Lattice Physics Calculations using the TRITON Sequence of SCALE

Mark D. DeHart, PhD

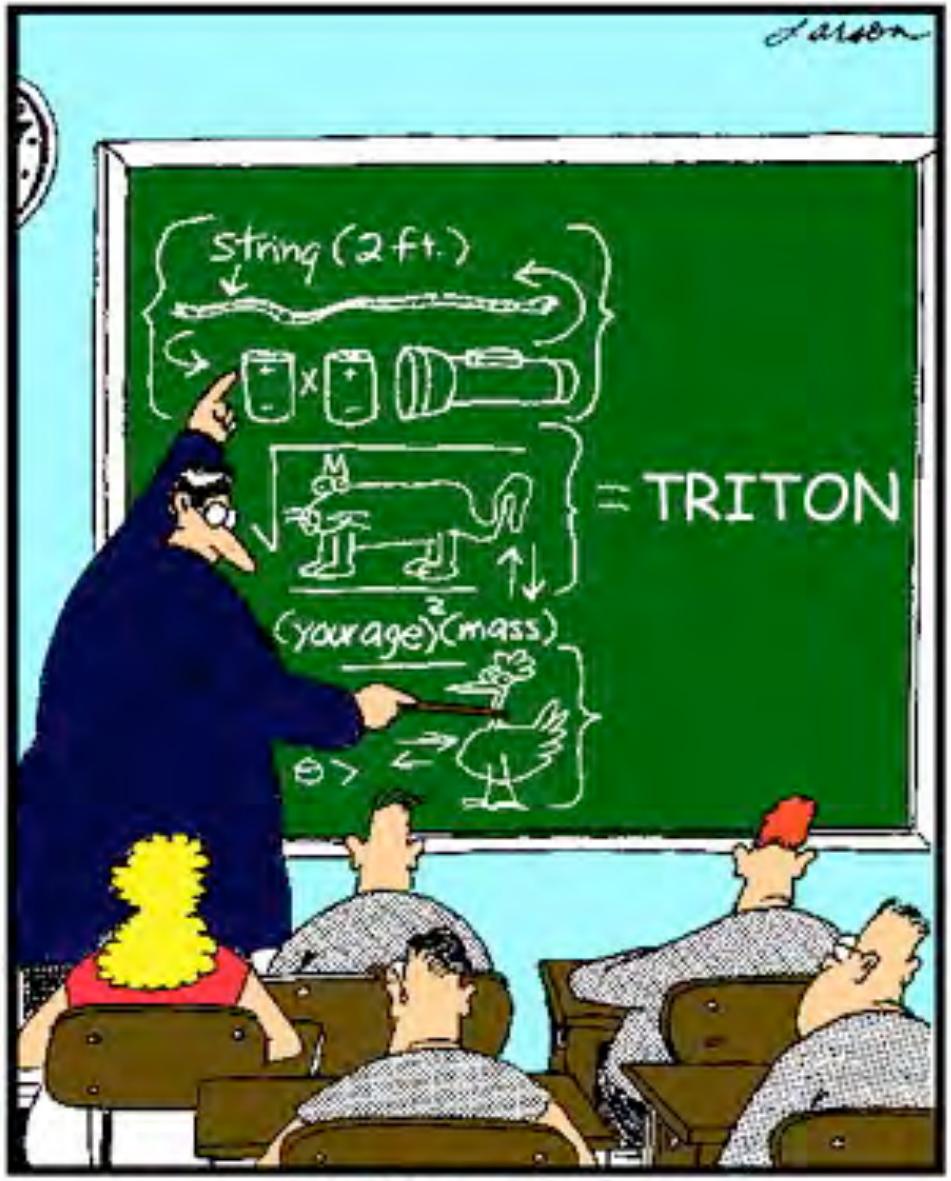
Senior R&D Staff, Reactor Analysis Group

Nuclear Science and Technology Division

Oak Ridge National Laboratory

Opening Words

- I will be providing an overview of TRITON capabilities, with NEWT transport modeling, primarily for lattice physics applications.
- I will be presenting SCALE 5.1 inputs formats - significant changes (improvements) have been introduced since the initial 5.0 release.
- Format will be informal - please stop me to ask questions at any time.
- This material is typically taught in a 3.5 day workshop - this tutorial may only whet your appetite.

