GenPMAXS

Code for Generating the PARCS Cross Section Interface File PMAXS

Y. Xu T. Downar

Purdue University
School of Nuclear Engineering

November, 2006

ABSTRACT

The Purdue Macroscopic cross section file PMAXS was developed in order to provide an interface between lattice physics codes and the PARCS core simulator. PMAXS provides all of the data necessary to perform core simulation for steady-state and transient applications, to include the principal macroscopic cross sections, the microscopic cross sections of Xe/Sm, the group-wise form functions with several different branch states for the appropriate fuel burnup states, and all of the appropriate kinetics data. The GenPMAXS program (Generation of the Purdue Macroscopic XS set) was written specifically to generate the PMAXS file from the macroscopic cross section libraries and the results of lattice codes such as HELIOS, TRITON, and CASMO. This manual describes the representation of the cross sections and the major methodologies employed in the PMAXS format, as well as input data for the GenPMAXS code and the results of the several benchmark problems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

1.1. Purpose

This manual describes the theory and procedure for generating PMAXS (Purdue Macroscopic cross section, XS) files which are used by PARCS[1] (Purdue Advanced Reactor Core Simulator) for core simulation and depletion analysis. The original PARCS cross section model which utilizes the XSEC block in the PARCS input has been retained as the default option. The new model described here uses PMAXS in lieu of XSEC and is initiated using 2 new cards in the PARCS CNTL input block (TREEXS and DEPLETION). This manual provides a detailed description of the methods for generating cross sections and the procedures used to execute the GENPMAXS/PARCS code sequence to perform core simulation and depletion analysis.

1.2. Overview

PARCS is a neutronics code for the prediction of nuclear reactor core steady-state and transient behavior at a specific core burnup state. PARCS solves the steady-state and time-dependent neutron diffusion and SP3 transport equations in three-dimensional geometry to obtain the neutron flux distribution. Because PARCS is capable of performing core eigenvalue calculations, as well as such functions as analyzing the control rod movement and searching for the critical boron concentrations, it has the functionality necessary to analyze both short term (kinetic) and longer term (depletion) core behavior. In order to provide depletion capability to the PARCS code, a depletion module was added to PARCS, as well as a new cross section module for retrieving node-wise cross section for its burunp history and current thermal-hydraulic state from PMAXS. The overview of PARCS package for core depletion analysis is shown in Figure 1.1. The depletion module generates new burnup and other history state information corresponding to the PARCS neutron flux solution. The cross section module calculates cross section based on burnup and other history state information, as well as on the current thermal-hydraulic state. The PARCS neutronic module then calculates the neutron flux with the cross sections generated by the cross section module.

Currently, PARCS employs a macroscopic depletion method in which the microscopic cross sections and the fuel number densities are not tracked individually during core depletion. Only the macroscopic cross sections are determined by the depletion module with burnup and history indices. The macroscopic cross sections at the appropriate fuel conditions are prepared using a lattice physics code such as HELIOS[3], CASMO[4], TRITON[5], etc. The PMAXS code is constructed using the output of the lattice code and provides the cross section data in the specific format that can be read by PARCS. Because the output format of the each lattice code is unique, the cross section processing program, GENPMAXS, was written to process the output of the lattice codes and prepare the PMAXS formatted cross section. An overview of the cross section generation scheme is shown in Figure 1.1. The GENPMAXS (Generation of the Purdue

XS set) code is therefore the interface between the lattice code and the depletion code. It processes the output of the lattice code and generates the PMAXS formatted files. More detailed guidelines for execution the GenPMAXS/PARCS code system are provided in Appendix B.

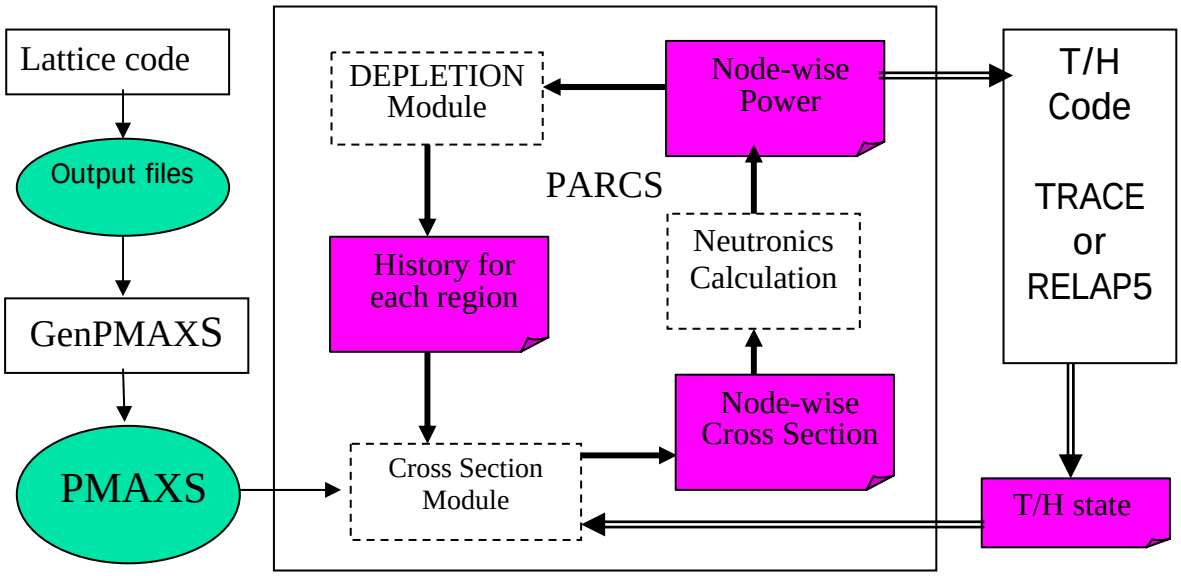


Figure 1.1 Overview of PARCS Code System for Core Depletion Analysis

2. Purdue Macroscopic Cross Section file, PMAXS

2.1. Cross Section Representation

PMAXS is structured in a macroscopic cross section format to be read by the PARCS depletion routine. It has been developed for representation of the macroscopic cross sections as a function of the state variables, such as the control rod poison (*CR*), Density of Coolant (DC), soluble Poison concentration in Coolant (PC), Temperature of fuel (TF), Temperature of Coolant (TC), Impurity of Coolant (IC) and also density, soluble poison concentration, temperature, impurity of the moderator (DM, TM, PM, IM). The cross sections are also functions of burnup (B) and other history variables. Since the macroscopic cross sections are strongly dependent on history effects, especially for the BWR, up to 5 history variables have also been introduced in the most recent version of the PMAXS format. The 5 history variables are control rod history (HCR), Coolant density history (HDC), Coolant soluble Poison history (HPC), fuel temperature history (HTF) or coolant temperature history (HTC). The history variables together with the fuel burnup determines the history state, $H=[h_1,\ldots,h_{nh,}\ B]$, where nh is number of history variables used in PMAXS. In the most recent version of GenPMAXS, the functionality was added to allow cross sections to depend on the conditions of the neighboring assemblies. This feature was added to improve the accuracy of flux solutions for heterogeneous core configurations, such as a checkerboard voided Advanced CANDU reactor. The differences of neighboring assembly coolant density (DN) and burnup (BN) from these of current nodes were introduced as two new variables in PARCS which are then used for the cross section feedback model. Although the neighbor's burnup is a history index, we treat both variables for neighboring conditions in the same manner as instantaneous variables.

Because the absorption cross sections of Xenon and Samarium are considerably larger than other isotopes and are strongly dependent on the flux level of each node, the absorption cross sections of the Xenon and Samarium are represented by their microscopic cross sections and number densities. The typical representation of the macroscopic cross section at a certain state is given by:

$$\Sigma^{l}(C,S,N,H)=\Sigma^{E,l}(C,S,N,H)+N_{Xe}^{l}\sigma_{Xe}^{l}(C,S,N,H)+N_{Sm}^{l}\sigma_{Sm}^{l}(C,S,N,H) \qquad (2.1)$$

where Σ^{E} is the macroscopic cross section, which does not include Xenon, Samarium The superscript *l* and the various subscripts denote the node index and isotope name respectively. To make use of these macroscopic cross sections, they must be determined systematically for each of the state variables. C, S, N and H are 4 sets of state variables.

C indicates the insertion fraction for each node and is provided for each control rod composition, $C=[c_1,..\ c_{Nc}]$, since there are multiple (Nc) control rod compositions in some BWR models (e.g. Ringhalls). If the c_i is flux weighted then the control rod effect is non-linear. If flux weighting is not available or not used, then the control rod effect is linear. Most standard codes used in the industry (e.g. CASMO/SIMULATE) employ

linear weighting. [Reference: Simulate User Manual, SOA-92/02, p. 2-33]. PARCS provides for flux weighting by solving a "3-node" problem with very fine mesh for the node with the partially inserted control rod. This is the standard method for treating the "control rod cusping" effect and is described in detail in the PARCS User Manual. Alternatively, for special cases where the standard method is not possible the user can provide for flux weighting using several branches for the control rod compositions in which different rod fractions are used to represent the non-linear effect of a partially inserted control rod in a node. The user can then employ simple volume fraction weighting and obtain the non–linear effect of a partially inserted control rod. In this treatment, PMAXS would contain branches for $0 < c \le 1$ for the first control rod composition and $c = 2,3,...$ for other control rod compositions if necessary. A more detailed explanation is provided in Section 2.2

S represents the state variables of the current nodes except for the control rod state, S=[s_2, s_3,..s_{Ns}]=[DC,PC,….]. Ns is the number of state variables for the current node. There are Ns-1 elements in vector S since the control rod state is also a state variable for current node and is treated separately. N contains 4 pairs of neighbor assembly information for the 4 adjacent assemblies in the plane N=[($n_{1,1}$ … $n_{1,Nn}$), ($n_{2,1}$ … $n_{2,Nn}$), ($n_{3,1}$ … $n_{3,Nn}$), ($n_{4,1,}$ … $n_{4,Nn}$),]. Nn is the number of state variables for the neighbor information and H=[h_1, …,$h_{nh,}$ B] represents the history state.

The macroscopic cross sections in PARCS are constructed with the assumption of a linear superposition of the partial cross sections on a base reference state. PMAXS supports all information which is required in representing the macroscopic cross-sections using this approximation as:

$$\Sigma(C,S,N,H) = c_{01}\Sigma(\frac{c_1}{c_{01}},S,N,H) + \sum_{i=2}^{Nc} c_i \Sigma(c_i,S,N,H) \tag{2.2}$$

where c_{01} is the sum of the unrodded fraction and the first composition fraction in the current node.

$$c_{01} = 1 - \sum_{i=2}^{Nc} c_i \tag{2.3}$$

$\Sigma(Cr,S,N,H)$ is determined by equation (2.4)

$$\begin{aligned}\Sigma(Cr,S,N,H) = \Sigma^r(H) + Cr\left.\frac{\partial\Sigma}{\partial Cr}\right|_{(Cr/2,H)} + \sum_{j=2}^{Ns} \Delta s_j \left.\frac{\partial\Sigma}{\partial s_j}\right|_{(Cr,S_j^m,N^r,H)} \\ + \sum_{j=1}^{4}\left(\sum_{k=1}^{Nn} n_{j,k}\left.\frac{\partial\Sigma}{\partial n_k}\right|_{\left(Cr,S,N_{j,k}^m,H\right)}\right)\end{aligned} \tag{2.4}$$

where a reference cross section Σ^r (first term on the right hand side) is modified with contributions from the control rod insertion, $Cr\left.\frac{\partial\Sigma}{\partial Cr}\right|_{(Cr,H)}$. For $Cr \le 1$, $\left.\frac{\partial\Sigma}{\partial Cr}\right|_{(Cr,H)}$, the partial derivatives can be obtained by interpolating between branches for the first control rod composition. When $Cr > 1$, CR must be an integer and the partial derivatives are obtained from the branch for control rod composition Cr. The contributions from the other independent variables are determined using the product of a partial cross section, $\partial\Sigma/\partial x$, and the amount of the perturbation for each independent variable:

$$\Delta s_j = s_j - s_j^r$$

As the state variables for neighbor information are defined as differences of the neighbor state values from the state values of the current node, the reference values for these variables are always zero, and therefore

$$\Delta n_{j,k} = n_{j,k}$$

The partial derivatives of cross sections are taken at the midpoint between the reference state and the current node state. These partials are obtained by piecewise interpolation of the pre-tabulated data using a "tree structure", which is discussed in the next section of the manual. This provides a second order accurate estimate of the cross section.

$$S_j^m = [s_2, s_3 ... s_{j-1}, (s_j + s_j^r)/2, s_{j+1}^r, ... s_{Ns}^r]$$
$$N_{j,k}^m = [n_{j,1}, ... n_{j,k-1}, n_{j,k}/2, 0 ... 0]$$

If there is only one type of control rod in each rod, then the cross section formulation can be simplified as:

$$\Sigma(c_1, S, N, H) = \Sigma^r(H) + c_1 \left.\frac{\partial\Sigma}{\partial Cr}\right|_{(c_1/2, H)} + \sum_{j=2}^{Ns} \Delta s_j \left.\frac{\partial\Sigma}{\partial s_j}\right|_{(c_1, S_j^m, N^r, H)} + \sum_{j=1}^{4} \left(\sum_{k=1}^{Nn} n_{j,k} \left.\frac{\partial\Sigma}{\partial n_k}\right|_{\left(c_1, S, N_{j,k}^m, H\right)} \right) \tag{2.5}$$

The microscopic cross sections of Xenon and Samarium have the same representation as Σ^E as in Eqs 2.2~2.5. Σ represents both Σ^E and the microscopic cross sections of Xenon and Samarium.

Each of the three groups of independent variables:

1. the control rod fractions
2. variables of the current node

3. variables of neighbors

are treated in a different manner and the user must select the appropriate independent variables for their problems.

The sequence of independent variables in Eqs. 2.2 to 2.5 is fixed and is very important for organizing data in the PMAXS file. The control rod fraction is always the first variable even if there is no control rod involved in the problem. The variables of neighbors will be the last variables if they are selected. In addition to the control rod fraction, the density of the coolant (DC), the soluble poison concentration in the coolant (PC), the temperature of the fuel (TF), and the temperature of the coolant (TC) are the four most often used variables. The sequence of these four variables was established based on the magnitude of the dependencies of the neutronic properties with respect to these variables. A study of these dependencies was performed in order to verify the optimum order for each of the variables. A pin cell model was analyzed with each combination of the parameters shown in Table 2.1.

Table 2.1. Parametrics for Cross Section Dependence Study

Independent Variable	Low value	Middle value	High value
DC(g/cc)	0.1	0.5	0.9
PC(ppm)	0	1000	2000
$\sqrt{Tf}$ ($\sqrt{K}$)	25	30	40
TC(K)	300	450	600

The partial derivatives of the cell k-infinite with respect to each independent variable were evaluated and the variations of the partial derivatives are shown in Table 2.2. The dependency of the partial derivatives of the k-infinite with respect to each variable to all of the independent variables is summaried in Table 2.3.

Table 2.2. Variation (%) of Kinf and its Partial Derivative for Each Variable

	Kinf	∂Kinf/∂DC	∂Kinf/∂PC	∂Kinf/∂$\sqrt{Tf}$	∂Kinf/∂TC
DC	39.4	74	70	24	148
PC	21.5	42	27	24	127
$\sqrt{Tf}$	4.4	6	8	8	70
TC	0.4	1	2	2	196

Table 2.3. Dependencies of Kinf and its Partial Derivative to Each Variable

	Kinf	∂Kinf/∂DC	∂Kinf/∂PC	∂Kinf/∂$\sqrt{Tf}$	∂Kinf/∂TC
DC	Very strong	Very strong	Very Strong	Mild	Very strong
PC	Strong	Strong	Mild	Mild	Very strong
$\sqrt{Tf}$	Mild	Weak	Weak	Weak	Strong
TC	Weak	Almost none	Almost none	Almost none	Very Strong

Other variables, such as the impurity of the coolant (IC), the density, the soluble poison concentration, the temperature, and the impurity of the moderator (DM, TM, PM, IM), were recently introduced for heavy water reactors. They should be placed after CR, DC, PC, TF, and TC, and before the neighbor variables. In summary, the 12 independent variables which are currently accepted by PMAXS and PARCS should be provided in the following order:

```
 1. CR control rod fraction
 2. DC density of coolant
 3. PC soluble poison concentration in coolant
 4. TF temperature of fuel
 5. TC temperature of coolant
 6. IC impurity of coolant
 7. DM density of moderator
 8. PM soluble poison concentration in moderator
 9. TM temperature of moderator
10. IC impurity of moderator
11. DN density difference between neighbor and current assembly
12. BN burnup difference between neighbor and current assembly
```

2.2. PMAXS Content

The PMAXS file contains all information which is required to implement Eqs. 2.2-2.5. The PMAXS format was developed with several basic principles in mind. First, all information is provided in a hierarchical system. For example, the branch information is provided first and then the cross sections are provided sequentially for each branch. The second principle is that all information which is necessary to allocate computer memory is provided first and then followed by the cross sections. Thus, by reading the branch information, it is possible to know how many branch cases are stored in this particular PMAXS file and to allocate the required amount of memory. The detail structure of the PMAXS format is given in Appendix A.

The cross section data, which includes the reference cross sections (Σ^r),the partial derivatives of cross sections with respect to the control rod fraction and other feedback variables ($\partial\Sigma/\partial Cr$, $\partial\Sigma/\partial DC$, $\partial\Sigma/\partial PC$, $\partial\Sigma/\partial\sqrt{Tf}$, $\partial\Sigma/\partial TC$, …), for all burnup points at the same history state are listed sequentially as branches of a history cases in PMAXS. One PMAXS file contains one or more history cases of an assembly. All history cases in

a PMAXS file have an identical branch structure, but there may be a different number of burnup points in the branches with different histories. The treatment of history and burnup dependencies will be described in detail in section 2.5.

2.3. Procedures for Generating Cross Section Data for All Branches

This section will describe the methods used to treat the dependence of the instantaneous variable dependence. The method for treating history variables will be described in section 2.5. In the treatment of the control rods it is assumed that there is at most 1 control rod composition in a branch, and for the compositions other than first compositon, it is either fully withdrawn or fully inserted in the node. The control rod fractions are then represented with a single rod fraction Cr. Cr=0 indicates that the branch is for the unrodded state, and $0 < Cr \le 1$ indicates that the branches with fraction Cr are for the first control rod composition and zero for all others Cr>1 indicates that the branches are for the fully inserted case with control rod composition Cr. Flexibility is provided to treat different types of control rods and control rods with heterogeneous compositions.

$$Cr = \begin{cases} c_1, if\ c_i = 0\ for\ all\ i > 1 \\ i,\ if\ c_i = 1, and\ c_j = 0\ for\ all\ j \neq i \end{cases} \tag{2.6}$$

2.3.1. Reference Branch

The cross sections at the reference state ($Cr^r = 0, S^r, N^r$) are directly stored in the reference branch.

