejected in node n, but retained in node $n+1$. In such cases, we choose to set the value of the "unimportant" moment to zero in the nodes in which they are rejected.

This strategy requires that the higher-order calculation, be it CQLA or a full higher-order of order M, is performed for a number of outer iterations, after which an estimation of the importance is performed and rejection implemented. This process may be repeated later on in the iteration process, but numerical analysis for typical problems have shown that further rejections, beyond the first one, do not contribute much to the calculational efficiency. This issue is more comprehensively discussed further in Chapter 5. Iteration parameters specific to the implementation of the model reduction schemes are listed in Table 4.3.

Table 4.3: Specific model reduction iteration parameters.

Parameter	**Typical value**	**Description**
$\varepsilon_{\mathrm{ho}}$	1000	Activate higher-order calculation when $\varepsilon < \varepsilon_{\max} \times \varepsilon_{\mathrm{ho}}$
$\varepsilon_{\mathrm{mr}}$	100	Perform model reduction when $\varepsilon < \varepsilon_{\max} \times \varepsilon_{\mathrm{mr}}$
$\varepsilon_{\max}$	10^{-6}	Target maximum node-averaged flux convergence criteria
C_{crit}	0.01	Disregard terms in node n and group g if $C^g_{n,klm} < C_{\mathrm{crit}}$

4.4.3 SQLA correction

SQLA works well for most problems and it is fair to expect that the aim could be to simply correct the shape where needed, whilst still applying the three-node fit as a solution algorithm. In this regard a scheme is developed, termed Quadratic Leakage

Approximation Correction (QLAC), where both SQLA and CQLA are solved simultaneously for a relatively small number of outer iterations, and then the correction factors are tabulated. These correction factors are defined for each node and energy group (index dropped here) for direction u, node n and order m of the leakage polynomial as

$$\zeta_{n,m}^{u,\text{qlac}} = \frac{\frac{1}{h_v}\left(J_{n,m0}^{wu,v}\left(+\frac{h_v}{2}\right) + J_{n,m0}^{wu,v}\left(-\frac{h_v}{2}\right)\right) + \frac{1}{h_w}\left(J_{n,0m}^{uv,w}\left(+\frac{h_w}{2}\right) + J_{n,0m}^{uv,w}\left(-\frac{h_w}{2}\right)\right)}{q_{n,m}^{u}} \tag{4.6}$$

where $J_{n,m0}^{wu,v}\left(\pm\frac{h_v}{2}\right)$ once again refers to the higher-order current moments on the $\pm$ surfaces in direction v of order $m0$ (analogous for direction w) and $q_{n,m}^{u}$ refers to the standard quadratic transverse leakage approximation coefficient of order m in direction u in node n. Note that the $q_{n,m}^{u}$ coefficients are produced from the fit over three neighbouring nodes and can be arbitrarily small or even zero. Clearly, we should ensure that the corrections are only applied if $\left|q_{n,m}^{u}\right| > \varepsilon_{\text{small}}$, where $\varepsilon_{\text{small}} = 10^{-6}$ is typical. These correction factors are found to converge much faster than the leakage shapes themselves, and thus the solution can proceed via a standard SQLA, augmented with higher-order correction factors which are frozen for the remainder of the calculation. Algorithmically stated, the following procedure is applied:

1. Standard SQLA is run up to a predefined convergence set point ε_{ho};

2. Higher-order (typically CQLA) and SQLA is run simultaneously and convergence of the ratios between the predicted CQLA and SQLA leakage coefficients are monitored and compared to a preset convergence criteria ($\varepsilon_{\text{qlac}}$). Typically this quantity need only to converge to 10^{-1} or 10^{-2} for good accuracy. It is important to state that during the simultaneous calculation of CQLA and SQLA, CQLA reports leakage coefficients to the nodal calculation, while SQLA is simply used for tabulation. Correction factors can clearly only be generated for the first and second moment of the leakage source, since the lower-order solution, in this case SQLA, can only represent shapes up to the second order;

3. Continue this process until the correction ratio has converged in a sufficiently large number of nodes and groups in the system, denoted by p_{qlac}. This additional criterion is needed given the fact that SQLA often suffers from convergence difficulties in isolated cases. In most cases no more than 10 - 15 higher-order iterations are needed to achieve convergence; and

4. Revert back to the standard SQLA, but in this case apply the correction factors obtained to each three-node fit in order to recover the accuracy of the higher-order solution $\left(q_{n,m}^{u,corr} = \zeta_{n,m}^{u,\text{qlac}} q_{n,m}^{u}\right)$. The correction factors are no longer updated, but the leakage source is still allowed to vary via SQLA as the iteration process continues.

The relevant iteration parameters for QLAC is listed in Table 4.4

Table 4.4: Specific SQLA correction iteration parameters.