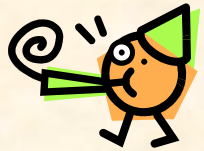


TRITON Explained

Overview

- Modular Nuclear Analysis Using SCALE
  - SCALE philosophy
  - Control modules
  - Functional modules
  - General capabilities
- The TRITON sequence
  - Functionality
  - MIPLIB processing
  - Cross section processing
  - Transport solution
  - Depletion using ORIGEN-S

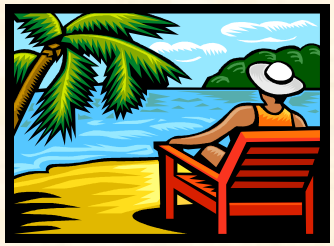
BREAK



Overview (continued)

- The NEWT discrete ordinates transport solver
  - Capabilities
  - Geometric specification
  - Grid control
  - Acceleration
  - Boundary conditions

BREAK



- TRITON depletion models
- Q and A

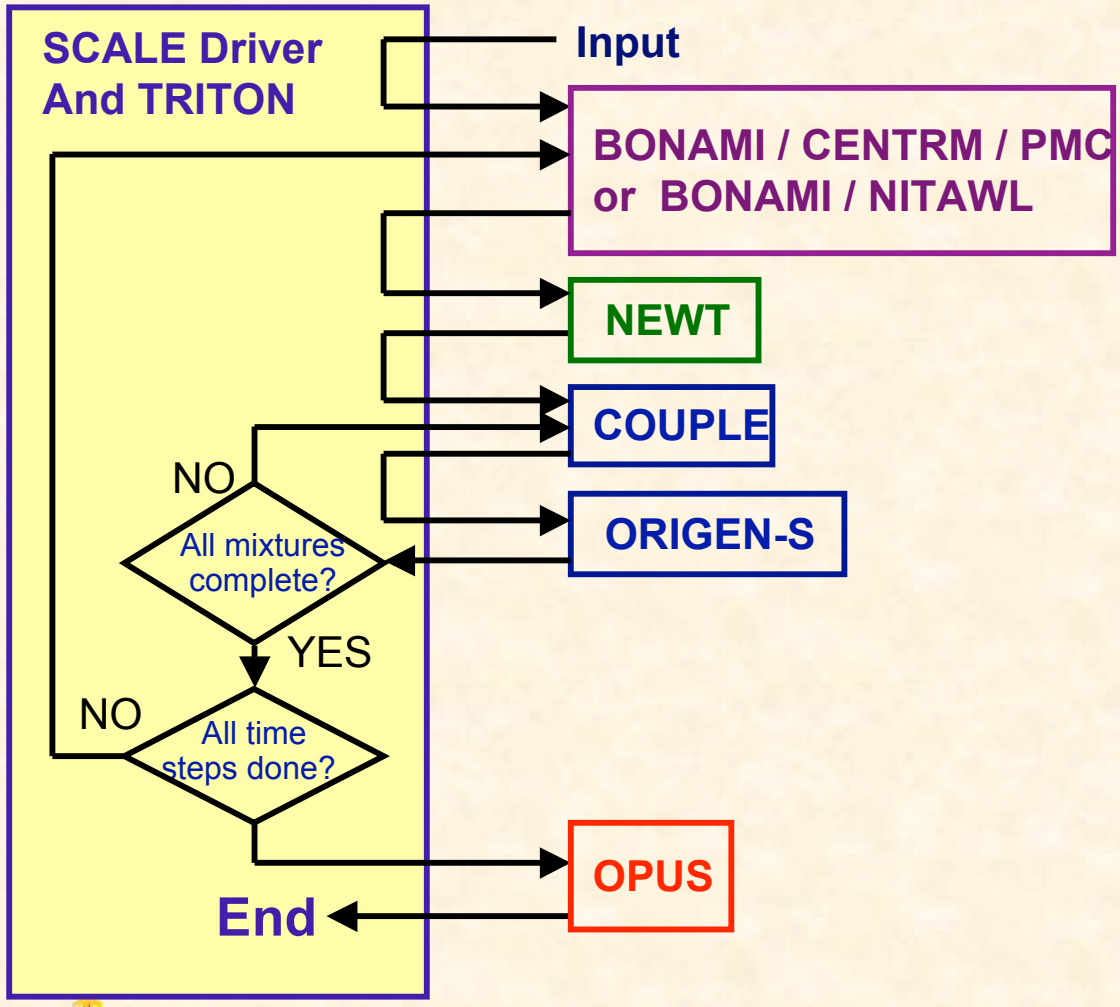
Modular Analysis using SCALE

- SCALE - Standardized Computer Analysis for Licensing Evaluation
  - Developed at Oak Ridge National Laboratory for the Nuclear Regulatory Commission (NRC)
  - Maintained/enhanced under co-sponsorship of NRC and Department of Energy (DOE)
  - A multi-purpose computational system for analyses of nuclear facilities and packages:
    - Criticality safety
    - Radiation source terms and shielding
    - SNF/HLW characterization (e.g., nuclide inventory)
    - Lattice physics

SCALE Features and Terminology