2.3.2. Control Rod Branches

The states with reference parameters (S^r, N^r) but with control rods inserted are called control rod branches. The partial derivatives of cross section differences with respect to the rod fraction, Cr, are computed and stored.

$$\left.\frac{\partial \Sigma}{\partial Cr}\right|_{Cr/2} = \frac{\Sigma(Cr, S^r, N^r) - \Sigma^r(Cr^r, S^r, N^r)}{Cr} \tag{2.7}$$

Note that for the branches with Cr>1, equation (2.7) is not really the physical derivative of the cross section with respect to the rod fraction, but represents $1/Cr$ times the cross section difference due to the control rod composition Cr.

The only instantaneous variable which control rod dependence is the control rod state (Cr).

2.3.3. Branches for other variables

The partial derivatives with respect to other independent variables are stored in corresponding branches. The branch states are generated using the following process. First, a reference state is selected and the values of independent variables at this state are assigned as reference values. A new branch state is then generated by changing the value of one independent variable from the reference state or from any branch state previously generated by this process. The name of the independent variable which has been changed for generating this branch state is also used to name the type of branch. The changed variable is referred to as a branch variable for this type of branch. For example, if the density of the coolant is changed from the reference state or from a control rod branch state to generate a new branch state, then this new branch state is called the density branch of the coolant branch state, and the density of the coolant is the branch variable of the density of the coolant branch.

The branch states in each type of branch have reference values for all independent variables after the branch variable. A value different from the reference value is used for the branch variable. For each branch state i, there must be a state j which has a reference value for the branch variable of state i, and the values of all other independent variables are the same as those of state i. State i is generated by changing the value of one independent variable from state j. State j is called the base state of state i. The partial derivatives for the mid-point between branch state i and its base state j are stored in branch i. These partial derivatives are computed by second order central differencing:

$$\left.\frac{\partial \Sigma}{\partial x_k}\right|_{X^m} = \frac{\Sigma(X^i) - \Sigma(X^{B(i)})}{x_k^i - x_k^r} \qquad (2.8)$$

Where $X^i = (x_1^i, x_2^i, ...x_n^i)$ is the vector for the variables of state i, $X^{B(i)}$ is the vector for the variables of the base state of state i. X^m is the vector for the mid-point between state i and its base state. x_k is the branch variable of state i.

The values of the variables of state i and its base state satisfy the following relations:

$$x_m^i = x_m^{B(i)} = x_m^r \quad \text{for all m>k}$$
$$x_m^i = x_m^{B(i)} \quad \text{for all m<k}$$
$$x_k^{B(i)} = x_k^r$$

where x_m^r are the reference values for the state variables.

An illustrative example of the branch cases in PMAXS is shown in Figure 2.1. In this example there are 6 states "Ref", "Cr^1", "DC^1", "DC^2" , "DC^3" , "DC^4", at which cross sections are provided from the lattice code. In PMAXS, the original XS at the "Ref" state are stored as the Reference branch. The difference of the cross sections between the "Cr^1" state and the "Ref" state are stored in the control rod branch case. The partial derivatives with respect to the density of the coolant at the states "DC^{1m}", "DC^{2m}" , "DC^{3m}" and "DC^{4m}" are computed using Eq. 2.8 and the data from the 6 states "DC^1", "DC^2" , "DC^3" , "DC^4" and "Ref", "Cr^1". These partial derivates are then stored as 4 density branches of the coolant branch cases in PMAXS.

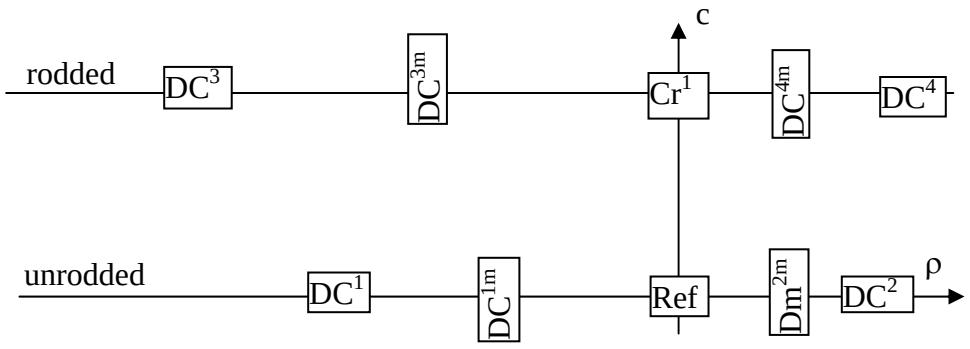


Figure 2.1. Example of Branch Cases in PMAXS

2.4. Tree structure and linear interpolation between branches

Although the branch cases have a multi-dimensional structure, they are listed sequentially as 1-dimensional cases in PMAXS with the branch information given at the beginning of each file. PARCS reads the branch information, allocates memory, and then constructs the "tree structure" for the partial derivatives. After the tree structure is constructed, the partial derivatives required in Eqs. 2.2-2.5 are then obtained by multi-dimensional linear interpolation in PARCS. Figure 2.2 illustrates the procedure for using Eq. 2.9 to compute a cross section for a specified state. In the example, the cross section is computed for point 1 at which (c, DC).

$$\Sigma(c, DC) = \Sigma^r + c\frac{\partial\Sigma}{\partial Cr}\bigg|_{(Cr^1/2)} + (DC - DC^r)\frac{\partial\Sigma}{\partial DC}\bigg|_{(c, DC^m)} \tag{2.9}$$

The first term on the right hand side of Eq. 2.9, Σ^r, is the cross section at the reference state. The second term, $c\frac{\partial\Sigma}{\partial Cr}\Big|_{(Cr^1/2)}$, is the contribution from insertion of the control rod and is depicted as point 2 in Figure 2.2. $\frac{\partial\Sigma}{\partial Cr}\Big|_{(Cr^1/2)}$ is computed as the difference between the 'Ref' state and the control rod insertion state, and is stored as a control rod

branch. The third term is the density of coolant term, and is computed as the difference between point 1 and point 2, $(DC - DC^r)\left.\frac{\partial \Sigma}{\partial DC}\right|_{(c, DC^m)}$. The partial derivative with respect to density of coolant at point 3, $\left.\frac{\partial \Sigma}{\partial DC}\right|_{(c, DC^m)}$, is obtained by linear interpolation between the partial derivatives with respect to coolant density at "DC^{1m}", "DC^{2m}", "DC^{3m}", "DC^{4m}", which are stored in 4 coolant density branches. The formula for computing the partial derivative is given by Eq. 2.10:

$$\left.\frac{\partial \Sigma}{\partial DC}\right|_{(c, DC^m)} = w_1 \left.\frac{\partial \Sigma}{\partial DC}\right|_{(0, DC^{1m})} + w_2 \left.\frac{\partial \Sigma}{\partial DC}\right|_{(0, DC^{2m})} + w_3 \left.\frac{\partial \Sigma}{\partial DC}\right|_{(1, DC^{3m})} + w_4 \left.\frac{\partial \Sigma}{\partial DC}\right|_{(1, DC^{4m})} \tag{2.10}$$

where the weights for the four points are determined by linear interpolation:

$$w_1 = (1-c)\frac{DC - DC^2}{DC^1 - DC^2} \qquad w_2 = (1-c)\left(1 - \frac{DC - DC^2}{DC^1 - DC^2}\right)$$
$$w_3 = c\frac{DC - DC^4}{DC^3 - DC^4} \qquad w_4 = c\left(1 - \frac{DC - DC^4}{DC^3 - DC^4}\right) \tag{2.11}$$

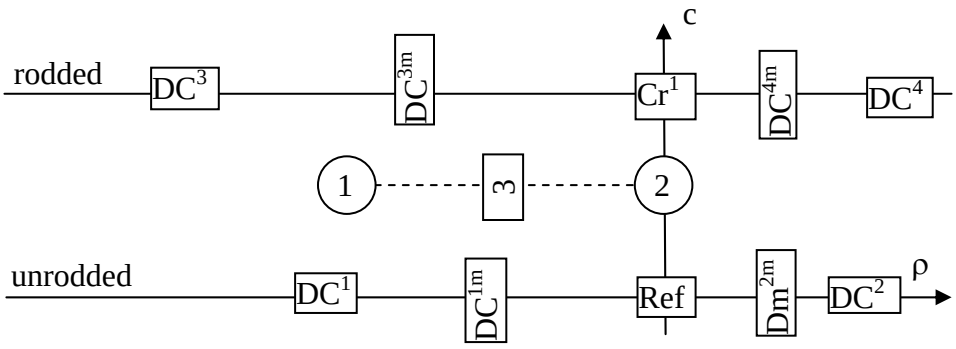


Figure 2.2. Example of Computing a Cross Section at Point 1

Even though the derivatives are defined at the mid-point, the state variable values for the mid-point are not used in the interpolation. Only the values of the original states are used, and therefore only the values of the original states are stored as branch

information in PMAXS. The other partial derivatives which are required in Eqs.2.2-2.5 are obtained by multi-dimensional linear interpolation using equations similar to Eq. 2.10 and 2.11. It should be noted that the branches do not need to form a regular grid, and the points in each linear interpolation do not need to be equidistant. This is why the method here is referred to as a 'tree' structure instead of a table.

2.5. Tree structure and linear interpolation for history states

The dependencies of the cross section data, which includes the cross sections at the reference branch and the partial derivatives at other branches, to the history variables (except burnup) are treated with partial derivatives with respect to these history variables.

$$\Sigma(\bar{H},B)=\Sigma(\bar{H}^r,B)+\sum_{j=2}^{nh}\Delta h_j\left.\frac{\partial\Sigma}{\partial h_j}\right|_{(\bar{H}_j^m,B)} \tag{2.12}$$

where Σ represents the cross sections at reference state and all partial derivations w.r.t. all instantaneous variables which are needed in equation (2.4). $\bar{H}=(h_1,...h_{nh})$ is the combination of all history variables except burnup, which represents the history state of interest. $\bar{H}^r=(h_1^r,...h_{nh}^r)$ is the reference history state and $\bar{H}_j^m=(h_1,...,h_{j-1},\left(h_j+h_j^r\right)/2,h_{j+1}^r,...,h_{nh}^r)$ is the middle point between $\bar{H}$ and $\bar{H}^r$ w.r.t. the jth history value: $\Delta h_j=h_j+h_j^r$.

The partial derivative w.r.t. the jth history variable, $\partial\Sigma/\partial h_j$ has a j-dimensional dependence to the 1st of the jth history variables. $\partial\Sigma/\partial h_j$ at history state $\bar{H}_j^m$ is obtained by multi-dimensional piece wise linear interpolation with $\partial\Sigma/\partial h_j$ at the neighbor history states. These $\partial\Sigma/\partial h_j$ at neighbor history states are computed implicitly with the cross section data Σ of the neighbor history cases.

The burnup dependence of the cross section data is treated by piece wise linear interpolation. The cross section data at burnup B is computed as:

$$\Sigma^i(\bar{H}^j,B)=\frac{B_k^{i,j}-B}{B_k^{i,j}-B_{k-1}^{i,j}}\Sigma^i(\bar{H}^j,B_{k-1}^{i,j})+\frac{B-B_{k-1}^{i,j}}{B_k^{i,j}-B_{k-1}^{i,j}}\Sigma^i(\bar{H}^j,B_k^{i,j}) \tag{2.13}$$

Where Σ^i represents the cross section data in ith branch, $\bar{H}^j$ is history state of jth history case, $B_k^{i,j}$ is first burnup point in ith branch of jth history case which be greater than B.

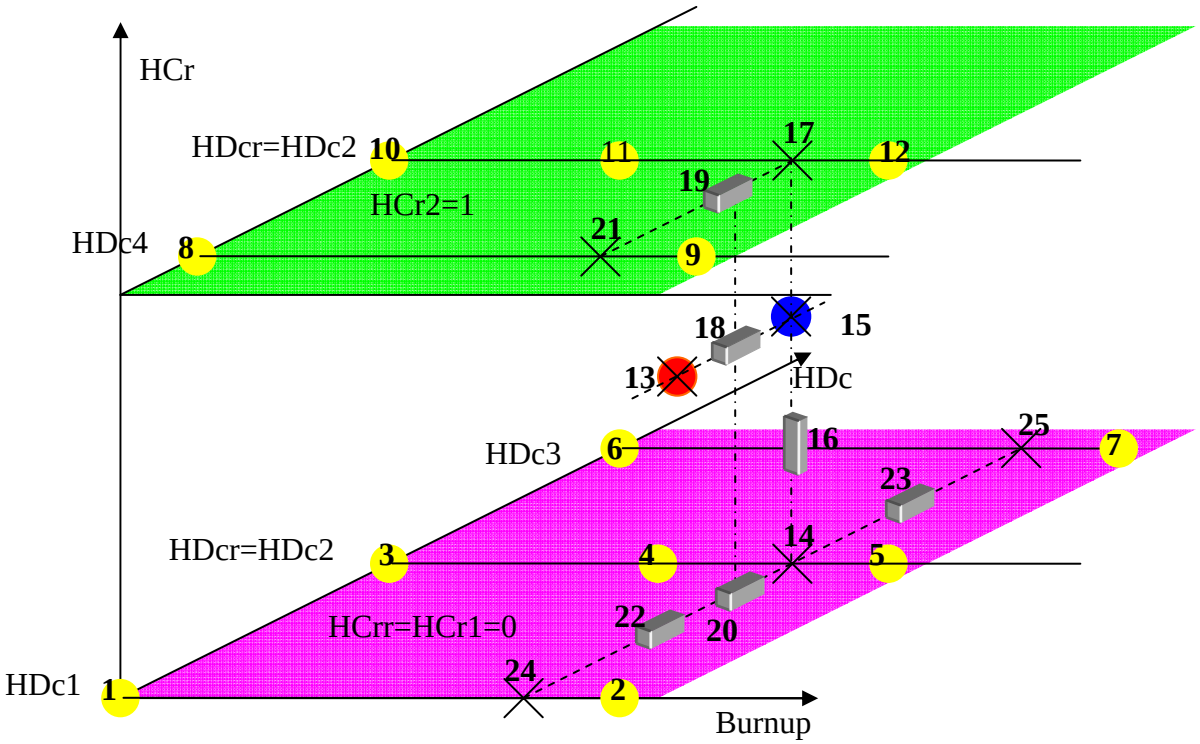


Figure 2.3. An Example for Computing History Dependence

An example of the method used to compute the history dependence of the XS in PMAXS is depicted in Figure 2.3. The locations of the history state point in PMAXS can be irregular, and are stored in a tree structure. In the example shown in Figure 2.3, HCR and HDC are taken as the first and second history variables. The fuel burnup is always the last history variable in the tree structure of PMAXS. 12 burnup points are cataloged into 5 history cases, 3,4,5 in reference history case, 10,11,12 in control rod history case, and the other points in the 3 coolant density history cases.

The cross section set at point 13 in the example shown in Figure 2.3 can be obtained using the following procedure:

1) The cross section data at point 14, which in reference history case and has same burnup as point 13, are computed by linear interpolation between point 4 and 5.

2) The cross section data at point 15, which has a control rod history the same as point 13, and the reference coolant density history are computed by adding differences to point 14. The partial derivatives with respect to control rod history at point 16 which are need for computing the difference between point 15 and 14 are determined by the cross section data at 14 and 17. The cross section data at 17 is then computed by linear interpolation between points 11 and 12.

3) The cross section data at point 13 are then obtained by adding the products of the coolant density history difference between point 13 and 15 and the partial derivatives w.r.t. coolant density history at point 18 which are obtained by linear interpolation between point 19 and 20. The partials at point 19 are determined by finite differencing between points 14 and 21, and the cross section data at point 21 are computed by linear interpolation between point 8 and 9. The partials at point 20 are computed by linear interpolation between the partials at 22 and 23 which require cross sections at 14, 24 and 25. The cross section data at point 1, 2 and 6, 7 are then used to calculate the cross sections at points 24 and 25, respectively, by linear interpolation.

Instead of explicitly applying linear interpolation or finite differencing to all cross section data, PARCS first determines the weights for 10 burnup points 1, 2, 4~9, 11,12, then performs a linear combination of the cross section data at these 10 burnup points for the cross section data of interest at point 13.

3. Special Treatment for Cross Sections in GenPMAXS

3.1. Scattering Cross Sections Treatment in the GenPMAXS Code

Most diffusion codes for analyzing light water reactors do not consider up-scattering because its effect may be negligible in the standard two group energy group solution. However, it is inaccurate to neglect upscattering in multi-group calculations and in most multigroup applications upscattering data is provided. However, in some cases the user may choose to treat upscattering implicitly for the multigroup calculation and that option is provided in GenPMAXS. Using the concept of conservation of neutron spectra, the down-scattering cross-sections can be modified to treat the up-scattering effect implicitly. In the GenPMAXS code, the down-scattering cross-sections can be corrected by using the P0-scattering cross section and flux data:

$$\Sigma'_{s,g\leftarrow g'} = \Sigma'_{s,g\leftarrow g'} - \Sigma_{s,g'\leftarrow g}\frac{\phi_g}{\phi_{g'}}, \text{ for } g' < g, \qquad (3.1)$$

where $\phi_g, \phi_{g'}$ are the spectra flux either provided by lattice code or obtain from infinite spectra computed in GenPMAXS.

3.2 Generation of Reflector Cross Sections

In order to conserve the neutron reaction rate at the interface between the core and reflectors, it is necessary to provide both the effective macroscopic cross sections of the homogeneous reflector and the assembly discontinuity factors (ADF). Codes such as CASMO-3, MASTER, ANS, etc., obtain the effective homogenized reflector cross sections including ADF by solving the one-dimensional spectral geometry problems as shown in Figure 3.1.