Parameter	**Typical value**	**Description**
ε_{ho}	1000	Activate correction factor tabulation when $\varepsilon < \varepsilon_{\max} \times \varepsilon_{\text{ho}}$
$\zeta_{n,m}^{u,\text{qlac}}$	N/A	Node, group, direction and order dependent correction ratio
$\varepsilon_{\text{small}}$	10^{-6}	Numerical tolerance regarding value of SQLA coefficients
$\varepsilon_{\max}$	10^{-6}	Target maximum node-averaged flux convergence criteria
p_{qlac}	95%	Percentage of leakage moments to converge
$\varepsilon_{\text{qlac}}$	$10^{-2} - 10^{-3}$	Convergence criteria for maximum relative error in $\zeta_{\text{qlac}}^{d,m}$

4.4.4 An integrated strategy - The reduced leakage correction scheme

It is possible to integrate the three proposed schemes, as discussed in Sections 4.4.1 - 4.4.3, in order to optimize the efficiency of the higher-order solution further. In this optimal Reduced Leakage Correction Scheme (RLCS), the three schemes are integrated in the following manner:

1. Standard SQLA is run up to a predefined convergence set point ε_{ho};

2. Apply the QLAC scheme, still using p_{qlac} as the target percentage of leakage moments in the system for which the correction factors should converge. Two additional parameters are introduced namely f_{mr} and f_{plc};

3. f_{mr} refers to the fraction of p_{qlac} at which model reduction is activated. Thus when $f_{\text{mr}} \times p_{\text{qlac}}$ of the moments in the system have converged, model reduction is performed and the remaining higher-order calculations proceeded with the reduced set of unknowns;

4. f_{plc} refers to the fraction of p_{qlac} at which the partial leakage convergence is activated. Thus when $f_{\text{plc}} \times p_{\text{qlac}}$ of the moments in the system have converged,

higher-order moments are frozen, and the correction factor tabulation continues with only SQLA updates; and

5. The solution scheme reverts back to SQLA with static higher-order correction factors when p_{qlac} of the moments in the system have converged.

In Table 4.5 the additional parameters needed for RLCS are summarized and provide typical values for these parameters.

Table 4.5: Reduced leakage correction scheme iteration parameters.

Parameter	**Typical value**	**Description**
ε_{ho}	1000	Activate correction factor tabulation when $\varepsilon < \varepsilon_{\text{max}} \times \varepsilon_{\text{ho}}$
p_{qlac}	95%	Percentage of leakage moments to converge
f_{mr}	0.6	Fraction of p_{qlac} at which MR is activated
f_{plc}	0.8	Fraction of p_{qlac} at which PLC is activated

The RLCS approach is intended to minimize the number of higher-order transverse leakage iterations needed during nodal calculations. It is expected that this scheme would be highly efficient and accurate in the cases where the SQLA is smoothly convergent, but that p_{qlac} converged correction factors would not be achieved in a reasonable number of iterations for problems where SQLA diverges. In such cases, the tabulation of correction factors would terminate and only fully converged corrections would be used. In unconverged nodes, the usage of the last available higher-order moments proved more stable and accurate than utilizing SQLA.

Algorithm 2 describes the HOTR module, utilizing the RLCS scheme.

Algorithm 2: HOTR solution algorithm

Once-off input: *Geometry, boundary conditions, calc mode*
Feedback level input: *Nodal cross-sections*
Input: k-eff, Nodal fluxes, Average side-currents, energy group option
Optional Input: *Average side-fluxes, one-dimensional flux moments*
Result: Leakage coefficients, reconstructed fluxes
if *initialization* **then** process geometry
if *cross-section update* **then** calculate nodal coupling coefficients
if *energy group option* = *allgroups* **then**
  $g_i = 1$ and $g_f = numgroups$
else
  $g_i = g_f = inputgroup$
end
if *reconstruction* **then** `// converge ho moments in xy only`
  $outer\_iterations = max\_iter$
  $mode = FHO_n$
else `// single iteration for leakage update`
  $outer\_iterations = 1$ or 2
  if *RLCS converged* **then** `// correction factors already converged`
    Set *mode* to SQLA
    Perform three-node-fit + corrections
  else `// still require HO calculations`
    $mode = CQLA$
  end
end
HO iter **while** $mode \neq SQLA$ *and outers* $\leq outer\_iterations$ **do**
  1 **for** $g \leftarrow g_i$ **to** g_f **do**
  2 **forall the** *ho_equations* **do**
      Update HO scattering source moments: eq. (2.25)
      Update HO trans. leakage sources: eq. (A.30) and eq. (A.31)
      Update HO nodal source moments: Sect. (A.1.3)
      Update HO one-dimensional particular solution: eq. (A.26)
      Update HO tensorial sources: eq. (2.18)
      Update HO full flux moments: eq. (3.20)
      Update HO side-flux and current moments: eq. (3.17)
      Update HO one-dimensional flux moments: eq. (A.29)
    end
  end
  Update HO fission source moments: eq. (2.25)
  if *RLCS converged* (p_{qlac}) **then** `// SQLA correction factors converged`
    Revert back to SQLA mode with corrections
  else `// Continue CQLA and tabulation`
    Continue converging corrections: eq. (4.6)
    Perform model reduction if needed (f_{mr}): eq. (4.4)
    Perform HO freezing if needed (f_{plc})
  end
end

In Algorithm (2) the basic process followed in HOTR is described and where applicable reference is made to the related equations for the stated step in the algorithm. Algorithm (2) is followed every time the driver nodal code requires an update of the transverse leakage moments and is thus called from Algorithm (1) at the label marked **HOTR call 1**.