- Modular system of well-established and well-tested programs (functional modules) and data
- Control modules automate data processing and functional (system) analysis into sequences
- Engineering (not physics) parameters input with keywords and free format
- Designed for ease of use by new/occasional users, but with flexibility for expert users

SCALE Is Built on a Modular Concept



Resonance cross-section processing (repeat for all cells)

2-D discrete ordinates

Fuel depletion/decay (repeat for all depletion mixtures)

Output edits (repeat for all requested mixtures)

General Capabilities

- **Criticality Safety**
  - **Functional Modules:**
    KENO V.a, KENO-VI, NEWT, XSDRNPM
  - **Cross-section processing:**
    BONAMI/CENTRM, BONAMI/NITAWL
  - **Cross-section libraries:**
    Broad, Fine, and Point data, ENDF/B-V and -VI
  - **Control sequences:**
    csas25, csas26, csas1x, t-newt, smores, tsunami

General Capabilities (continued)

- **Radiation Shielding**
  - **Functional Modules:**
    MORSE-SGC, XSDRNPM, QAD-CGGP,
    Monaco (SCALE 6)
  - **Cross-section libraries:**
    Broad group ENDF/B-IV,
    Fine and broad group ENDF/B-VI (SCALE 6)
  - **Control sequences:**
    sas4, sas1, qads,
    mavric (SCALE 6)

General Capabilities (continued)