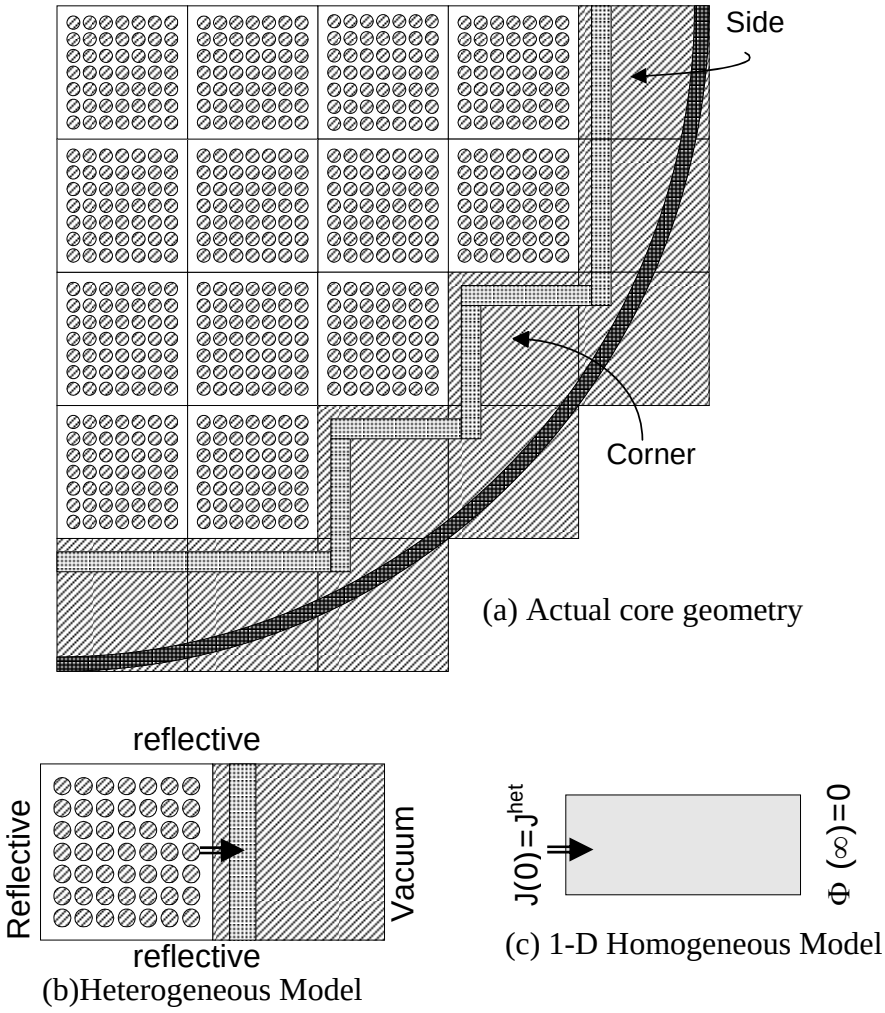


Figure 3.1. Reflector Model for Cross Section Generation

The effective homogeneous cross-sections are generated using the 1-D heterogeneous model of the reflector. However, because the surface averaged fluxes of the homogeneous reflector are not explicitly provided in most lattice calculation, an alternate method is used to determine the assembly discontinuity factors. The surface averaged fluxes of the homogeneous reflector are determined by solving the 1-D diffusion equations in the homogeneous reflector region. In the homogeneous reflector, the multi-group diffusion equations are given by;

$$\frac{\partial^2 \Phi}{\partial x^2} = D^{-1} M \Phi \qquad (3.2)$$

where $\Phi = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_g \end{bmatrix}$, $M = \begin{bmatrix} \Sigma_{r1} & -\Sigma_{1<-2} & \cdots & -\Sigma_{1<-g} \\ -\Sigma_{2<-1} & \Sigma_{r2} & \cdots & -\Sigma_{2<-g} \\ \vdots & \vdots & \ddots & \vdots \\ -\Sigma_{g<-1} & -\Sigma_{g<-2} & \cdots & \Sigma_{rg} \end{bmatrix}$, $D = diag\left(D_1, D_2 ... D_g\right)$

$$\Sigma_{ri} = \Sigma_{ai} - \Sigma_{i<-i} + \sum_{j=1}^{g} \Sigma_{j<-i}$$

$$A = D^{-1}M = \begin{bmatrix} D_1^{-1}\Sigma_{r1} & -D_1^{-1}\Sigma_{1<-2} & \cdots & -D_1^{-1}\Sigma_{1<-g} \\ -D_2^{-1}\Sigma_{2<-1} & D_2^{-1}\Sigma_{r2} & \cdots & -D_2^{-1}\Sigma_{2<-g} \\ \vdots & \vdots & \ddots & \vdots \\ -D_g^{-1}\Sigma_{g<-1} & -D_g^{-1}\Sigma_{g<-2} & \cdots & D_g^{-1}\Sigma_{rg} \end{bmatrix} \qquad (3.3)$$

Assume the matrix A is positive definite and diagonalizable, then there exists an invertible matrix V such that:

$$V^{-1}AV = \Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_g \end{bmatrix} = B^2 = \begin{bmatrix} B_1 & 0 & \cdots & 0 \\ 0 & B_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & B_g \end{bmatrix}^2 \qquad (3.4)$$

where $B_i = \sqrt{\lambda_i}$ (3.5)

Therefore Eq. 3.2 can be rewritten,

$$\frac{\partial^2 V^{-1}\Phi}{\partial x^2} = V^{-1}AVV^{-1}\Phi = \Lambda V^{-1}\Phi \tag{3.6}$$

Differential equation (3.7) has the analytic solution:

$$V^{-1}\Phi = e^{-Bx}C + e^{Bx}F \tag{3.7}$$

where C, F are determined by boundary conditions.

Using the right boundary condition:

$$\Phi(\infty) = 0 \tag{3.8}$$

provides

$$F = 0 \tag{3.9}$$

and

$$\Phi = Ve^{-Bx}C \tag{3.10}$$

Using the left boundary condition:

$$J(0) = -D\frac{\partial \Phi}{\partial x} = DVBe^{-B\times 0}C = (DVB)C \tag{3.11}$$

$$C = (DVB)^{-1}J(0) \tag{3.12}$$

provides the analytic solution

$$\Phi(x) = Ve^{-Bx}(DVB)^{-1}J(0) \tag{3.13}$$

And the homogenous flux at left surface

$$\Phi(0) = V(DVB)^{-1}J(0) \tag{3.14}$$

Therefore, the assembly discontinuity factors of the reflector can be given as:

$$ADF_k = \frac{\phi_k^{Het}(0)}{\phi_k(0)}, \quad k = 1,2,...g\ . \tag{3.15}$$

Since there is no fission in reflector, the matrix A is in block lower triangle form:

$$A=\begin{bmatrix} A_{1,1} & 0 & \cdots & 0 \\ A_{2,1} & A_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \cdots & A_{n,n} \end{bmatrix} \tag{3.16}$$

where $A_{i,i}$ is square matrix with order l_i. An efficient algorithm can be used for this problem which avoids computing all eigenvectors of matrix A. All vectors and matrices are presented in block form, as for example: $\Phi=[\phi_1,\phi_2,...,\phi_n]$, where ϕ_i is vector with length l_i. Assuming the matrix A is positive definite and diagonalizable, then there exists invertible matrices V_i, $i=1,n$, such that:

$$V_i^{-1}A_{i,i}V_i=\Lambda_i=B_i^2 \tag{3.17}$$

where B_i, $i=1,n$, are positive definite diagonal matrices.

Considering the right boundary condition, $\Phi(\infty)=0$, the analytic solution can then be express as:

$$\phi_k=\sum_{i=1}^{k}F_{k,i}V_ie^{-B_ix}C_i \tag{3.18}$$

where $F_{i,i}$ is identical matrix with order l_i.

Substituting (3.18) into (3.2):

$$\sum_{i=1}^{k}\left(F_{k,i}V_iB_i^2e^{-B_ix}C\right)_i=\sum_{i=1}^{k}\left(F_{k,i}A_{i,i}V_ie^{-B_ix}C_i\right)=\sum_{i=1}^{k}\left(A_{k,i}\sum_{j=1}^{i}\left(F_{i,j}V_je^{-B_jx}C_j\right)\right)$$

$$\sum_{j=1}^{k}\left(\left(F_{k,j}A_{j,j}-A_{k,k}F_{k,j}\right)V_je^{-B_jx}C_j\right)=\sum_{i=1}^{k-1}\left(A_{k,i}\sum_{j=1}^{i}\left(F_{i,j}V_je^{-B_jx}C_j\right)\right)=\sum_{j=1}^{k-1}\left(\sum_{i=j}^{k-1}\left(A_{k,i}F_{i,j}\right)V_je^{-B_jx}C_j\right)$$

For $1\le j<k$:

$$F_{k,j}A_{j,j}-A_{k,k}F_{k,j}=\sum_{i=j}^{k-1}A_{k,i}F_{i,j} \tag{3.19}$$

Elements of matrix $F_{k,j}$ can be solved from (3.19) which is a system of $l_k\times l_j$ linear equations. Using the left boundary condition:

$$D_k^{-1} J_k(0) = -\frac{\partial \phi_k}{\partial x} = \sum_{i=1}^{k} F_{k,i} V_i B_i C_i \tag{3.20}$$

$$C_k = (V_k B_k)^{-1} \left(D_k^{-1} J_k(0) - \sum_{i=1}^{k-1} F_{k,i} V_i B_i C_i \right) \tag{3.21}$$

All of coefficients in flux solution (3.18) can be computed sequentially. In this method, the eigenvectors of the diagonal blocks are needed instead of the eigenvectors of matrix A. This is advantageous since the eigenvectors of the diagonal blocks are normally lower order and therefore much easier to compute. If there is no up-scattering, matrix A becomes a lower triangle matrix. All blocks have a size of 1×1, and the algorithm becomes even simpler:

$$\phi_k = \sum_{i=1}^{k} F_{k,i} e^{-B_i x} \tag{3.22}$$

$$B_k = \sqrt{A_{k,k}} \tag{3.23}$$

$$F_{k,k} = (B_k)^{-1} \left(\frac{J_k(0)}{D_k} - \sum_{i=1}^{k-1} B_i F_{k,i} \right) \tag{3.24}$$

$$F_{k,j} = \left(A_{j,j} - A_{k,k} \right)^{-1} \sum_{i=j}^{k-1} A_{k,i} F_{i,j} \tag{3.25}$$

Typically, four types of reflector cross sections are required for three dimensional core calculations: top, bottom, side and corner. The structures of the top and bottom reflectors can be different and they may have different cross sections. However, the cross sections for the axial reflector can usually be used for the top reflector since there is typically a small difference in the respective cross sections which will have little influence on the neutronics behavior in the axial reflector region. The 1-D reflector model shown in Figure 3.1 is generally adequate to generate the cross sections for side reflectors. However, it is cumbersome to solve the 2-D homogeneous diffusion equation for the corner reflector region and it has been shown that the corner reflector cross section can be well represented by correcting the scattering cross sections. Therefore, the corner reflector cross section is easily obtained by modifying the scattering cross section with a correction factor[5];

$$r_{2D} = \frac{P_{FA} - d}{P_{FA}}, \tag{3.26}$$

where P_{FA}, and d denote the fuel assembly pitch and shroud thickness, respectively.

4. The GenPMAXS Code

4.1. Introduction to the GenPMAXS Code

GenPMAXS (Generation of the Purdue XS set) generates the macroscopic cross section file, PMAXS and thus serves as the interface between lattice codes and PARCS. It reads other macroscopic cross section libraries and the results of lattice codes, such as HELIOS, TRITON, and CASMO, and produces the macroscopic cross section file in the PMAXS format. GenPMAXS was written in FORTRAN 90 and Figure 4.1 shows the overall flow of GenPMAXS.

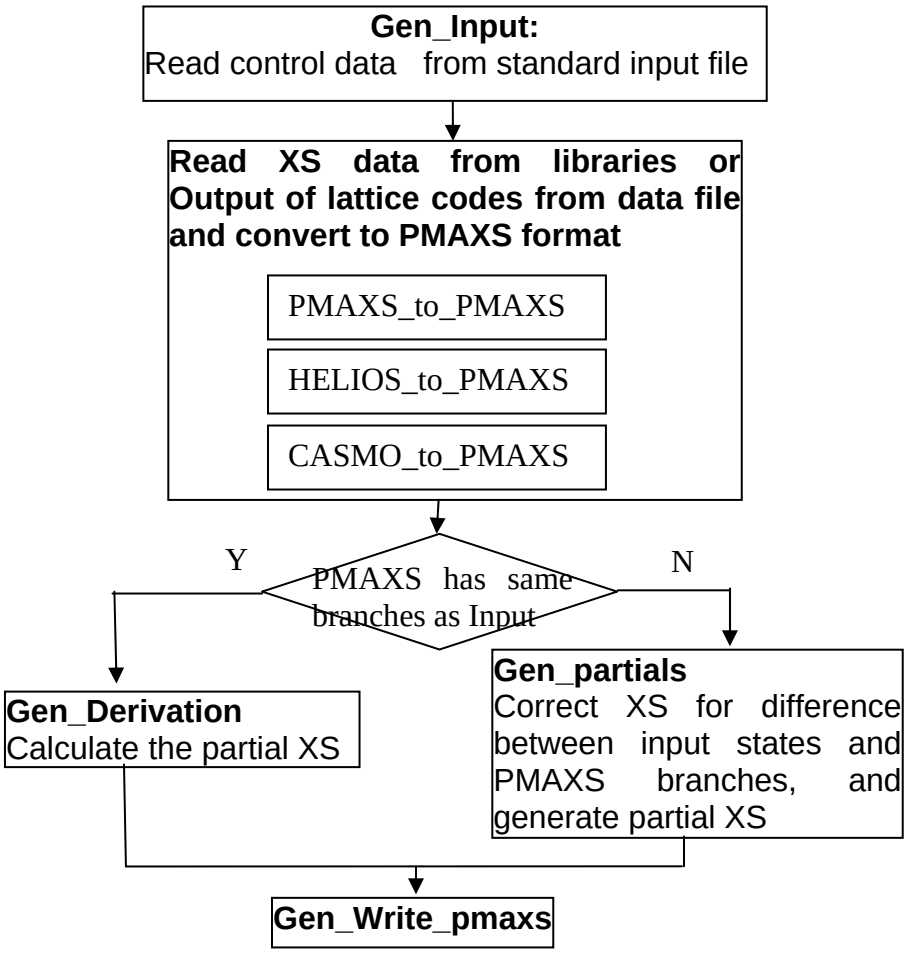


Figure 4.1. Overall Flow in GenPMAXS

4.2. GenPMAXS Code Structure

Because GenPMAXS was written in FORTRAN 90, memory can be allocated or de-allocated in any of its subroutines without restriction using the modularity concept of FORTRAN 90. There are two modules for data structure in the code, xsblock_data and pmaxs_data modules, which define the data structure consistent with the PMAXS format and provides the memory allocation which is performed after reading the proper inputs (see Figure 4.1). Therefore the GenPMAXS code has no pre-defined compilation options and employs standard memory allocation methods.

GenPMAXS first reads the primary control data from the standard input file which contains information about XS data file and job options. A detailed description of standard input file is given in Appendix B. GenPMAXS then reads XS data from libraries or the output of lattice codes from data files and converts them to PMAXS format with different modules according to the source of XS data. These modules will be introduced in later sections of the manual. If PMAXS has the same number of branches as the input and the branches have same state values as the input, then the subroutine Gen will be called to generate partial derivatives of the cross sections. Otherwise a new subroutine, Gen_partials, will be called which was developed for converting CASMO output to PMAXS file. In this subroutine, the cross section data is modified if the input states are different from PMAXS branches and then partial derivatives of the cross section data are generated with the corrected data. After partial derivatives of the XS are generated, GenPMAXS then generates a PMAXS file. The format of PMAXS file is given in Appendix A.

5. Generating PMAXS from PMAXS files

5.1. Introduction

PMAXS-to-PMAXS module provides functionality to maintain PMAXS. There are 4 kinds of maintenance operations: merging PMAXS files, converting between raw cross sections and partial derivations, sorting branches, and converting old format PMAXS to current PMAXS. The most important operation performed on PMAXS files is the merging of two or more PMAXS files into one PMAXS file. Currently, GenPMAXS allows two kinds of merging, merging branches and merging history cases.

First, GenPMAXS can merge branches in different PMAXS files for the same history case. In this case, the reference state of the first PMAXS will be defined as the reference state of the merged PMAXS. All branches from all input PMAXS files should be different from each other except that the reference state of the merged PMAXS can appear in each input PMAXS. Second, GenPMAXS can merge different history cases together as long as the branch structures of each input PMAXS are the same. The burnup sets of input PMAXS files can be different from each other but the first burnup set of all input PMAXS files must be the same.

GenPMAXS can also convert between raw cross sections and partial derivatives. Although the PMAXS files used by PARCS should have partial derivatives for non-reference branches, GenPMAXS also can accept PMAXS files which contain raw cross section data for non-reference branches. Merging PMAXS files with raw cross sections is simpler than merging PMAXS files with partial derivatives. When GenPMAXS merges branches in different PMAXS files for the same history, GenPMAXS will first convert those partial derivatives to raw cross section. If any PMAXS file contains partial derivatives. It is also more convenient for other lattice codes, such as TRITON, to print out files similar to PMAXS with raw cross sections instead of partial derivatives.

GenPMAXS always outputs PMAXS with branches in a sorted order. The branches are sorted according to values of the first independent variable in ascending order, then the second variable, until the last variable. If there is a PMAXS file with branches not in the correct order, GenPMAXS will read it and print out a new PMAXS file with branches in the correct sorted order.

During development from GENPXS to GenPMAXS, the format of PMAXS file was changed. We refer the format of PMAXS file generated by GENPXS as the old PMAXS format and the format of the PMAXS files generated by the new GenPMAXS as current PMAXS format. GenPMAXS can read PMAXS files in old format and current format and print them out in current PMAXS format. Since cross section files from some codes such as TRITON are still in old PMAXS format, the capability of reading old format PMAXS file is valuable.

5.2. PMAXS-to-PMAXS Code Structure

The module PMAXS-to-PMAXS reads XS data from one or more PMAXS files, and performs maintenance operations as specified by the user and prints them in one PMAXS file. The code structure of PMAXS-to-PMAXS is shown in figure 5.1.