It can be seen that no explicit fission source, up-scatter or leakage iterations are present in HOTR when it is used for updating leakage coefficients, but rather that the higher-order sources (specifically fission and scattering) converge gradually as the driver nodal code converges. Either one or at most two higher-order iteration per call are sufficient for smooth convergence behaviour. The number of higher-order equations which are solved by HOTR in loop 3, depends on the solution mode and varies (in second order) from:

- None when in SQLA mode or when PLC (higher-order moment freezing) is applied;
- 15 when in CQLA mode or when tabulating RLCS correction factors. This number is potentially reduced during iterations by the model reduction procedure;
- 27 when in FHO_2 mode (full higher-order with expansion order 2); and
- $3(M+1)^2$ when in FHO_M mode (full higher-order with expansion order M).

Outer iterations are however performed when the module is called to perform flux reconstruction after the driver nodal code has converged, since the higher-order moments are present in FHO_M, but those which are not solved during CQLA, still need to converge.

4.5 HOTR as Flux Reconstruction Tool

As discussed in previous chapters, homogeneous flux reconstruction is a critical step in the standard deterministic calculational path. The step involves the reconstruction of fine-mesh diffusion fluxes based on some devised scheme dependent on the quantities standardly available from nodal codes. These include node-averaged fluxes, side-averaged currents and side-averaged fluxes. Often, in order to determine additional expansion coefficients that are needed for reconstruction, corner fluxes are also utilized, although these generally first have to be determined from an auxiliary interpolation step.

In the case of higher-order nodal methods, flux reconstruction is potentially a natural extension of the approach. This can be understood by recalling the trial function as given in eq. (3.2) in actual fact represents a direct 3D intra-nodal flux tabulation function and can be used as it stands in order to reproduce the 3D reconstructed flux. It is of course typical to perform flux reconstruction in 2D (since heterogeneous form factors exist only in 2D), in which case all higher-order moments in z may simply be neglected during the tabulation. An important acceleration in this regard is the fact that these spatial moments in z do not need to be resolved during the flux reconstruction calculation and thus the calculational time for the flux reconstruction solution is typically 30% faster than its associated full higher-order solution. Accuracy is thus a matter of selection and full second, fourth or sixth order flux reconstruction are options implemented in HOTR.

Reconstructed fluxes are further typically needed as averages on defined submeshes, in which case eq. (3.2) may simply be integrated over the desired mesh spacing. Although not the primary focus of this work, some examples of flux reconstruction via HOTR is presented in Chapter 5. The logic applied in HOTR for performing flux reconstruction is described in the list below:

1. The standard nodal calculation is performed as normal until convergence in eigenvalue and power distribution is reached. This calculation may be performed via utilizing HOTR for the sake of resolving the transverse leakage terms via either the SQLA, CQLA or full higher-order options. This implies that a given subset of the higher-order flux moments needed for flux reconstruction may thus already be known at convergence of the nodal solution, depending on the option selected;

2. Reconstruction mode is activated if requested, at which point HOTR performs a fixed source calculation with the zero-order quantities of eigenvalue, lower-order flux distribution and lower-order one-dimensional flux moments frozen to their obtained values. HOTR is placed in full higher-order mode up to the selected reconstruction method order (typically 2, 4 or 6). We reiterate that if e.g., the basic nodal calculation was performed via CQLA and the reconstruction is set to be performed to the second order, only the additional cross-term moments required for the full higher-order solution need to be resolved and thus results in a much faster reconstruction solution;

3. The fixed source calculation continues until the higher-order moments under consideration have converged to within the set convergence criteria (typically

10^{-3}). Note that no solution is needed in the z-direction since reconstruction is required in 2D. 3D reconstruction is however possible if required and naturally available within the method; and

4. Once all moments have converged, the intra-nodal trial function is utilized to tabulate fluxes or powers over the mesh spacing prescribed by the user. Two options are available, namely tabulated point fluxes or integrated submesh averages.

4.6 Conclusion

This chapter discussed three proposed iteration schemes built around the simplified higher-order solution described in Chapter 3. These schemes are termed the partial leakage convergence scheme, the model reduction scheme and the SQLA correction scheme. The three schemes are combined into an integrated solution strategy designed to achieve the required balance between accuracy and efficiency, and are built into a Fortran 95/2003 based higher-order code module. Both this code module (term HOTR), and a standard ANM based nodal solver designed for the work (termed SANS), are algorithmically described in this chapter. Nevertheless, an important feature of HOTR is that it is designed to function with almost all the transversely-integrated nodal codes, independent of the solution method, in order to resolve the difficulties with the treatment of the transverse leakage term, as well as to provide a consistent flux reconstruction.

In the upcoming chapter (Chapter 5) the performance and accuracy of HOTR, and thus the proposed solution and iteration schemes, are evaluated via a series of numerical benchmarks, spanning 2D, 3D, two-group and multi-group problems.