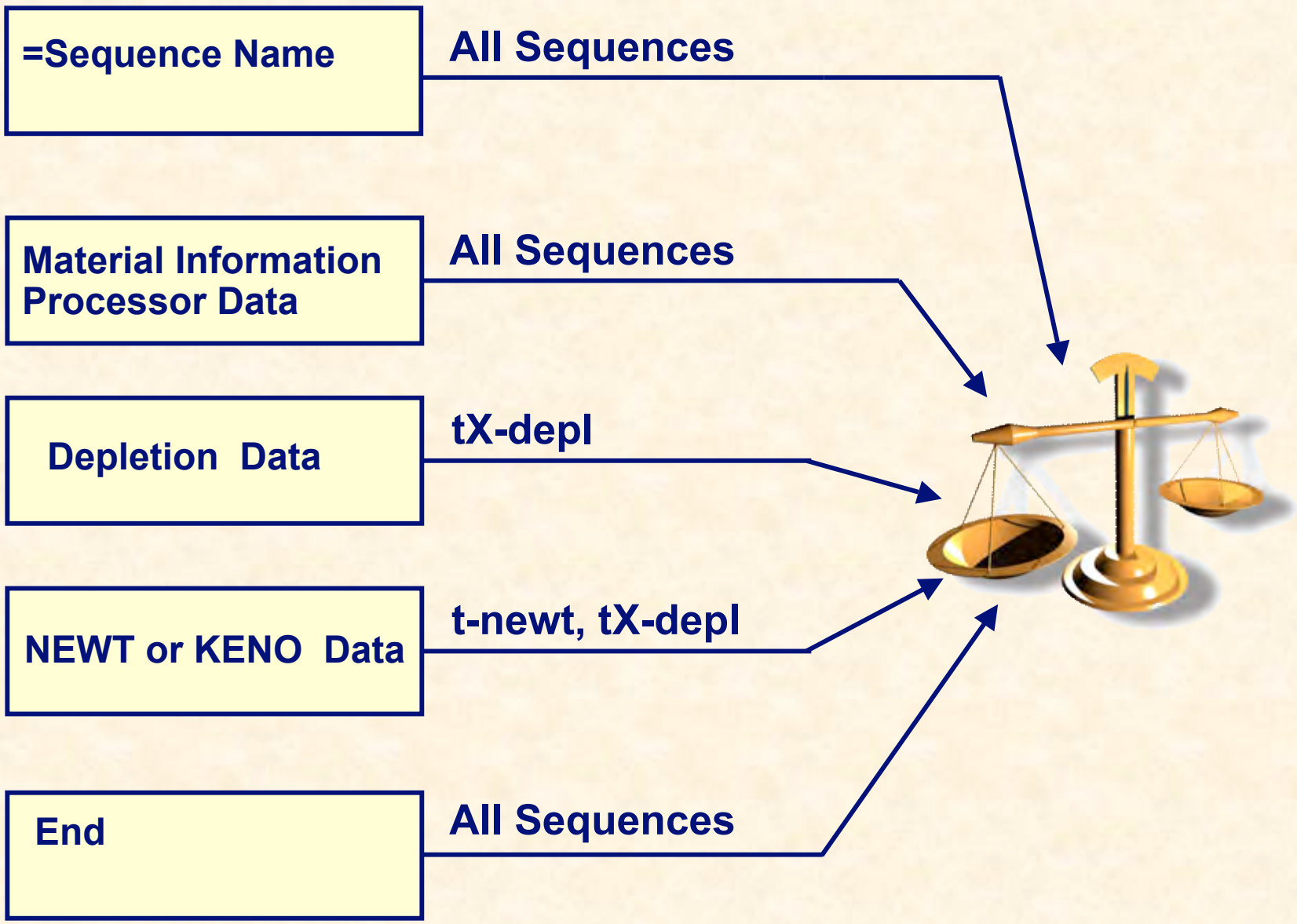
- **Spent Fuel Characterization**
  - **Functional Modules:**
    KENO V.a, KENO-VI, NEWT, XSDRNPM, ORIGEN-S
  - **Cross-section processing:**
    BONAMI/CENTRM, BONAMI/NITAWL, COUPLE
  - **Physics libraries:**
    Broad, Fine, and Point data, ENDF/B-V and -VI ORIGEN-S data:
    Fission yields: ENDF/B-VI
    Decay data: ENDF/B-VI and ENSDF
    Cross sections: ENDF/B-VI, EAF-99, and FENDL-2 pointwise data
  - **Control sequences:**
    sas2h, t-depl, t5-depl, t6-depl, starbucs