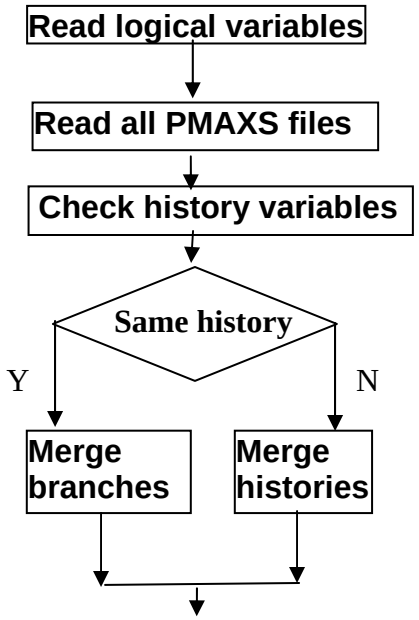


Figure 5.1. Code structure of PMAXS-to-PMAXS

PMAXS-to-PMAXS module consists with three major subroutines, check_history, single _history, multi_history. In subroutine check_history, the values of history variables from all PMAXS files are checked to determine whether all PMAXS files contain data for the same history. If this is the case, then subroutine single_history will be called to merge branches for the same history, otherwise subroutine multi_history will be called to merge histories.

6. Generating PMAXS from HELIOS

6.1. Introduction

The PMAXS file can be constructed using output from different lattice physics codes. However, to demonstrate the functionality of PMAXS and to familiarize the user with the specific principles described above, this section will describe the preparation of a PMAXS file using the HELIOS lattice physics code. HELIOS is a well established neutron and gamma transport code for lattice burnup calculations in two-dimensional geometries. One of the attractive features of the HELIOS code is that it has a geometric flexibility which is enabled by the use of the collision probability method (CPM) and the current coupling collision probability (CCCP) solution methods. Since HELIOS can calculate almost any two-dimensional geometry for various fuel compositions, it can generate the cross sections for most any current nuclear reactor application.

6.2. HELIOS Input Concept

The HELIOS code consists of three sub-codes: AURORA, HELIOS, and ZENITH. AURORA is a input pre-processing code for treatment of the system geometry, assignment of the composition into the region, and defining some parameters, etc. ZENITH is an output post-processing code for generating the output. A general schematic of these codes is shown in Figure 6.1. The HERMES file is a database shared by the three codes. Figure 6.2 shows the typical flow diagram for using AURORA, HELIOS, and the ZENITH code. As indicated in Figure 3.2 there are two types of inputs: one for AURORA another for ZENITH. Because some input data is not changed while some data may be specific to the the type of calculations or geometric modeling, the input is divided into an expert and short input. The expert input is a large input deck and contains the unchangeable properties. The short input contains the changeable properties such as the fuel loading pattern, the fuel enrichment, fuel geometry data, etc. Therefore, if the expert input is constructed for a particular fuel assembly type, then various calculations are possible by just changing the short input. Appendix C describes the sample inputs for the typical 17x17 PWR fuel assembly and 8x8 BWR fuel assembly, respectively. In Figure 6.2, there is an additional code, ORION, which is an input checking tool which draws the lattice shape from the AURORA input.

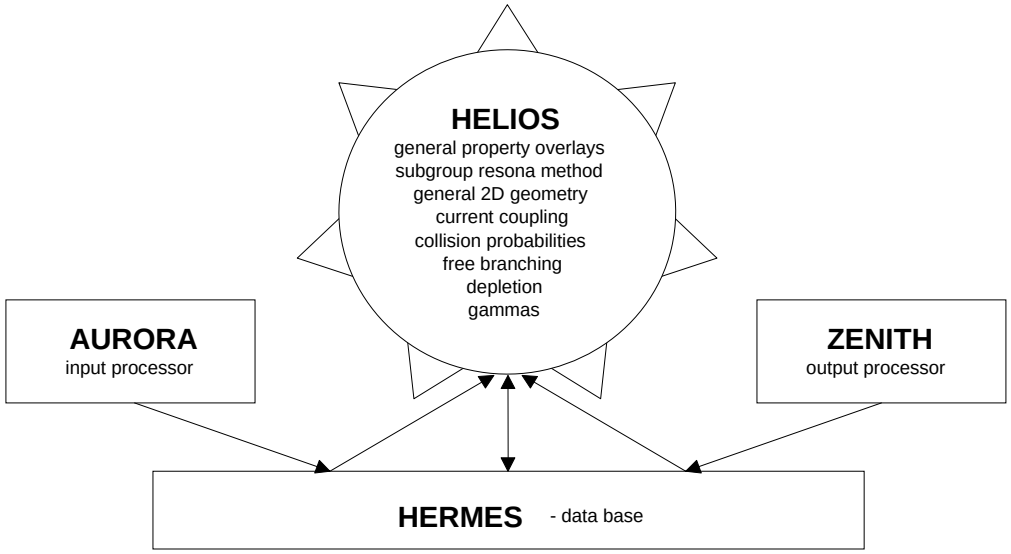


Figure 6.1. Overview of HELIOS Code

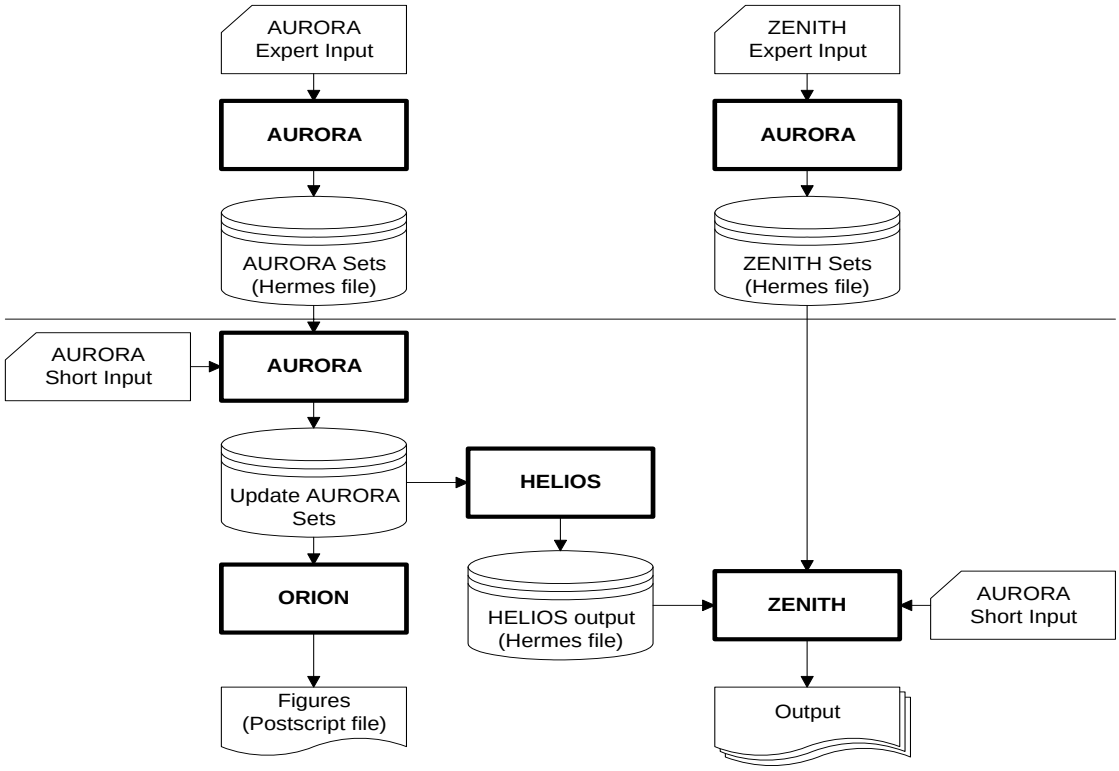


Figure 6.2. Flow Diagram of AURORA, HELIOS and ZENITH codes

6.3. Cross-Section Definitions

The definitions of the principal cross sections used in HELIOS[3] are consistent with industry standards. However, some data are not generated by the default output procedures of HELIOS, such as the effective Xenon/Samarium yield, the effective delayed neutron decay constant, the group-wise form functions, etc. This section provides the definitions of these specific data.

6.3.1. The Effective Xenon/Samarium Yield

Because the absorption cross sections of Xenon and Samarium are represented by their microscopic cross sections and number densities, the fission yields of Xenon and Samarium are important to accurately model the absorption due to Xenon and Samarium. In the typical PWR case, Xenon and Samarium reach an equilibrium state very quickly and the effective fission yield of Xenon and Samarium can be calculated from the equilibrium number densities. In the equilibrium steady state, the number density of the Iodine, Xenon, Promethium and Samarium of a node are given by

$$N_I = \frac{\gamma_I \Sigma_f \phi}{\lambda_I}, \tag{6.1}$$

$$N_{Xe} = \frac{(\gamma_I + \gamma_{Xe})\Sigma_f \phi}{\lambda_{Xe} + \sigma_{Xe,a}\phi}, \tag{6.2}$$

$$N_{Pm} = \frac{\gamma_{Pm} \Sigma_f \phi}{\lambda_{Pm}}, \tag{6.3}$$

$$N_{Sm} = \frac{\lambda_{Pm} N_{Pm}}{\sigma_{Sm,a}\phi} = \frac{\gamma_{Pm}\Sigma_f}{\sigma_{Sm,a}}, \tag{6.4}$$

where

Σ_f = node average macroscopic fission cross section ,
ϕ = node average flux,
γ = effective fission yield,
λ = decay constant (/sec),
N = node average number density (/cm^3).

After the node average number densities, macroscopic fission cross sections, and flux at the equilibrium state are obtained using the lattice code, the effective yield data can be generated as follows:

$$\gamma_I = \frac{\lambda_I N_I}{\Sigma_f \phi}, \tag{6.5}$$

$$\gamma_{Xe} = \frac{N_{Xe}\left(\lambda_{Xe} + \sigma_{Xe,a}\phi\right)}{\Sigma_f \phi} - \gamma_I , \tag{6.6}$$

$$\gamma_{Pm} = \frac{N_{Sm}\sigma_{Sm.a}}{\Sigma_f} . \tag{6.7}$$

To insure consistency with the PARCS code, it is recommended that Table 6.1 values are used as the decay constants of Eqs. (6.5) ~ (6.7).

Table 6.1. Recommended Decay Constants (/sec)

Isotope	Decay Constant
Xe-135	2.09167E-05
I-135	2.89500E-05
Pm-149	3.55568E-06

6.3.2. Delayed neutron data

Because HELIOS code does not calculate the adjoint fluxes, the effective delayed neutron fraction in HELIOS for delayed neutron group d is given by

$$\beta_{eff,d} = \frac{k^x}{k} \frac{\sum_g \beta_{g,d} \upsilon\Sigma_{fg} \phi_g V}{\sum_g \upsilon\Sigma_{fg} \phi_g V} , \tag{6.8}$$

where

$$k = \frac{\sum_g \upsilon\Sigma_{fg} \phi_g}{\sum_g (\Sigma_{ag} + DB^2)\phi_g} , \tag{6.9}$$

$$\beta_{g,d} = \frac{\sum_r \sum_{g' \in g} \sum_i \beta_{i,d} N^{i,r} \left(\upsilon\sigma\right)_{fg}^{i,r} \phi_g^r V_r}{\sum_r \sum_{g' \in g} \sum_i N^{i,r} \left(\upsilon\sigma\right)_{fg}^{i,r} \phi_g^r V_r} . \tag{6.10}$$

In Eqs. (6.8) ~ (6.10), the r, i, and g denote the region, isotope and neutron group, respectively, and k^x is the same as Eq. (6.9), except that the summation over g is for the groups below about 0.45MeV. The decay constants of the delayed neutrons are calculated using the following equations:

$$\lambda_d = \frac{\sum_i \beta_d^i N^i \sigma_f^i}{\sum_i \frac{\beta_d^i N^i \sigma_f^i}{\lambda_d^i}}, \qquad (6.11)$$

where

β_d^i = d group delayed neutron yield fraction from isotope i,
λ_d^i = decay constant of d group delayed neutron from isotope i.

Values of λ_d^i, y^i, and a_d^i can be taken from a textbook or a topical report and typical values are given in Table 6.2 and 6.3.

Table 6.2. Delayed Neutron Yield Data

Isotope	β_1^i	β_2^i	β_3^i	β_4^i	β_5^i	β_6^i
Th232	7.43E-04	2.57E-03	3.06E-03	8.99E-03	3.39E-03	1.65E-03
U233	2.55E-04	6.80E-04	5.28E-04	1.04E-03	3.39E-04	1.21E-04
U235	2.40E-04	1.24E-03	1.18E-03	2.65E-03	1.09E-03	4.56E-04
U238	2.13E-04	1.72E-03	2.00E-03	5.88E-03	3.88E-03	1.57E-03
Pu239	8.15E-05	5.31E-04	4.02E-04	7.34E-04	3.82E-04	1.16E-04
Pu240	9.20E-05	7.28E-04	4.34E-04	9.49E-04	5.16E-04	1.57E-04
Pu241	9.92E-05	1.23E-03	7.84E-04	1.92E-03	1.09E-03	3.75E-04
Pu242	1.20E-04	1.42E-03	7.70E-04	2.00E-03	1.38E-03	4.39E-04

Table 6.3. Delayed Neutron Decay Constant (1/sec)

Isotope	λ_1^i	λ_2^i	λ_3^i	λ_4^i	λ_5^i	λ_6^i
Th232	1.31E-02	3.50E-02	1.27E-01	3.29E-01	9.10E-01	2.8203E+0
U233	1.29E-02	3.47E-02	1.19E-01	2.86E-01	7.88E-01	2.4417E+0
U235	1.33E-02	3.27E-02	1.21E-01	3.03E-01	8.49E-01	2.8530E+0
U238	1.36E-02	3.13E-02	1.23E-01	3.24E-01	9.06E-01	3.0487E+0
Pu239	1.33E-02	3.09E-02	1.13E-01	2.93E-01	8.57E-01	2.7297E+0
Pu240	1.33E-02	3.05E-02	1.15E-01	2.97E-01	8.48E-01	2.8796E+0
Pu241	1.36E-02	3.00E-02	1.17E-01	3.07E-01	8.70E-01	3.0028E+0
Pu242	1.36E-02	3.02E-02	1.15E-01	3.04E-01	8.27E-01	3.1372E+0

6.3.3. Form Function for Pin Power Reconstruction

The purpose of the form functions is to generate the local pin power distribution and to treat the local heterogeneous flux distribution in the fuel assembly. Two types of form functions are required to provide the heterogeneous power and the group-wise flux distributions in the fuel assembly. The power form function is defined by the normalized power distribution in the fuel assembly, and the group-wise flux form function is defined group-wise as:

$$f_g(x,y) = \frac{\kappa\Sigma_{fg}(x,y)\phi_g(x,y)}{\kappa\bar{\Sigma}_{fg}\bar{\phi}_g}, \qquad (6.13)$$

where

$\kappa\bar{\Sigma}_{fg}$ = g-th group, assembly averaged fission yield energy,

$\bar{\phi}_g$ = g-th group, assembly averaged fluxes ,

$\kappa\Sigma_{fg}(x,y)$ = g-th group, cell averaged fission yield energy at the position (x,y),

$\phi_g(x,y)$ = g-th group, cell averaged fluxe at position (x,y),

(x,y) = the center of fuel cell.

6.4. The ZENITH Output Keywords for the GenPMAXS Code

In order to insure the consistency of HELIOS output with PMAXS, GENPMAXS requires several keywords from ZENITH in order to interpret the HELIOS output. These are shown in Table 6.4. The GENPMAXS code provides two output files: the PMAXS formatted file and an execution summary for GENPMAXS (See the GENPMAXS input description in Appendix B). Guidelines for the overall execution of the HELIOS/GENPMAXS/DEPLETOR/PARCS code system are also provided in Appendix B.

Table 6.4. The ZENITH Output Keywords for the GENPMAXS Code.

Keyword	Purpose
%FILE_CONT 1	File control flag. It contains number of neutron groups, number of fuel pins, etc.
%FILE_CONT 2	File control flag. It contains the minimum energy bound of each neutron group.
%FILE_CONT 3	File control flag. It contains flow areas for in-channel, by pass and water holes.
%FILE_CONT 4	File control flag. It contains assembly pitch and position for rod array.

%STAT_**** #	Branch state flag. **** will have BRBS, BRCR, BRDC, BRPC, BRTF, BRTM… etc. And # denotes the sequential number of the same branch state.
%XS_PRIN %****	Principle cross section flag. **** will have KINF, VEL, CHI, STR, SAB, SFI, SKF, SNF, SNU, SDF. KINF = infinit multiplication factor. VEL = group wise neutron velocity. CHI = fission spectrum. STR = transport cross section. SAB = absorption cross section, it includes fission. SFI = fission cross section. SKF = kappa-fission cross section. SNF = nu-fission cross section. SNU = prompt neutron yield per fission. SDF = discontinuity factor.
%XS_SCT %SCT	Scattering cross section flag. Up-scattering is ignored.
%XS_XESM %****	Xe/Sm cross section flag. *** will have YLDXE, YLDID, YLDPM, XENG, SMNG, XEND, SMND. YLDXE = effective yield of Xe-135. YLDID = effective yield of I-135. YLDPM = effective yield of Pm-149. XENG = microscopic absorption Xs of Xe. SMNG = microscopic absorption XS of Sm. XEND = assembly averaged Xe-135 number density. SMND = assembly averaged Sm-149 number density.
%XS_SB %****	Soluble boron cross section flag. *** will have SBNG, SBND. SBNG= microscopic absorption XS of natural boron. SBND= number density of natural boron in coolant.
%XS_BETA %****	Effective delayed neutron flag. **** will have DCAYB, BETA. DCAYB = decay constant of delayed neutron. BETA = effective beta.
%XS_PFF %****	Power form function flag. *** will have PAXIS 1, PAXIS 2, PFF. PAXIS 1 = x-axis coordinate of fuel pin. PAXIS 2 = y-axis coordinate of fuel pin. PFF = power form function.
%XS_GFF %***	Group-wise form function flag. *** will have FAXIS 1, FAXIS 2, GFF. FAXIS 1 = x-axis coordinate of pin cell. FAXIS 2 = y-axis coordinate of pin cell. gfF = power form function.

6.5. HELIOS-to-PMAXS Code Structure

The module HELIOS-to-PMAXS reads XS data from output of HELIOS code and converts to PMAXS format. The code structure of HELIOS-to-PMAXS is shown in Figure 6.3.