TRITON

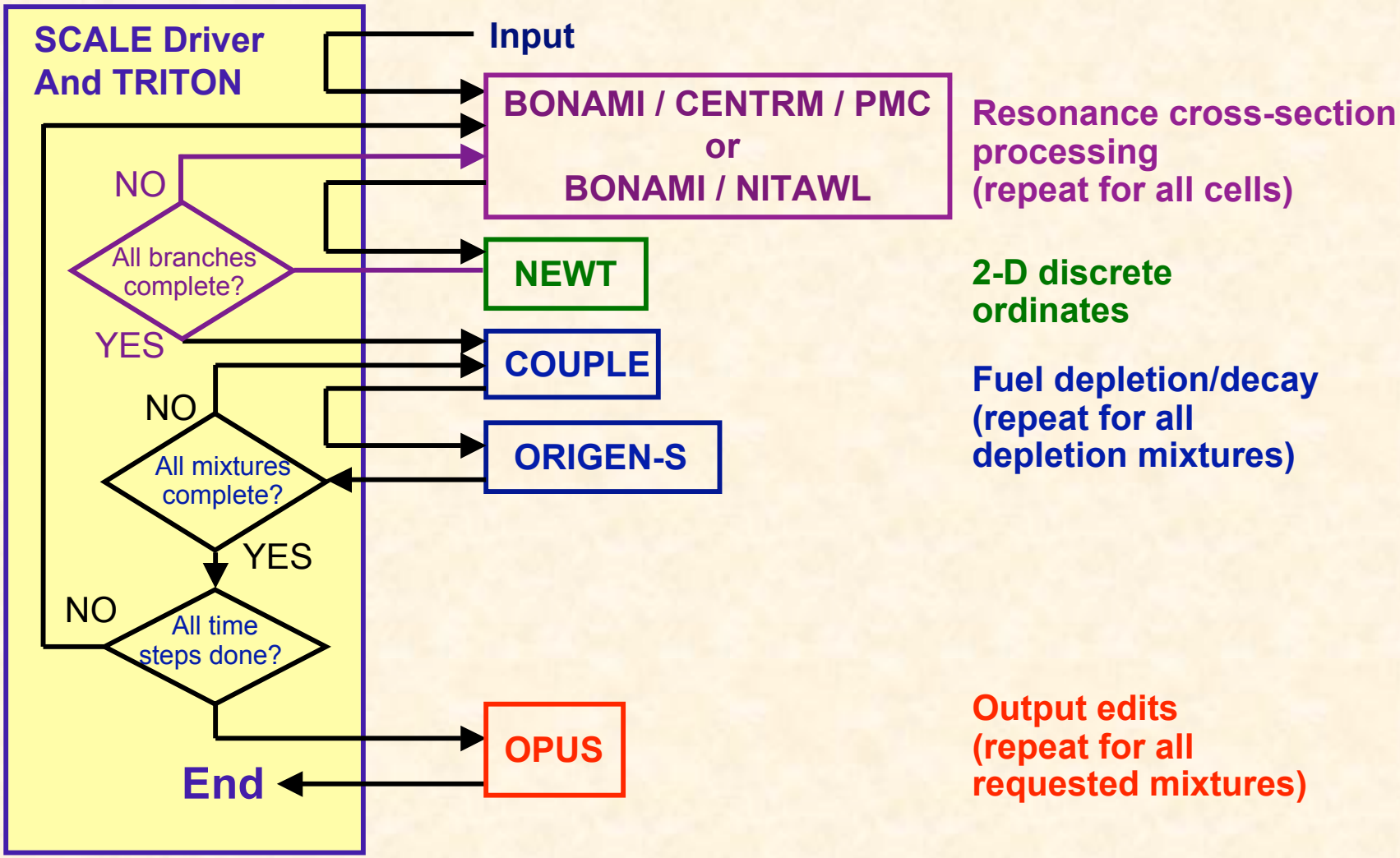
- TRITON (Transport Rigor Implemented with Time-dependent Operation for Neutronic depletion) is a multipurpose transport sequence
  - SCALE 5 control module
  - Drives five sequences
    - t-xsec - cross-section processing
    - t-newt - NEWT transport calculation
    - t-depl - NEWT/ORIGEN-S depletion calculation
    - t5-depl - KENO V.a/ORIGEN-S depletion calculation
    - t6-depl - KENO-VI/ORIGEN-S depletion calculation
  - Lattice Physics are currently supported only in the NEWT-based t-depl sequence.

TRITON Functionality

- TRITON processes input, prepares data, and handles calling sequence for each calculation type.
  - Material processing
  - Generation of problem-dependent cross sections
  - Transport solution (NEWT, KENO V.a, or KENO-VI)
  - Cross section update
  - Depletion and decay calculations
  - Update isotopic number densities
  - Iterate as required
- Remainder of discussion will focus primarily on t-depl


=Sequence Name
All Sequences
Material Information Processor Data
All Sequences
Depletion Data
tX-depl
NEWT or KENO Data
t-newt, tX-depl
End
All Sequences