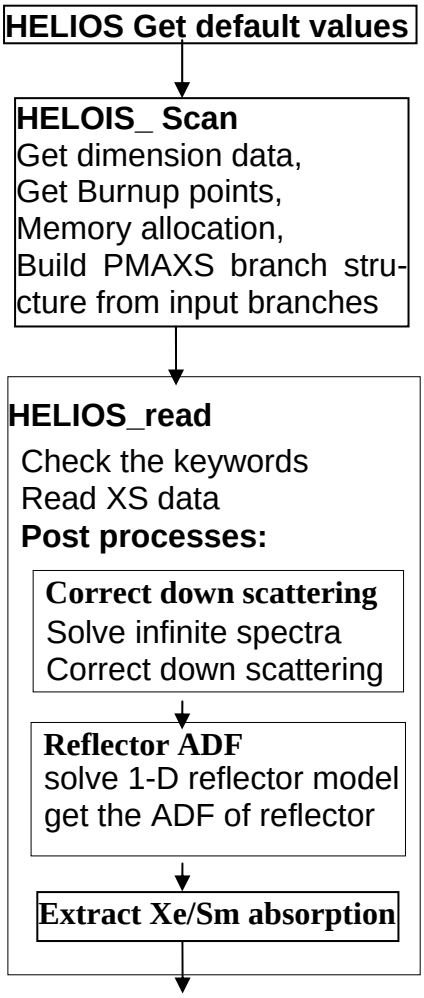


Figure 6.3. Code structure of HELIOS-to-PMAXS

The HELIOS-to-PMAXS module consists of three major subroutines, HELIOS-Get-default-values, HELIOS-scan, and HELIOS_read. In the subroutine HELIOS-Get-default-values, the dimensions and some control flags are set to default values according to assembly type, fuel assembly or reflector. In subroutine HELIOS_scan, the information about the dimensions of the XS data are scanned and memory are allocated according to these dimensions. The burnup points for all branches in all histories will be also scanned if not provided in the GenPMAXS input file. In the subroutine

HELIOS_read, the XS data are read by first checking the keywords described in sections 6.4. After reading the XS data, three post processing steps are performed: (1) correct the down scattering data, (2) generate ADF for reflector, and (3) extract Xe/Sm absorption. The XS data are stored in the PMAXS data structure and prepared for generating partial derivatives and printing into the PMAXS file.

6.6. Benchmark I

In order to verify the procedures used in GenPMAXS, PMAXS and the depletion module in PARCS, A simple PARCS model shown in Figure 6.4 is used to reproduce the k-infinite of assemblies with PMAXS generated from lattice codes with an infinite spectrum.

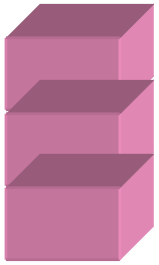


Figure 6.4. PARCS Model for Benchmark Problems

In this Benchmark, a BWR fuel assembly with a 10x10 fuel bundle shown as Figure 6.5 is calculated with HELIOS code. This fuel assembly is depleted at reference state and restarted for 10 branch calculations. The reference and branches states are described in Table 6.5 In order to provide reference for the PARCS calculation, all HELIOS calculation are performed with an infinite spectrum.

Table 6.5. Reference and Branches States

Branches	Ind	Control Rod State	Coolant Density (g/cc)	Boron Concentration (ppm)	Fuel Tmepearture (K)	Coolant Tmepearture (K)
Reference		0	0.456652	0	933	561.22
Control rod	1	1	0.456652	0	933	561.22
Coolant Density	1	0	0.177504	0	933	561.22
	2	0	0.317078	0	933	561.22
	3	0	0.596226	0	933	561.22
	4	0	0.7358	0	933	561.22
Soluble Boron	1	0	0.177504	1000	933	561.22
	2	0	0.456652	1000	933	561.22
	3	0	0.7358	1000	933	561.22
Fuel Temperature	1	0	0.456652	0	561.22	561.22
	2	0	0.456652	0	2000	561.22

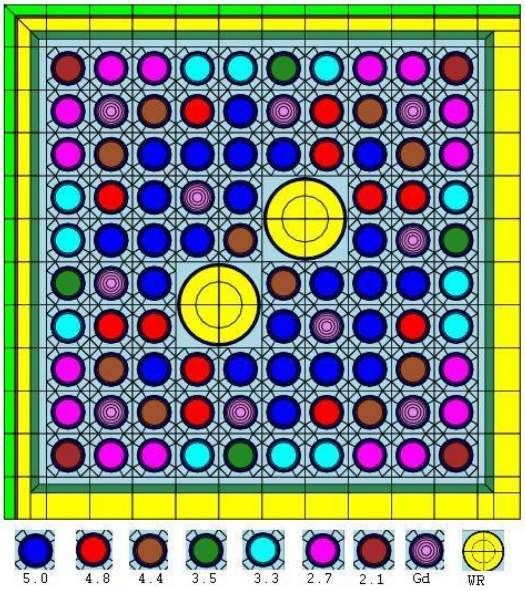


Figure 6.5. BWR Assembly in Benchmark I

The HELIOS output is processed with GenPMAXS to generate the PMAXS file. The input file for GenPMAXS is given in section 2.1 of appendix B. Two PARCS cases were run for this benchmark: a depletion case for the reference state and a restart case for a low coolant density branch. The first depletion case verified the procedure for selecting HELIOS output, reading HELIOS output and generating PMAXS by GenPMAXS, and reading and using PMAXS by PARCS. It also has verified the following functions in PARCS:

1) Depletion capability in PARCS
2) Equilibrium Xe/Sm calculation

The second restart case verified the following functions in PARCS:

3) Restart from provided depletion states
4) Interpolation for burnups
5) Retrieve cross section from reference cross section and partial derivatives.

The k-inf from HELIOS and PARCS calculated for the BWR assembly are shown in Figure 6.6 and the differences are shown in Figure 6.7 which shows that the maximum k-inf difference between PARCS and HELIOS is less then 4 pcm for the reference state and less than 10 pcm for the low coolant density branch. Figure 6.6 shows the k-inf from PARCS are right on the curve of the k-inf from HELIOS, even for the points at which k-infs from HELIOS are not provided. The input and output files of HELIOS and GenPMAXS for this benchmark are stored in the GenPMAXS repository and the input and output files of PARCS are stored in PARCS repository.

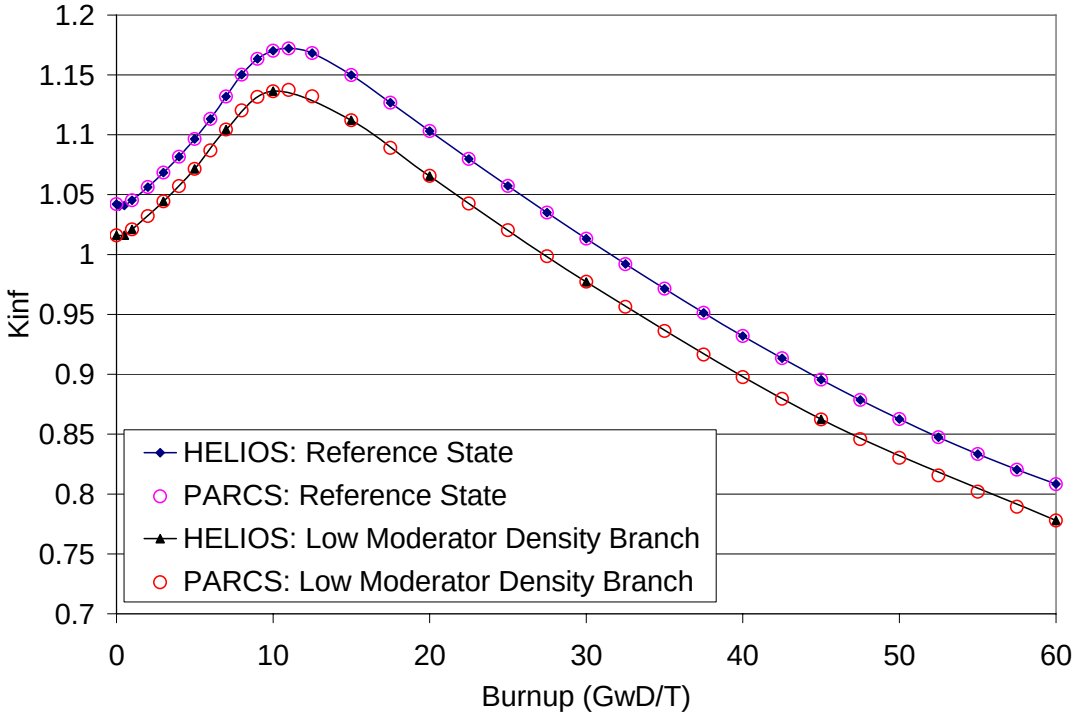


Figure 6.6. PARCS and HELIOS K-inf for BWR assembly in Benchmark1

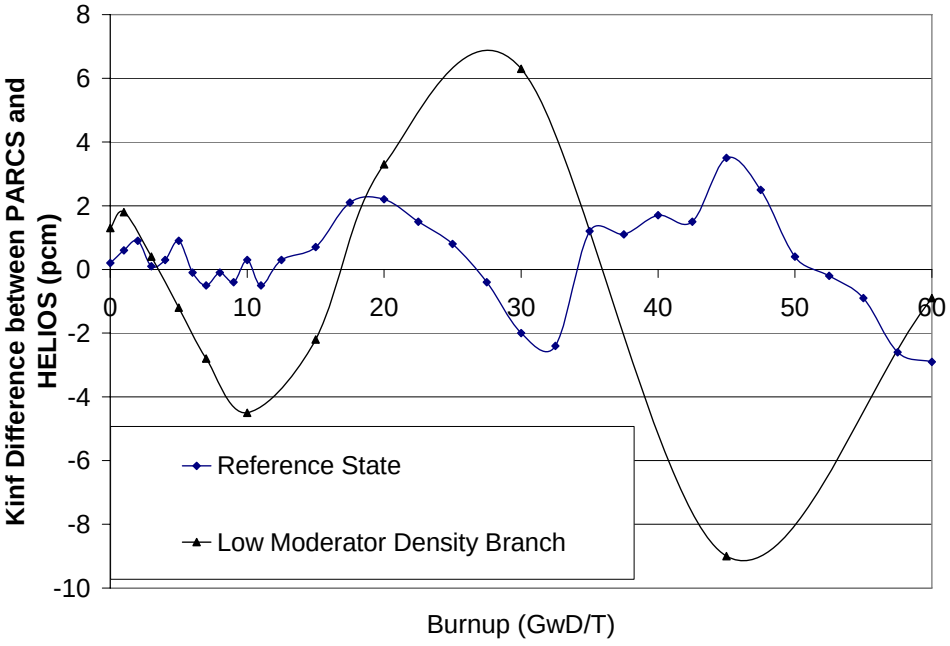


Figure 6.7. PARCS and HELIOS K-inf difference for BWR assembly in Benchmark1

7. Generating PMAXS from CASMO

7.1. Introduction

CASMO is a multigroup two dimensional integral transport theory code for burnup calculations on two-dimensional BWR and PWR fuel assemblies or simple pin cells. It was developed by Studsvik and widely used by nuclear utilities. This section will describe the preparation of a PMAXS formatted cross section file using the CASMO lattice physics code.

7.2. CASMO output file

CASMO generates several output files to include a Card Image File containing data to be used by other programs. The Card Image (CI) file, which is referred to as a CAX file with the extension name of "cax", has is used to generate the PMAXS file. The data in CAX file are divided into blocks with three character names. Several of the data blocks are written on the CI file, while other blocks are written only if activated by input cards in the CASMO run. The blocks which contain data for PMAXS are listed in Table 7.1. If a user chooses to include direct energy deposition information in PMAXS, the GAM card must be added in the CASMO input file.

Table 7.1. Blocks in CAX which are used for generating PMAXS

block	Activated by input cards	Content
TIT		Burnup and Current/history of void, Tf, TC, PC,CR
HYD		Hydraulic data and geometryedit
STP		Coolant and bypass densities
RHC		Rehomogenization cross section
ADF		ADF and CDF
POW XPO GPO	 XPO,PRI GAM	Power distribution w/o gamma smearing
SAM		Sm Xe, yeild, cross section
GED	GAM	Direct energy deposition, volume fractions
POL		cross sections, peaking factor
DET		Detector information
DEL		Delay constants, beta, lambda, 1/v

These blocks provide all the cross section data as well as the state values for the history and branch cases. GenPMAXS can read these state values if they are not provided in the GenPMAXS input file. The STP block is written in the CI file only by the most recent version of CASMO. If the coolant or bypass density is selected as a state variable

for PMAXS and the STP block is not in the CI file, the user must provide all branch, history, and file content information in the GenPMAXS input.

7.3. Construct PMAXS Branches from CASMO Output

The CAX files generated by CASMO may contain histories and branches which do not satisfy PMAXS requirements and hence can not be directly convert to PMAXS. Some of these PMAXS requirements which may not be satisfied include history and branch case information such as:

1) Every branch except the reference branch should have a base branch. The base branch has reference values for the variables of the current branch and the same state values as the current branch for all other state variables. This requires all branches must be derived from the reference branch by changing one state variable for each branch from its base branch.
2) All histories have the same branches which include the number of branches and the state value of each branch.
3) The same branch in different histories should have the same burnup points.

Most existing CAX files can not satisfy these requirements. There are four general problems in most existing CAX files in which branches are automatically generated by the typical “S3C” expansion.

1) For the cold state, many state variables are changed at the same time from the other branches which violates the second requirement of PMAXS.
2) The reference history often has more branches then other histories.
3) Since the burnup step sizes are normally automatically determined by the CASMO code according to the concentration of burnable poison in the calculated model, the same branch with different histories may have different burnup points.
4) Data for the shut down cooling period are generated by the “S3C” expansion which can not be included in the current PMAXS structure.

These four issues are treated by GenPMAXS as follows:

1) The set of branches which satisfy the PMAXS requirements and may be different from branches in the CAX file are stored in PMAXS. The branches in the CAX file will hereafter be referred to as input branches. The cross section data will be modified for the state difference between PMAXS branches and input branches to insure that PARCS can reproduce cross sections as input branches. Some branches may be added to PMAXS so that it is possible PMAXS can contain more branches than specified as input branches.
2) Data will be created for branches which does not exist for non-reference histories.
3) GenPMAXS and PARCS codes are modified to allow same branches in different histories to have different burnup points.

4) The data for the shut down cooling period are excluded from PMAXS in the current version of GenPMAXS.

An example of the history and branches for a typical PWR fuel assembly from an “S3C” expansion are shown in Table 7.2.

Table 7.2. Histories and Branches for a PWR fuel Assembly from an “S3C” Expansion

Index	Descriptions*	CR	DC	PC	TF	TC
1	H1	0	.713	600	820.5	579.5
2	PC	0	.713	0.1	820.5	579.5
3	PC	0	.713	1200	820.5	579.5
4	PC	0	.713	2400	820.5	579.5
5	DC/TC	0	.753	600	820.5	559.5
6	DC/TC	0	.662	600	820.5	599.5
7	TF	0	.713	600	559.5	579.5
8	CR	1	.713	600	820.5	579.5
9	DC/TF/TC	0	1.01	0.1	293	293
10	DC/TF/TC	0	1.01	600	293	293
11	DC/TF/TC	0	1.01	1200	293	293
12	DC/TF/TC	0	.923	0.1	425	425
13	DC/TF/TC	0	.923	600	425	425
14	DC/TF/TC	0	.923	1200	425	425
15	DC/TF/TC	0	.753	0.1	559.5	559.5
16	DC/TC	0	.753	600	559.5	559.5
17	DC/TF/TC	0	.753	1200	559.5	559.5
18	DC/TF/TC	0	.753	2400	559.5	559.5
19	DC/PC/TF/TC	1	1.01	1200	293	293
20	DC/PC/TF/TC	1	.923	1200	425	425
21	DC/PC/TF/TC	1	.753	1200	559.5	559.5
1	H2	0	.753	600	820.5	559.5
2	DC/PC/TF/TC	0	1.01	1200	293	293
1	H3	0	.713	1200	820.5	579.5
2	DC/TF/TC	0	1.01	1200	293	293
1	H4	0	0	600	559.5	579.5
1	H5	1	0	600	820.5	579.5
2	CR	0	0	600	820.5	579.5

*In this column, H1~H5 indicate 5 histories, other letters indicate the variables which have values different from their most possible base branches.

In this example, the coolant density is always changed together with the coolant temperature. The coolant density or the coolant temperature can be selected as a state variable in PMAXS. There are still 2 or 3 variables changed in the same branch so the

branch structure can be different from the input branch structure used in PMAXS. If the coolant temperature is selected as the state variable in PMAXS, then the 5 history cases are used as shown in Table 7.3

Table 7.3. History Cases of PWR S3C Example with TC as State Variable

PMAXS Index	CAX Index	Type*	CR	PC	TF	TC
1	H1	RE	0	600	820.5	579.5
2	H5	CR	1	600	820.5	579.5
3	H3	PC	0	1200	820.5	579.5
4	H4	TF	0	600	559.5	579.5
5	H2	TC	0	600	820.5	559.5

*Type of history is defined same as type of branches which is the first state variable which has value different from reference history.

The corresponding PMAXS branch structure is shown in Table 7.4. The input branches in the reference history can be mapped one-to-one onto PMAXS branches as shown in Table 7.5.

Table 7.4. PMAXS Branch Structure of PWR S3C Example with TC as State Variable

PMAXS Index	Type	CR	PC	TF	TC
1	RE	0	600	820.5	579.5
2	CR	1	600	820.5	579.5
3	PC	0	0.1	820.5	579.5
4	PC	0	1200	820.5	579.5
5	PC	0	2400	820.5	579.5
6	PC	1	600	820.5	579.5
7	TF	0	0.1	293	579.5
8	TF	0	0.1	425	579.5
9	TF	0	0.1	559.5	579.5
10	TF	0	600	293	579.5
11	TF	0	600	425	579.5
12	TF	0	600	559.5	579.5
13	TF	0	1200	293	579.5
14	TF	0	1200	425	579.5
15	TF	0	1200	559.5	579.5
16	TF	0	2400	559.5	579.5
17	TF	1	1200	293	579.5
18	TF	1	1200	425	579.5
19	TC	0	600	559.5	559.5

20	TC	0	600	820.5	559.5
21	TC	0	600	820.5	599.5

Table 7.5. Branch Mapping for Reference History of PWR S3C Example with TC as State Variable

PMAXS Index	CAX Index	Type	CR	PC	TF	TC
1	1	RE	0	600	820.5	579.5
2	8	CR	1	600	820.5	579.5
3	2	PC	0	0.1	820.5	579.5
4	3	PC	0	1200	820.5	579.5
5	4	PC	0	2400	820.5	579.5
6	21	PC	1	1200	559.5/820.5*	559.5/579.5
7	9	TF	0	0.1	293	293/579.5
8	12	TF	0	0.1	425	425/579.5
9	15	TF	0	0.1	559.5	559.5/579.5
10	10	TF	0	600	293	293/579.5
11	13	TF	0	600	425	425/579.5
12	7	TF	0	600	559.5	579.5
13	11	TF	0	1200	293	293/579.5
14	14	TF	0	1200	425	425/579.5
15	17	TF	0	1200	559.5	559.5/579.5
16	18	TF	0	2400	559.5	559.5/579.5
17	19	TF	1	1200	293	293/579.5
18	20	TF	1	1200	425	425/579.5
19	16	TC	0	600	559.5	559.5
20	5	TC	0	600	820.5	559.5
21	6	TC	0	600	820.5	599.5

* Inupt/PMAXS state values

There are 12 branches in PMAXS which have state values different from the input branches. The non-reference histories have fewer input branches than the PMAXS branches as shown in Table 7.6 which shows the branch mapping for non-reference histories. There are only 1 or 2 branches out of the 21 branches for the non-reference histories shown in Table 7.5.