t-depl Sequence With Branches



Capabilities of t-depl Depletion Sequence

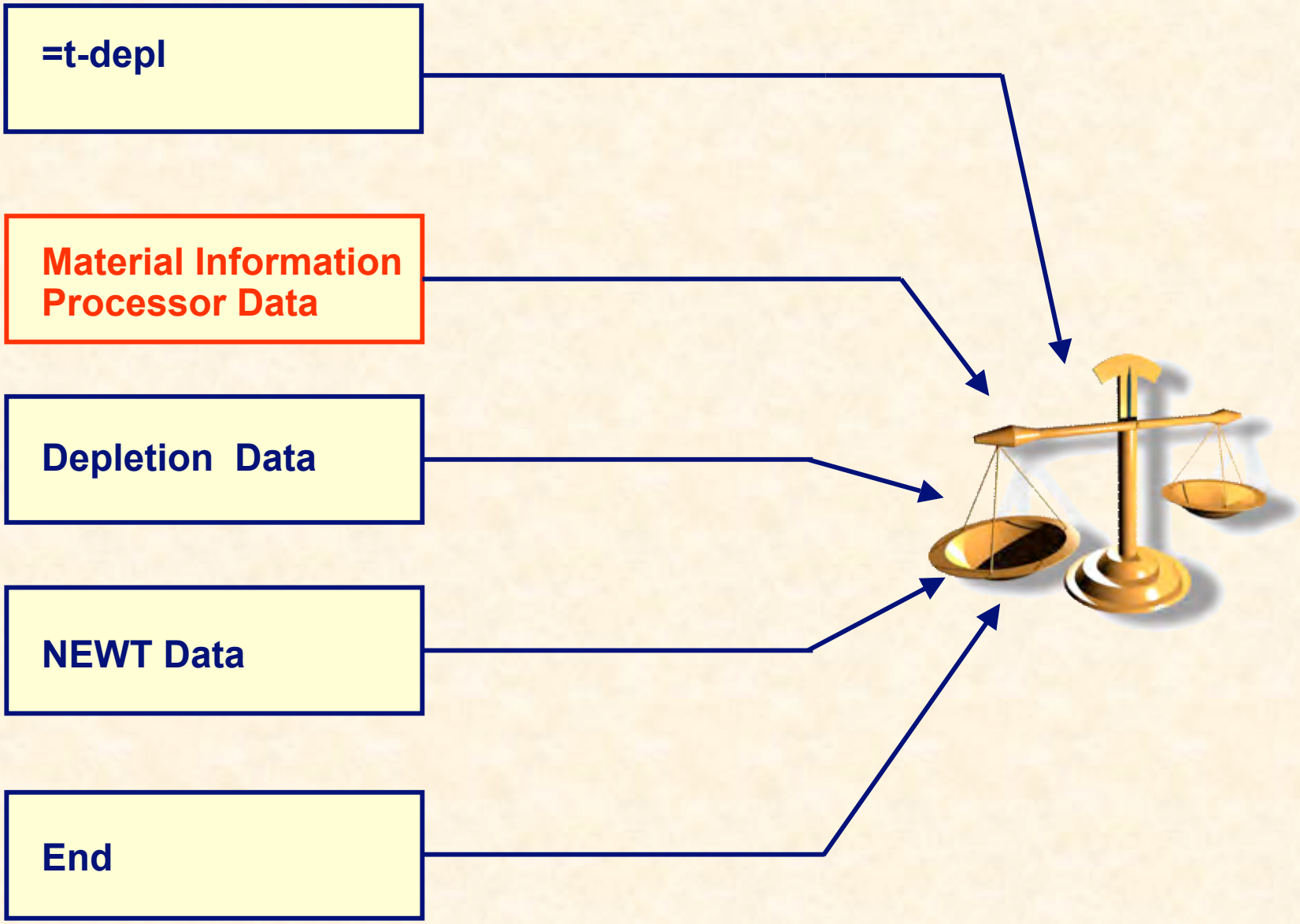
- Independent depletion of multiple materials.
- Isotopic prediction with rigorous transport model.
- Source terms.
- Decay heat.
- Depletion based on assembly power, local power, or region power.

Capabilities of t-depl Depletion Sequence (continued)

- Treatment of time-dependent changes in configuration (boron concentration, poison rod removal, etc.).
- Branch calculations.
- Database of burnup/branch cross-sections.
- Limited restart capabilities.

Multiregion depletion

- TRITON has the ability to deplete each fuel location in a system using local fluxes.
- A unique mixture must be specified for each region to be depleted.
- ORIGEN-S depletion is performed based on the average flux within each mixture, irrespective of location(s).
- One fuel element can be modeled as multiple mixtures (rings or regions within a contiguous volume).
- Conversely, large discontinuous regions (i.e., sets of pins) can be modeled as a single mixture.