Table 7.6. Branch Mapping for Non-reference Histories of PWR S3C Example with TC as State Variable

PMAXS Index	CAX Index	Type	CR	PC	TF	TC
CR history						
1	2	RE	0	600	820.5	579.5
2	1	CR	1	600	820.5	579.5
PC history						
4	1	PC	0	1200	820.5	579.5
13	2	TF	0	1200	293	293/579.5
TF history						
12	1	TF	0	600	559.5	579.5
TC history						
13	2	PC	0	1200	293	293/579.5
20	1	TC	0	600	820.5	.753

After reading the history and branch information from the GenPMAXS input file, or scanning this information from the CAX file, GenPMAXS builds the PMAXS branch structure, maps the input branches to the PMAXS branches, reads cross section data of the input branches from the CAX file, and generates cross sections for reference branches of all histories and partial derivatives for all other branches of PMAXS. This is performed using the following steps.

Step 1: A branch is selected as the reference branch from all branches in all histories which is closest to the desired reference state. The reference state can be specified in the input file in which case the user specified reference states will be used as the reference state. Detailed information is provided in Appendix B for the input format. The procedure for distinguishing branches cases is as follows. Assume branch A and branch B have the same values for state variables before the “kth” state variable, and different values for the “kth” state variable. Branch A is said to be closer to the reference branch than branch B if branch A has smaller difference in the “kth” state variable from the reference values than that of branch B.

Step 2: Select a history from all histories which has state values closest to the reference state as the reference history.

Note: If the reference branch or reference history is specified in the GenpMAXS input file, then steps 1 and step 2 will be skipped.

Step 3: Construct a PMAXS branch structure with the information from the branches in the reference history using the following algorithm.

> **Step 3.1**: Initially ITP=0 for all INPUT branches. Then ITP=1 for the branches exist in reference history, then set ITP=-1 for reference branch j. A positive ITP is

a group index which contains 1 or more branches to be determined. A PMAXS branch is created for group 1, and set PTIr(1)=j.

Step 3.2: Generate new groups from the existing groups which have the next state variable different from its PMAXS branch value: SV$\in$ {CR,DC,PC,TF,TC,DM,PM,TM}. Create a SV type PMAXS branch for each new group g which has the same state values for variable SV and variables before SV with all branches in the group g and reference state values for variables after SV. If there is a branch k in group g which has state values the same as PMAXS branch g, or there are branches in group which have the next state values different from all other branches in group, then select branch k which is closest to reference branch and set PTIr(g)=k, and ITP(k)=-g. Otherwise PTIr(g)=0. Repeat step 3.2 until ITP<=0 for all branches

Step 3.3: Rearrange PMAXS branches into PMAXS order

Step 4: Map input branches in the non-reference history to PMAXS branches.

Step 4.1: Initialize ITP=0 for all INPUT branches and TPIk=0 for all PMAXS branches, and then set ITP= 1 for the branches existing in history k. If ITP(TPIr(i))==1, then set TPIk(i)=TPIr(i), and ITP(TPIr(i))=-I. This step matches all branches in the reference history to the same PMAXS branch as in the reference history.

Step 4.2: Process PMAXS branches from the last branch to the first branch and find branches with the same base branch If there are unmatched branches or if the base branch is unmatched, then try to match the unmatched input branches to these branches using the following guidelines:

1. Input branches must have same state values for variable before the type variable of the PMAXS branches.
2. Input branches for each PMAXS branch must have state values for the type variable between neighbors of the PMAXS branch.
3. Match as many PMAXS branches as possible
4. The total difference between PMAXS branches and their matched input branches should be minimized.

Repeat step 4.1 and 4.2 for each non-reference history.

Step 5: Generate XS partials for all non-reference branches in the reference history moving backwards from the last type of branch to the first type of branch. If there is no branch in the current type of branch, go to the previous type of each branch until all branches are processed. If there are branches in the current type of branch, then the following algorithm is used:

Step 5.1: Find their base branch.

Step 5.2: If there is no input branch for each base branch, or if the input branch is different from the PMAXS branch, then create (correct) the base branch by piece wise quadratic interpolation.

Step 5.3: Generate the XS partials.

Step 5.4: Correct XS in all branches in each type of branch for the current type variable that is different between the PMAXS branch and its input branch.

Step 5.5: Go to the previous type of branch until all branches are processed

Step 6, Generate XS partials for all non-reference branches in the non-reference history and if necessary, create or correct XS data for the reference branch in non-reference histories. The branches are processed backwards, from the last type of branch to the first type of branch. If there is no input branch in this history for the current types of branches, then all of these types of branches will have the same partials with its base history. If there are input branch in this history for the current type branch, then the following algorithm will be used:

Step 6.1: Find (create or correct) their base branch

Step 6.2: Generate XS partials for these branches which have input branches mapped to them.

Step 6.3: Generate partials for all branches in the current type by piece wise linear interpolation of the difference between partials from this history and from its base history.

Step 6.4: Correct XS in all branches in the early type of branches for the current type variable difference between PMAXS branch and its input branch.

Step 6.5: Go to the previous type of the branch until all branches are processed.

Using this process, all input branches of the reference history have been mapped to PMAXS branches. There may be more PMAXS branches than input branches of the reference history. Not all input branches in non-reference histories are mapped in PMAXS, and some PMAXS branch may have no corresponding input branch. All cross sections data in the branches which are mapped to PMAXS branches can be reproduced from PMAXS by PARCS.

For the previous example of a PWR fuel assembly, the coolant density is selected as the state variable in PMAXS and the PMAXS histories, PMAXS branch, and the mapping of input branches to PMAXS branches for the reference history and non-reference histories are shown in Tables 7.7-7.10

Table 7.7. Histories of PWR S3C Example with DC as State Variable

PMAXS Index	CAX Index	Type	CR	DC	PC	TF
1	H1	RE	0	.713	600	820.5
2	H5	CR	1	.713	600	820.5
3	H2	DC	0	.753	600	820.5
4	H3	PC	0	.713	1200	820.5
5	H4	TF	0	.713	600	559.5

Table 7.8. PMAXS Branch Structure of PWR S3C Example with DC as State Variable

PMAXS Index	Type	CR	DC	PC	TF
1	RE	0	.713	600	820.5
2	CR	1	.713	600	820.5
3	DC	0	1.01	600	820.5
4	DC	0	.923	600	820.5
5	DC	0	.753	600	820.5
6	DC	0	.662	600	820.5
7	DC	1	1.01	600	820.5
8	DC	1	.923	600	820.5
9	DC	1	.753	600	820.5
10	PC	0	.713	0.1	820.5
11	PC	0	.713	1200	820.5
12	PC	0	.713	2400	820.5
13	PC	0	1.01	0.1	820.5
14	PC	0	1.01	1200	820.5
15	PC	0	.923	0.1	820.5
16	PC	0	.923	1200	820.5
17	PC	0	.753	0.1	820.5
18	PC	0	.753	1200	820.5
19	PC	0	.753	2400	820.5
20	TF	0	.713	600	559.5
21	TF	0	.753	600	559.5

Table 7.9. Branch Mapping for Reference History of PWR S3C Example with DC as State Variable

PMAXS Index	CAX Index	Type	CR	DC	PC	TF
1	1	RE	0	.713	600	820.5
2	8	CR	1	.713	600	820.5

3	10	DC	0	1.01	600	293 / 820.5
4	13	DC	0	.923	600	425 / 820.5
5	5	DC	0	.753	600	820.5
6	6	DC	0	.662	600	820.5
7	19	DC	1	1.01	1200/600	293 / 820.5
8	20	DC	1	.923	1200/600	425 / 820.5
9	21	DC	1	.753	1200/600	559.5 / 820.5
10	2	PC	0	.713	0.1	820.5
11	3	PC	0	.713	1200	820.5
12	4	PC	0	.713	2400	820.5
13	9	PC	0	1.01	0.1	293/ 820.5
14	11	PC	0	1.01	1200	293 / 820.5
15	12	PC	0	.923	0.1	425 / 820.5
16	14	PC	0	.923	1200	425 / 820.5
17	15	PC	0	.753	0.1	559.5/ 820.5
18	17	PC	0	.753	1200	559.5 / 820.5
19	18	PC	0	.753	2400	559.5 / 820.5
20	7	TF	0	.713	600	559.5
21	16	TF	0	.753	600	559.5

Table 7.10. Branch Mapping for Non-reference Histories of PWR S3C Example with DC as State Variable

PMAXS Index	CAX Index	Type	CR	DC	PC	TF
CR history						
1	2	RE	0	.713	600	820.5
2	1	CR	1	.713	600	820.5
DC history						
5	1	DC	0	.753	600	820.5
14	2	PC	0	1.01	1200	293/820.5
PC history						
11	1	PC	0	.713	1200	820.5
14	2	DC	0	1.01	1200	293/820.5
TF history						
20	1	TF	0	.713	600	559.5

With different state variables, the generated PMAXS files will appear different in their branch structures, but the cross sections for all input branches will be reproduced by PARCS from both PMAXS files. The coolant temperature, TC, is often been chosen as

the state variable for PWR fuel assemblies. In the next example, the coolant density is chosen as the state variable for a BWR fuel assembly. An example of the history and branches for a BWR fuel assembly with the "S3C" expansion is given in Table 7.11

Table 7.11. Histories and branches for BWR fuel assembly with "S3C" expansion

Index	Descriptions	CR	DC	PC	TF	TC
1	H1	0	.738	0	813	560
2	DC	0	.458	0	813	560
3	DC	0	.177	0	813	560
4	CR	1	.738	0	813	560
5	TF	0	.738	0	560	560
6	DC/TF/TC	0	.998	0	293	293
7	CR/DC/TF	1	.998	0	293	293
1	H2	0	.458	0	813	560
2	DC	0	.738	0	813	560
3	DC	0	.177	0	813	560
4	CR	1	.458	0	813	560
5	TF	0	.458	0	560	560
6	DC/TF/TC	0	.998	0	293	293
7	CR/DC/TF/TC	1	.972	0	353	353
1	H3	0	.177	0	813	560
2	DC	0	.738	0	813	560
3	DC	0	.458	0	813	560
4	CR	1	.177	0	813	560
5	TF	0	.177	0	560	560
6	DC/TF/TC	0	.998	0	293	293
7	CR/DC/TF	1	.998	0	293	293
8	DC/TF/TC	0	.972	0	353	353
9	DC/TF/TC	0	.862	0	475	475
10	CR/DC/TF/TC	1	.972	0	353	353
11	CR/DC/TF/TC	1	.862	0	475	475
12	DC/PC/TF/TC	0	.998	0.1	293	293
13	DC/PC/TF/TC	0	.998	660	293	293
1	H4	1	.458	0	560	560
1	H5	1	.458	0	813	560
2	CR	0	.458	0	813	560

For this example of a BWR fuel assembly, there are different coolant densityies corresponding to the same coolant temperature due to different void fractions. However, there is only one temperature for each density. So the coolant density instead of the temperature is selected as the state variable in PMAXS. The PMAXS histories, the PMAXS branches, and the mapping of input branches to PMAXS branches for all histories are given in Table 7.12 ~7.14. CR and TF histories have fewer branches than the reference history. There are several input branches in the lower density history which are not mapped to PMAXS branches.

Table 7.12. Histories of BWR fuel assembly

PMAXS Index	CAX Index	Type	CR	DC	PC	TF
1	H2	RE	0	.458	0	813
2	H5	CR	1	.458	0	813
3	H1	DC	0	.738	0	813
4	H2	DC	0	.177	0	813
5	H4	TF	0	.458	0	560

Table 7.13. PMAXS branch structure of BWR fuel assembly

PMAXS Index	Type	CR	DC	PC	TF
1	RE	0	.458	0	813
2	CR	1	.458	0	813
3	DC	0	.738	0	813
4	DC	0	.177	0	813
5	DC	0	.998	0	813
6	DC	1	.998	0	813
7	TF	0	.458	0	560

Table 7.14. Branch mapping for all histories of BWR fuel assembly

PMAXS Index	CAX Index	Type	CR	DC	PC	TF
Reference History						
1	1	RE	0	.458	0	813
2	4	CR	1	.458	0	813
3	2	DC	0	.738	0	813
4	3	DC	0	.177	0	813
5	6	DC	0	.998	0	293/813
6	7	DC	1	.998	0	293/813
7	5	TF	0	.458	0	560
CR History						
1	2	RE5	0	.458	0	813
2	1	CR	1	.458	0	813
Lower DC History						
1	3	RE3	0	.458	0	813
2	4	CR	1	.177/.458	0	813
3	2	DC	0	.738	0	813
4	1	DC	0	.177	0	813

5	6	DC	0	.998	0	293/813
6	7	DC	1	.998	0	293/813
7	5	TF	0	.177/.458	0	560
Higher DC History						
1	2	RE1	0	.458	0	813
2	4	CR	1	.738/.458	0	813
3	1	DC	0	.738	0	813
4	3	DC	0	.177	0	813
5	6	DC	0	.998	0	293/813
6	7	DC	1	.998	0	293/813
7	5	TF	0	.738/.458	0	560
TF History						
7	1	TF4	0	.458	0	560

7.4. CASMO-to-PMAXS Code Structure

The module CASMO-to-PMAXS reads Cross Section (XS) data from the output of the CASMO code and converts it to PMAXS format. The code structure of HELIOS-to-PMAXS is shown in Figure 7.1.

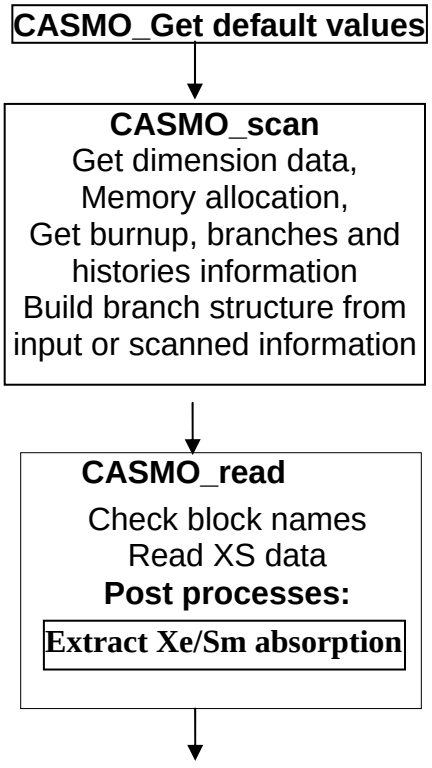


Figure 7.1. Code structure of CASMO-to-PMAXS

The CASMO-to-PMAXS module consists of three major subroutines similar to the HELIOS-to_PMAXS module. These modules are: CASMO-Get-default-value, CASMO_scan, and CASMO_read. CASMO-Get-default-values is almost the same as the Helios- Get-default-values. In the subroutine Get-default-value, the dimensions and several control flags are set to default values according to the assembly type, fuel assembly or reflector. The only difference is that only two group cross sections are converted to PMAXS from CASMO in the current version of GenPMAXS. In the subroutine CASMO_scan, the information about the dimensions of the XS data and burnup points, and the information about the state values for branches and histories are scanned. The PMAXS branch structure is built using the input or scanned branch information and the memory is allocated according to the dimensions. In the subroutine CASMO_read, the XS data are read by first checking block names in sections 7.2. There is only one post processing operation, extract Xe/Sm absorption, which been implemented into the CASMO_read. The other two post processing steps, (1) correct the down scattering data, (2) generate ADF for reflector, will be similar to the HELIOS_read and will be called by other module such as CASMO_read. The XS data are stored in the PMAXS data structure and prepared for generating partial derivatives and for printing into the PMAXS file.

7.5. Benchmark II

In order to verify the procedures used in GenPMAXS, PMAXS, and the depletion module in PARCS, the same PARCS model shown in Figure 6.4 was used to reproduce the k-infinite of assemblies with PMAXS generated from lattice codes with an infinite spectrum. In this benchmark, the PWR fuel assembly from the previous example was used. The k-infinite from CASMO and PARCS for several branches in the reference history and for all branches in non-reference branches are shown in Figures 7.2 to 7.5. The differences between the k-infinite from CASMO and PARCS for all branches at all burnup points are within 1 pcm. The CAX file and input and output files of GenPMAXS and PARCS for this benchmark are provided in the GenPMAXS repository.

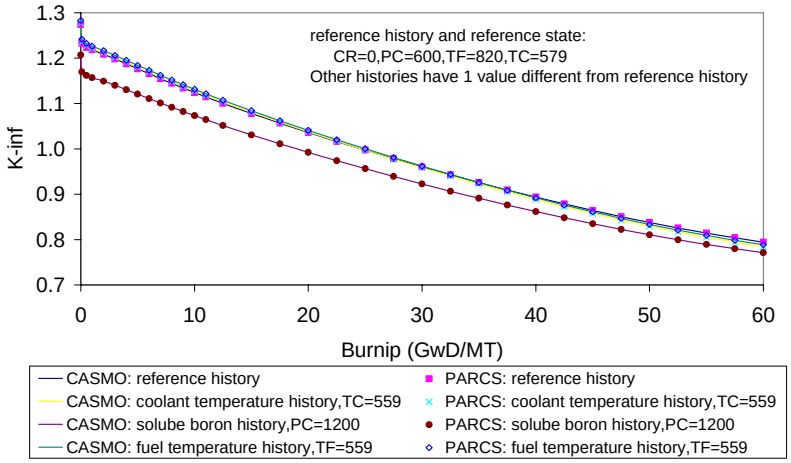


Figure 7.2. PARCS and CASMO K-inf for base branches of different histories in Benchmark II

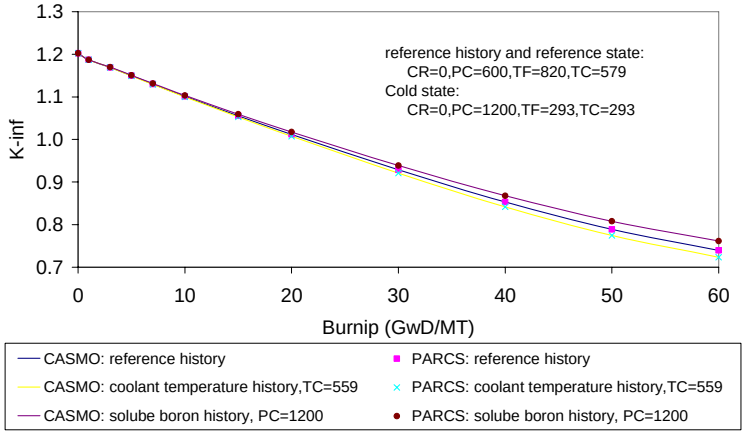


Figure 7.3. PARCS and CASMO K-inf for cold branches of different histories in Benchmark II

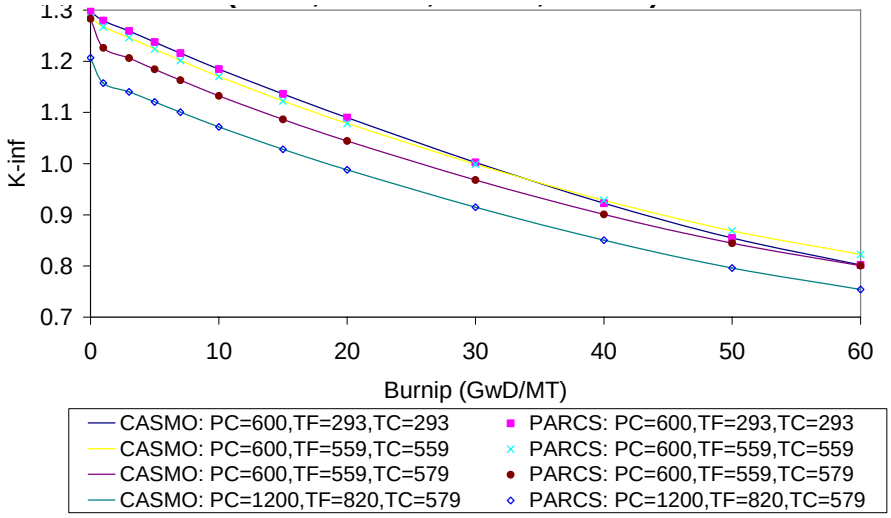


Figure 7.4. PARCS and CASMO K-inf for unrodded branches of reference history in Benchmark II

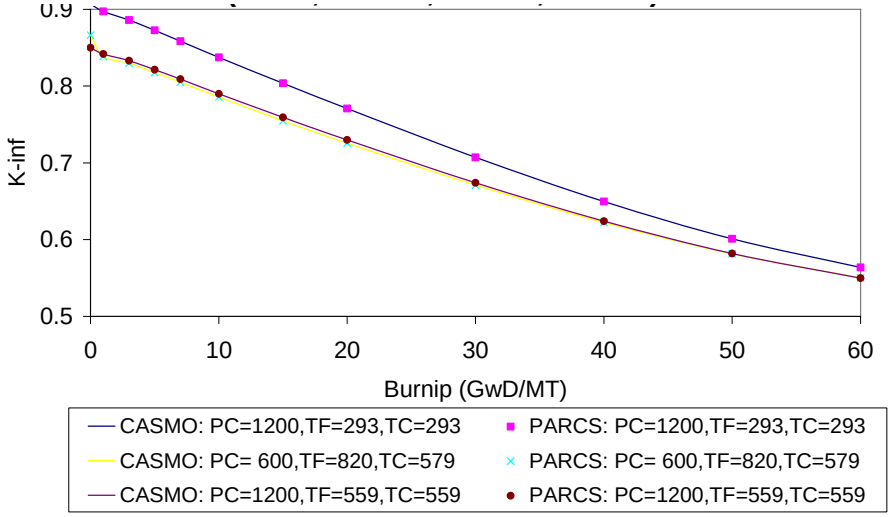


Figure 7.5. PARCS and CASMO K-inf for rodded branches of reference history in Benchmark II

APPENDIX A PMAXS and XSEC Format

(version 2.0, revision-01, 5/18/05)

The PMAXS file contains data for one set of cross sections, which may have one or more history cases and burnup points. The XSEC file contains XS for one or more sets. Only 3 cards, i.e. Burnup Information, History case identification and Burnup point identification, are needed in PMAXS. The NSET in XS Control Information means number of history cases in PMAXS and indicates the number of XS sets, everything else is common.

			Existence
1	XS Control Information		Always
2	Branches Information		Optional
3	Burnup Information		PMAXS
XS Set/(History case) wise data			Always
4	XS Set identification		Always
5	History case identification		PMAXS
6	T/H invariant variable block(repeat for burnup)		Always
	6.1	Chi, Chid, inV	Optional
	6.2	Yield	Optional
	6.3	Beta of Delayed neutron	Optional
	6.4	Lambda of Delayed neutron	Optional
	6.5	Decay heat data	Optional
	Reference state data		Always
7	State identification		Always
8	XS Data Block (repeated for burnup points)		Always
	8.1	Principal cross sections (tr,ab,nf,kf,fi,chi,inv,xe,sm,det)	Always
	8.2	Scattering cross sections	Always
	8.3	ADF	Optional
	8.4	Direct energy deposition and J1 factors	Optional
	8.5	CDF	Optional
	8.6	GFF	Optional
	Ith type branches (same structure with Ref. state case)		NBR(i)>0

*The data in XS Block are original data for reference state, and partials for other branches

1) XS Control Information

```
Format:(A, 8I,15L/5(A/))
Fields:Title NSET NGROUP MDLAY MDCAY MADF MCDF MRODS MCOLA
       ladf lxes lded lj1f lchi lchd linv ldet lyld lcdf
       lgff lbet lamb ldec Derivative/
       Comment1/Comment2/coment3/comment4/comment5/
Description:
  Title       ='GLOBAL_V'
  NSET   number of history cases in PMAXS. number of XS sets
         in XESC.
  NGROUP Number of energy groups.
  MDLAY  Maximum number of delay neutron groups for all XS
         set.
  MDCAY  Maximum number of decay Heat groups for all XS set.
  MADF   Maximum number of ADF in each group for all XS set.
  MCDF   Maximum number of CDF in each group for all XS set.
  MRODS  Maximum number of rods in computed part of assembly
         for all XS set.
  MCLOA  Maximum number of rod columns in whole assembly for
         all XS set.
```

The following logical flags indicate PMAXS contains the corresponding data, if it is 'F', then default values, which are given in following table, will be used in PARCS.

```
  Ladf   Assembly discontinuity factor
  Lxes   Microscopic cross section of Xe and SM
  Lded   Direct energy deposition fraction,default value 0
  Lj1f   J1    factor    for    minimal    critical    power
         ratio,default:1
  Lchi   Fission spectrum, default X(1)=1
  Lchd   Delay neutron fission spectrum, default Xd(i)=X(i)
  Linv   Inverse velocity
  Ldet   Detector response XS, no default
  Lyld   yield values of I, XE, Pm, the default values:
         0.06386,0.00228,0.0113
  Lcdf   Corner discontinuity factor, default 1
  Lgff   Group wise power form function, default 1
  Lbet   Beta, default:0.0002584,0.00152, 0.0013908
                     0.0030704,0.001102,0.0002584
  Lamb   Lambda, default: 0.0128,0.0318,0.119,
                          0.3181,1.4027,3.9286
  Ldec   Decay heat beta and lambda, default:
         Beta: 2.35402E-02,1.89077E-02,1.39236E-02
               6.90315E-03,3.56888E-03,3.31633E-03
         Lamb: 1.05345E-01,8.37149E-03,5.20337E-04
               4.73479E-05,3.28153E-06,1.17537E-11
  Derivative   T/F: Data store in non-reference branches are
         partial derivatives or raw cross sections. Default
         value: T
  Comment1~ Comment5     five line comments for describing
         content of PMAXS
```

2) Branches information

```
Format:( A8,I4,30(1x,A2)/
          A8,30I4/
         /(4x,A2,I4,7F12.5/20(10x,7F12.5/))
Fields:Title1,Nsta_var, Var_nam(1:Nsta_var)
       Title2,IST,NBR(1:Nsta_var)
       (name,ind, state(i,1: Nsta_var), i=1,NBRA)
Description:
  Title1 ='STA_VAR'
       If there is no Title1 before Title2, then Nsta_var
       will be set to 5, and Var_name will be
       'CR','DC','PC','TF',and 'TC'.
  Nsta_var number of state variables
  Var_nam  name of state variables

  Title2 ='BRANCHES'
  IST    Branches structure index
  NBR(i) branch cases for ith state variable.
         NBRA=1+sum(NBR)
  Name   name of branch, for user only
  Ind    index of branch, for user only
  state  values for state variables
```

If this block is default, then Nstat_var=1, Var_nam='CR', NBR(1)=0, NBRA=1

3) Burnup information

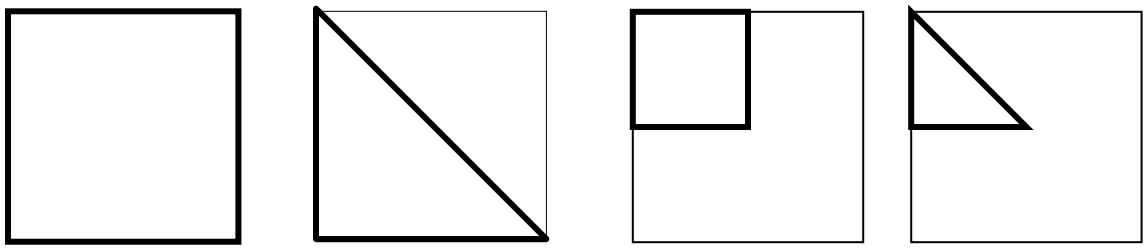
If this card is default, then NBset=1, NBP(1)=1,BURN(1,1)=0

```
Format:(A8,I4/(2I4,10F8.3/(8x,10F8.3)))
Fields:Title,NBset/
       (i,NBP(i),(Burn(j,i),j=1, NBP(i)))
Description:
  Title  ='BURNUPS'
  NBset  number of Burnups sets, default=1
  I      index for Burnups set
  NBP(i) Burnup points in Burnups set i
  Burns(j,i)   Burnup values.
```

4) XS Set identification

```
Format:(A,7I,8F)
Fields:Title,Series,IST,NADF,NCDF,NCOLA,NROWA,NPART,
       PITCH,XBE,YBE,iHMD,Dsat,ARWatR,ARByPa,ARConR
Description:
  Title  ='XS_SET'
  Series XS set series number
  IST    Branches structure index
  NADF   Number of ADF in each group.
  NCDF   Number of CDF in each group.
  NCOLA  Number of rod columns in whole assembly
  NROWA  Number of rod rows in whole assembly
  NPART  Index for computed part of assembly
```

```
        0/1/2/3:   whole/half/quarter/eighth
        See next picture
  PITCH  rod lattice pitch(cm)
  XBE    start position for first column rods
  YBE    start position for first row rods
  iHMD   initial heavy metal density (g/cc)
  Dsat   the saturated moderator density
  ARWatR the area ration of water rods to coolant
  ARByPa the area ration of bypass to coolant
  ARConR the area ration of control rods to coolant
```



5) History case identification

Only needed in PMAXS

```
Format:( A8,I4,7F12.5/,20(12x,7F12.5/))
Fields:Title,ibset,History
Description:
  Title  ='HISTORYC'
  Ibset    burnup set index for TH invariant block
  History(1:Nsta_var)    History variables.
```

The old format with title "HST_CASE" as following is still acceptable for GenPMAXS and PARCS. Burnup set index for TH invariant block is not included in old format, and will be set to 1.

```
Format:( A8,7F12.5/,20(10x,7F12.5/))
Fields:Title, History
Description:
  Title  ='HST_CASE'
  History(1:Nsta_var)    History variables.
```

6) T/H invariant variable block

This block contains 7 subblocks, repeats for all burnup points

The format of T/H invariant variable block and XS block are depend on NGROUP as following:

NGROUP	Format
n<3	8E12.5
n=3	6E12.5
n=4	8E12.5
n>4	nE12.5

6.1 Chi, Chid, inV

```
Fields:Chi(1:n),Chid(1:n),inV(1:n)
       (where n=NGROUP)
Description:
  Chi    fission neutron spectrum
  Chid   Delay fission neutron spectrum
  inV    inverse of netron velocity
```

6.2 Yield of I, Xe, and Pm

```
Fields:    YLDI, YLDXe, YLDPm
Description:
  YLDI   Effective Iodine Yield
  YLDXe  Effective Xenon Yield
  YLDPm  Effective Promethium Yield
```

6.3 Beta of Delayed neutron

```
Fields:BETA(1:NDLAY)
Description:
  BETA   Effective delayed neutron fraction
```

6.4 Lambda of Delayed neutron

```
Fields:LAMBDA(1:NDLAY)
Description:
  LAMBDA Decay constant of delayed neutron (/sec).
```

6.5 Decay heat data

```
Fields:DBET(1:NDCAY)
       DLAM(1:NDCAY)
   Description:
  DBET   Fraction of the total fission energy appearing as
         decay heat for decay group i.
  DLAM   Decay constant of decay heat group i. [/sec]
```

7) State identification

```
Format:(A8, 2I4)
Fields:Title, index, IBSET
Description:
  Title  ='REFERENC'/'CRBRANCH'/'MDBRANCH'/'SBBRANCH'
          /'TFBRANCH'/'TMBRANCH'/
  index  Branch case index
  IBSET  Burnups Set index
```

8) XS data block

This block contains 4 subblocks, repeats for all burnup points
The XS block are depend on NGROUP as following:

NGROUP	Format
n<3	8E12.5
n=3	6E12.5
n=4	8E12.5
n>4	nE12.5

8.1 Principal cross sections

```
Fields:STR(1:n),SAB(1:n),SNF(1:n),SKF(1:n),XENG(1:n),SM
       NG(1:n),SFI(1:n),Det(1:n)
Description:
  STR    Transport cross section
  SAB    Absorption cross sections.
  SNF    Nu-fission cross section
  SKF    Kappa-fission cross section
  XENG   Microscopic capture cross section of Xenon
  SMNG   Microscopic capture cross section of Samarium
  SFI    Fission cross section
  Det    Detector response parameter, it is product of
         cross section and local flux ratio.
```

8.2 Scattering cross sections

```
Fields:((SCT(j,i),j=1,NGROUP),i=l,NGROUP)
Description:
  SCT(j,i)     scattering xs from group j to group i.
```

8.3 ADF

```
Fields:((ADF(g,n),g=1,NGROUP) ,j=1,NADF)
Description:
  ADF    Assembly discontinuity factor.
         For Cartesian:
          If NADF=4: j=1/2/3/4  = W/S/E/N
          If NADF=2: j=1/2      = W/S
          If NADF=1: j=1        = average
```

8.4 Direct energy deposition and J1 factors

```
Format: (8E12.5)
Fields:   DED(1:4),J1F(1:4)
Description:
  DED(1) the ratio of DED fraction in coolant and its
         density
  DED(2) the ratio of DED fraction in water rod and its
         density
  DED(3) the ratio of DED fraction in bypass and its
         density
  DED(4) DED fraction in control rod.
  J1F(1) Local Power Peaking Factor.
  J1F(2) Inner  J1 factor (for Critical Power Ratio)
  J1F(3) Side   J1 factor (for Critical Power Ratio)
  J1F(4) Corner J1 factor (for Critical Power Ratio)
```

8.5 CDF

```
Fields:(CDF(g,j),g=1,NGROUP,j=1,NCDF)
Description:
  CDF    Corner discontinuity factor.
         For Cartesian:
          If NCDF=8: j=1/2/3/4  = NW/SW/SE/NE
                     /5/6/7/8   /W /S /E /N
```

```
If NCDF=5: j=1/2/3     = NW/SW/SE
            /4/5        /W /S
If NCDF=4: j=1/2/3/4   = NW/SW/SE/NE
If NCDF=3: j=1/2/3     = NW/SW/SE
If NCDF=2: j=1/2       = corner/mid
If NCDF=1: j=1         = corner
```

8.6 group-wise form function

```
Fields:((GFF(g,j),g=1,NGROUP),j=1,NRODS)
Description:
  GFF    group-wise form function from left to right,
         from top to bottom. It is product of pin flux
         and fission cross section.
```

Appendix B Input Manual of GenPMAXS

1. Input Description

GENPXS uses input files which follow the standard I/O rules of Windows 98/NT or the UNIX system. There are several keywords used in the GENPXS input file to identify the special meaning of the input followed by the data. The Table D.1 shows the keywords and the meanings of the data.

Table B.1. The keywords of GenPMAXS input and their meaning.

Index	keyword	Meaning
1	%JOB_TIT	Problem title
2	%JOB_OPT	The options of GenPMAXS program.
3	%DAT_SRC	The Files contain XS data.
4	%STA_VAR	State variables.
5	%HISTORY	History informations.
6	%BRANCHES	Numbers of branches, and State information all branches
7	%REFEREN	Desired reference state values
8	%BURNUP	Burnup points
9	%HEL_FMT	Format of HELIOS output file
10	%FIL_CNT	XS file contents
11	%JOB_END	Job ending flag.

In the GENPXS input uses all data is given in free format except for the keywords. The cards should be given with the order given in Table D.1. If they are not in the correct order, the subsequent cards will be not used in the code.

1.1. Job Title

- Format

```
    %JOB_TIT
    PMAXS_file Oderivative comments
Format: (A L A)
```

Variable	Description
PMAXS_file	The name of the output PMAXS format file. The maximum length is 40.
Oderivative	T/F: Data store in non-reference branches of output PMAXS are partial derivatives or raw cross sections.
Comments	Character.

- Example

```
%JOB_TIT
    PMAXS.C01 T “17x17” SMART CORE FUEL ASSEMBLY "08/03/2000"
```

In the above example, in order to distinguish the integer and character-string, the character strings starting with a numeric data are enclosed by double quotation marks. The output PMAXS format file will be created in the name of PMAXS.C01. Data store in non-reference branches of output PMAXS are partial derivatives

1.2. Job Option

- Format

```
%JOB_OPT
Ladf Lxes Lded Lj1f Lchi Lchd Linv Ldet Lyld Lcdf Lgff Lbet Lamb Ldec iups
```

The logical flags indicate write or not write corresponding data in to PMAXS file. If the flag is ‘F’, the PMAXS will not contain the corresponding data, and default values, which are given in following table, will be used in PARCS. If the values from XS file are same as default values, user may take ‘F’ to reduce computation cost.

Regardless what the values are given in input file, all logical flag except Ladf will be force to be ‘F’ for reflector.

Variable	Description
Ladf	Assembly discontinuity factor
Lxes	Microscopic cross section of Xe and SM, No default value
Lded	Direct energy deposition fraction, default value 0
Lj1f	J1 factor for minimal critical power ratio, default:1
Lchi	Fission spectrum, default X(1)=1
Lchd	Delay neutron fission spectrum, default Xd(i)=X(i)
Linv	Inverse velocity, no default value, must be ‘T’ for transient
Ldet	Detector response XS, no default
Lyld	yield values of I, XE, Pm, the default yield values are:0.06386,0.00228,0.0113
Lcdf	Corner discontinuity factor, default 1
Lgff	Group wise power form function, default 1
Lbet	Beta, default:0.0002584,0.00152,0.0013908 0.0030704,0.001102,0.0002584
Lamb	Lambda, default: 0.0128,0.0318,0.119,0.3181,1.4027,3.9286
Ldec	Decay heat beta and lambda, default: Beta: 2.35402E-02,1.89077E-02,1.39236E-02 6.90315E-03,3.56888E-03,3.31633E-03 Lamb:1.05345E-01,8.37149E-03,5.20337E-04 4.73479E-05,3.28153E-06,1.17537E-11
Iups	Integer 0 keep up scatter XS 1 remove up scatter XS, modify down scatter XS with HELIOS spectrum 2 remove up scatter XS, modify down scatter XS with infinite medium spectrum

- Example

```
%JOB_OPTION
    T T F F F F F F F F F F F 2

If the source cross section files are PMAXS, SRC_Kind in DAT_SRC
card is 1, then values of variables in this card will be assigned to
those value in first PMAXS file.
```

1.3. Data Source

- Format

```
%DAT_SRC
    SRC_kind Nfile kind_FA sfac
```

Variable	Description
SRC_kind	Integer, type of source files which contain XS IF 1, PMAXS files IF 2, HELIOS outputs IF 3, CASMO outputs If the XS files are PMAXS files, only %JOB_TIT, %JOB_OPT %DAT_SRC, %FIL_CNT and %JOB_END cards are needed
Nfile	Integer, number of XS files.
FA_kind	Integer. The flag of assembly type. IF 1, fuel assembly. If 0, reflector.
Sfac	Real. The scattering cross section factor. If ‘sfac’ is different from 1, then the scattering cross section will be multiply by ‘sfac’. This factor is designed for generate cross section for corner reflector as described in section 3.2. The modification factor can be computed as Eq. (3.26). For example, the fuel pitch and the thickness of shroud are 21.607 and 2.23398 cm, respectively. So the correction factor is 0.8966

- Example

```
%DAT_SRC
    2   1   1   1.0
```

In the above example, there is 1 XS file which contains the HELIOS results for a fuel assembly with no correction for scattering cross sections.

1.4. State variables

- Format

```
% STA_VAR
   Nstat_var
   Var_name(1:Nstat_var)
```

Variable	Description
Nstat_var	Integer. Number state variable, default Nstat_var=5
Var_name	Character*2, Names of state variables. There are 12 acceptable state variables: 1. CR control rod fraction 2. DC density of coolant 3. PC soluble poison concentration in coolant 4. TF temperature of fuel 5. TC temperature of coolant 6. IC impurity of coolant 7. DM density of moderator 8. PM soluble poison concentration in moderator 9. TM temperature of moderator 10. IC impurity of moderator 11. DN density difference between neighbor and current assembly 12. BN burnup difference between neighbor and current assembly These state variables must be given in this designed order. The default variables are first 5 variables. CR must be in Variable name list as first variable, even when there is no CR branch and no any CR dependent.

- Example

```
%STA_VAR
  4
  CR DC PC TF
```

In above example, control rod fraction, density of coolant, soluble poison in coolant and temperature of fuel are 4 state variables which will be considered in output PMAXS.

1.5. Histories

- Format

```
% HISTORY
   NHST ihref
  (label(i), history(1:Nstat_var,i),  i=1,NHST)
```

Variable	Description
NHST	Integer. Number history cases, when this card is default, NHST=1 if branches card is presented,

	otherwise NHST will determined by scan data source files.
ihref	Integer. Histroy ihref will be reference history, if default, there the history with history state values most close to eference branch will be selected as reference history.
Label(i)	charactors. Description of ith history.
History(:,i)	real, the value of history variables for ith history case. If this card is default, the values of state variables of reference case given in %BRANCH card will be used as history variables values.

- Example

```
%HISTORY
  1   1
  hist1  0.000000  0.456652     0.000   933.000
```

In above example, there is only 1 history case for this PMAXS, and values of 4 history variables, which corresponding 4 state variables given in %STA_VAR card, are given.

1.6. Branches

- Format

```
%BRANCH
   NBRA  ibref
   ( Bname(i) state(1:Nstat_var,i), i=1,NBRA)
```

Variable	Description
NBRA	Integer. Number of total branches include reference branch. If the SRC_kind==3, then this card can be default, Information for branches and histories will be scanned from CAX files.
ibref	Integer. branch ihref will be reference branch. If default, the branch which is most close to desired reference state will be selected as reference branch
Bname(i)	Characters. description of ith branch.
state(:,i)	real. values of state variables for ith branch

- Example

```
%BRANCH
     11 1
  1REFE1  0.000000  0.456652     0.000   933.000
  2CRBR1  1.000000  0.456652     0.000   933.000
  3DCBR2  0.000000  0.177504     0.000   933.000
  4DCBR2  0.000000  0.317078     0.000   933.000
```

```
  5DCBR3  0.000000  0.596226     0.000   933.000
  6DCBR4  0.000000  0.7358       0.000   933.000
  7PCBR1  0.000000  0.177504  1000.000   933.000
  8PCBR2  0.000000  0.456652  1000.000   933.000
  9PCBR3  0.000000  0.7358    1000.000   933.000
 10TFBR1  0.000000  0.456652     0.000   561.220
 11TFBR2  0.000000  0.456652     0.000  2000.000
```

There are 1 branch for control rod, 4 branches for coolant density, 3 branches for soluble boron in coolant, and 2 branches for fuel temperature. There are 11 branches in total included reference state. The burnup points for CR, DC, PC and TF branches are all same, and there are different from reference state.

1.7. Desired Reference State

- Format

```
%REFEREN
   rstate(1:Nstat_var)
```

Variable	Description
Rstate	real. Desired state values for reference state. The default values are CR DC PC TF TC IC DM PM TM 0 0.71 600 800 580 0 0.71 600 580 These values are good for PWR. The following values are suggested for BWR: CR DC PC TF TC IC DM PM TM 0 0.46 0 800 560 0 0.74 0 560

- Example

```
% REFEREN
  0.000000  0.456652     0.000   933.000
```

Four state values are given for CR, DC, PC and TF.

This card is used only when %BRANCHES card is not provided in input file.

1.8. Burnup

- Format

```
% BURNUP
    NBSET
   (  bname(i) NBURN(i)
      burn(1:NBURN(i),i), i=1:NBSET )
   (Hname(j) ihbset(1:NBRA,j), j=1,NHST)
```

Variable	Description
NBSET	Integer, |NBSET| is number of burnup points set. If NBSET>0, all burnup points need be given in this card, otherwise they will be read from XS file. For some case, NBSET must be positive number, see details in %FIL_CNT.
bname(i)	charactors, descriptions for burnup set i
NBURN(i)	Integer. Number of points in ith burnup points set.
Burn(:,i)	Real. The value of burnups in ith burnup points set.
hname(j)	charactors, descriptions for jth history
ihbset(k,j)	Integer. Burnup set index for kth branch in jth history, hname and ihbset should be provided if branches card is provided and NBSET>0

- Example 1

```
%BURNUP
  -2
```

In the above example, there are 2 sets of burnup points which will be read from XS files.

- Example 2

```
%BURNUP
   2
   bset1  34
0.000   0.200   0.500   1.000   2.000   3.000   4.000   5.000   6.000
7.000   8.000   9.000  10.000  11.000  12.500  15.000  17.500  20.000
22.500  25.000 27.500  30.000  32.500  35.000  37.500  40.000  42.500
45.000 47.500  50.000 52.500  55.000  57.500  60.000
    bset2  12
0.000   0.500   1.000   3.000   5.000   7.000  10.000  15.000  20.000
30.000 45.000  60.000
   HIST01   1  10*2
```

In the above example, there are 2 sets of burnup points. The number of burnup points are 34 and 12 for first and second sets respectively, and the reference branch has 34 burnup points as burnup set 1, all other branches has 12 burnup points specified in burnup set 2.

1.9. Helios Output File Format

- Format

```
%HEL_FMT
    NFMT
    (i Label(i) Width(i) Column(i), i=1,NFMT)
```

Variable	Description

NFMT	Integer. Number of format.
i	Integer. Index of format
Label(i)	Integer. Width of label column in Characters.
Width(i)	Integer. Width of each date columns in Characters.
Column(i)	Integer. Number of date columns in one block.

- Example

```
%HEL_FMT
    1
    1 23 13 8
```

In the above example, there is 1 format for HELIOS output files. The label takes 24 columns, and there are at most 8 data in a row, the width for each data is 13 columns.

1.10. File Content

- Format

```
%FIL_CNT
( i  Fname(i) nb(i), ifmt(i)
 (bn(j) hst(j,i) brn(j,i) bb(j,i) be(j,i), j=1:nb(i)) ,i=1:Nfile)
```

Variable	Description
I	Integer. Index of XS file. Not used in code
Fname(i)	Characters. Name of ith XS file.
nb(i)	Integer. This number is used for HELIOS output only If nb(i)>0, nb(i) indicates number of XS blocks in ith XS file. In HELIOS output, data for one branch are often printed together, we call it a block. So a block is often contains all of data for one branch. But in some case, data for different for burnup in one branch may be printed separately, so one branch may need several blocks. If any one of nb(i)>0, then NBSET in %BURNUP card must be a positive number. If nb(i)>0, The history, branch and burnup points information for each block should be given in this card, otherwise these information should not present in this card. If nb(i)<0, then this XS file contains all of data for |nb(i)|th history case. If the XS file is HELIOS output, each block contains all of data for one branch. If nb(i)=0, then this XS file contains all of data for all history cases. There must be only one XS file for this PMAXS. If the XS file is HELIOS output, each block contains all of data for one

	branch.
Ifmt(i)	Integer. Index of file format, only used when XS files are HELIOS output.
bn(j,i)	String, name of jth block
hst(j,i)	Integer. Index of History case which jth block contains its data. If hst(j,i)<=0, skip jth block
brn(j,i)	Integer. Index of branch which jth block contains its data.
bb(j,i)	Integer. Index of first burnup points in jth block.
be(j,i)	Integer. Index of last burnup points in jth block.

- Example 1

```
%FIL_CNT
   1  'Zenith.out' 11 1
      1  1  1  1 34
      2  1  2  1 12
      3  1  3  1 12
      4  1  4  1 12
      5  1  5  1 12
      6  1  6  1 12
      7  1  7  1 12
      8  1  8  1 12
      9  1  9  1 12
     10  1 10  1 12
     11  1 11  1 12
```

In the above example, there is 1 HELIOS output files which contains 11 blocks. The ith block contains data for ith branch. All of these blocks are for history case 1. All of these blocks begin from burnup point 1. The first block ends at burnup point 34, the rest blocks end at burnup points 12.

- Example 2

```
%FIL_CNT
   1  'Zenith.out' -1 1
```

In the above example, there is 1 HELIOS output files which contains all data for history case 1. If the XS file is HELIOS output, each block contains all of data for one branch.

- Example 3

```
%FIL_CNT
   1  'Zenith.out' 0 1
```

In the above example, there is 1 HELIOS output files which contains all data for all history cases. If the XS file is HELIOS output, each block contains all of data for one branch.

2. Sample Inputs for GenPMAXS

2.1. The Sample Inputs for HELIOS

Example 1

```
%JOB_TIT
 'bwr_hel.PMAX'  T  BWR 10X10 FUEL ASSEMBLY
%JOB_OPT
   T  T  T  F  F  F  T  F  T  F  F  T  T F   1
 !ad,xe,de,j1,ch,Xd,iv,dt,yl,cd,gf,be,lb,dc,ups
%DAT_SRC
    2   1   1   1.0
%STA_VAR
  4
  CR DC PC TF
%HISTORY
  1  1
  1  0.000000  0.456652     0.000   933.000
%BRANCH
     11   1
  1REFE1   0.000000  0.456652     0.000   933.000
  2CRBR1  1.000000  0.456652     0.000   933.000
  3DCBR1  0.000000  0.177504     0.000   933.000
  4DCBR2  0.000000  0.317078     0.000   933.000
  5DCBR3  0.000000  0.596226     0.000   933.000
  6DCBR4  0.000000  0.7358       0.000   933.000
  7PCBR1  0.000000  0.177504  1000.000   933.000
  8PCBR2  0.000000  0.456652  1000.000   933.000
  9PCBR3  0.000000  0.7358    1000.000   933.000
 10TFBR1  0.000000  0.456652     0.000   561.220
 11TFBR2  0.000000  0.456652     0.000  2000.000
%BURNUP
   2
   Bset1  34
         0.000   0.200   0.500   1.000   2.000   3.000   4.000   5.000   6.000   7.000
         8.000   9.000  10.000  11.000  12.500  15.000  17.500  20.000  22.500  25.000
        27.500  30.000  32.500  35.000  37.500  40.000  42.500  45.000  47.500  50.000
        52.500  55.000  57.500  60.000
   Bset2  12
         0.000   0.500   1.000   3.000   5.000   7.000  10.000  15.000  20.000  30.000
        45.000  60.000
   HIST01  1   10*2
%HEL_FMT
    1
    1 23 13 8
```

```
%FIL_CNT
  1  'Zenith.out' 11 1
     1  1  1  1 34
     2  1  2  1 12
     3  1  3  1 12
     4  1  4  1 12
     5  1  5  1 12
     6  1  6  1 12
     7  1  7  1 12
     8  1  8  1 12
     9  1  9  1 12
    10  1 10  1 12
    11  1 11  1 12
%JOB_END
```

In above example, the source XS file is a HELIOS output called 'Zenith.out'. There is only 1 history case in this file which contains 11 branches in 11 blcoks respectively. There two sets of burnup points, the reference state has 34 burnup points, all of other branches have 12 burnup points.

Example 2

```
%JOB_TIT
 'bwr_hel.PMAX'   T BWR 10X10 FUEL ASSEMBLY
%JOB_OPT
   T  T  T  F  F  F  T  F  T  F  F  T  T F   1
 !ad,xe,de,j1,ch,Xd,iv,dt,yl,cd,gf,be,lb,dc,ups
%DAT_SRC
    2   1   1   1.0
%STA_VAR
  4
  CR DC PC TF
%BRANCH
     11   1
  1REFE1   0.000000  0.456652     0.000   933.000
  2CRBR1   1.000000  0.456652     0.000   933.000
  3DCBR1   0.000000  0.177504     0.000   933.000
  4DCBR2   0.000000  0.317078     0.000   933.000
  5DCBR3   0.000000  0.596226     0.000   933.000
  6DCBR4   0.000000  0.7358       0.000   933.000
  7PCBR1   0.000000  0.177504  1000.000   933.000
  8PCBR2   0.000000  0.456652  1000.000   933.000
  9PCBR3   0.000000  0.7358    1000.000   933.000
 10TFBR1   0.000000  0.456652     0.000   561.220
 11TFBR2   0.000000  0.456652     0.000  2000.000
```

```
%BURNUP
 -2
%HEL_FMT
   1
   1 23 13 8
%FIL_CNT
  1  'Zenith.out' 0 1
%JOB_END
```

In above example, the source XS file is a HELIOS output called 'Zenith.out'. There is only 1 history case in this file which contains 11 branches in 11 blcoks respectively. There two sets of burnup points which will be read from 'Zenith.out'. the burnup set for each branch will be obtained by scan 'Zenith.out'.

2.2. The Sample Input to convert principal cross sections to partials

```
%JOB_TIT
 'Assembly.PMAX'   T
%DAT_SRC
   1   1   1   1.0
%FIL_CNT
   1 'Transformer.out' 0 0
%JOB_END
```

2.3.Sample Input to convert from old format to new format PMAXS

```
%JOB_TIT
 'bwr_hel.PMAX'  T
%DAT_SRC
   0    1  1  1.0
%FIL_CNT
  1  ' old.PMAX '  0 0
%JOB_END
```

2.4.Sample Input to generate corner reflector cross section

```
%JOB_TIT
   'corner.PMAX'   T  ‘corner reflector’
%DAT_SRC
   1    1   0   0.8966
%FIL_CNT
   1  'siderflector.PMAX' 0 0
%JOB_END
```

2.5. Sample Input for CASMO

Example 1

```
%JOB_TIT
 'CAX_PWR.PMAX' T  PWR Assyembly
%JOB_OPT
  T  F  T  F  F  F  T  T  T  F  F  T  T F   1
 !ad,xe,de,j1,ch,Xd,iv,dt,yl,cd,gf,be,lb,dc,ups
%DAT_SRC
   3   1   1   1.0
%STA_VAR
 4
 CR PC TF TC
%FIL_CNT
  1  'PWR.cax' 0 0
%JOB_END
```

In above example, the source XS file is a CASMO output called 'PWR.cax' for a PWR fuel assembly. 4 state variables, CR PC TF TC, are specified. Most of information, such as branches, histories, burnup sets and data block will be scanned from PWR.cax. The reference branch and reference history will be selected with default desired reference state values.

Example 2

```
%JOB_TIT
 'CAX_BWR.PMAX' T  PWR Assyembly
%JOB_OPT
  T  T  T  F  F  F  T  T  T  F  F  T  T F   1
 !ad,xe,de,j1,ch,Xd,iv,dt,yl,cd,gf,be,lb,dc,ups
%DAT_SRC
   3   1   1   1.0
%STA_VAR
 4
 CR DC PC TF
%REFEREN
 0.0 0.46 0.0 800.0
%FIL_CNT
  1  'BWR.cax' 0 0
%JOB_END
```

In above example, the source XS file is a CASMO output called 'BWR.cax' for a BWR fuel assembly. 4 state variables, CR DC PC TF, are specified. Desired reference state values are also specified for this case. Most of information, such as branches, histories, burnup sets and data block will be scanned from PWR.cax. The reference branch and reference history will be selected with specified desired reference state values.