=t-depl
Material Information Processor Data
Depletion Data
NEWT Data
End

Material Information Processor

- MIPLIB is a shared library of routines used by all SCALE control modules (e.g., TRITON, CSAS, CSAS6, TSUNAMI, SMORES, SAS1-SAS4, QADS)
- Reads the General Information, Standard Composition and Unit Cell input data in the TRITON input file
- Creates input data for the functional modules that perform resonance processing of the cross section data (BONAMI/CENTRM/PMC)
- These modules produce a problem-dependent multigroup cross-section library
- This library is used by NEWT and KENO

MIPLIB Input Layout (all SCALE modules)

=Sequence PARM=(CHK,CENTRM)
Title
Cross-section library
READ COMP
  Standard Composition Data
END COMP
READ CELLDATA
  Unit Cell Data
END CELLDATA
Sequence-specific data

General Information

=Sequence PARM=(CHK,CENTRM)
Title
Cross-section library
READ COMP
Standard Composition Data
END COMP
READ CELLDATA
Unit Cell Data
END CELLDATA
Sequence-specific data

General Information

Cross-Section Libraries

- **V6-238GROUP**

  238-group ENDF/B-VI (recommended in SCALE 5.1)
- **238GROUPNDF5**

  238-group ENDF/B-V (recommended in SCALE 5)
- **44GROUPNDF5**

  44-group ENDF/B-V collapsed with LWR fuel spectrum (recommended for LWR systems)
- **HANSEN-ROACH**

  16-group (not recommended)
- **Point data is available for both ENDF/B-V and ENDF/B-VI evaluations**
- **Create your own! (new TRITON capability)**

Material Information

- Standard Composition Data
  - Describe the materials used in the problem
- Unit Cell Data
  - Describe unit cell geometry (e.g., fuel rod cell) that impacts the flux spectrum and material cross sections

Basic Standard Composition Examples

SC MX [DEN=ROTH] VF [ADEN] TEMP [IZA WTP …] END

```
UO2    1  DEN=10.1  1.0  1027.0  92235  10.0
                92238  90.0  END

UF6    2  1.0  297.0  92235  5.0  92238  95.0  END

U-235  3  0  4.47-02  END

ZIRC2  4  1.0  550.0  END

H2O    5  END
```

User-specified Compound Examples

SC MX ROTH NEL (NCZA ATPM …) VF TEMP
[IZA WTP …] END

atomal2o3 1 2.55 2 13027 2 8016 3 1 293 end

atomC2H6SiO 2 0.97 4 6012 2 1001 6 14000 1
8016 1 1 293 END

User-specified Mixture/Alloy Examples

SC MX ROTH NEL (NCZA ATPM …) VF TEMP

[IZA WTP …] END

Mixture 3, Zirconium with 2.5% Niobium (6.5 g/cc):

wtptzirc-nb 3 6.5 2 40000 97.5 41093 2.5 1 293 END

wtptboral 4 2.65 2 5000 2.5 13027 97.5 1 293

5010 90 5011 10 END

Composition Data

```
=t-newt          parm=check
Prototypic reactor model
238groupndf5
READ COMP
uo2     1  1.0  293.0  92235  5.0  92238  95.0  end
wtptzirc-nb  2  6.5  2 40000 97.5 41093 2.5 1 293 end
h2o     3  end
ss304  4  end
END COMP
:
END
```

Unit Cell Data

- Used for:
  - Dancoff factors,
  - Resonance self-shielding of cross sections
  - Homogenization of cell mixtures (a.k.a.,"cell-weighting")
  - 1-D eigenvalue calculations
- Input includes:
  - Mixture numbers
  - Dimensions
  - Unit cell boundary conditions

TRITON Input CELLDATA Example

```
=t-newt     parm=check
Prototypic reactor model
238groupndf5
READ COMP
uo2     1  1.0  293.0  92235  5.0  92238  95.0  end
wtptzirc-nb  2  6.5  2 40000 97.5 41093 2.5 1 293 end
h2o      3  end
ss304  4  end
n        5   den=1e-4 end
END COMP
READ CELLDATA
 latticecell squarepitch fuelr=0.47 1 gapr=0.4875 5
            cladr=0.545 2 hpitch=0.72 3 end
END CELLDATA
.
.
END
```

CENTRM/PMC Methodology

- **CENTRM: Continuous ENergy TRansport Module**
  - **Computes space-dependent, continuous-energy neutron spectra for 1-D model of critical system**

- **PMC: Process Multigroup Cross sections**
  - **Computes problem-dependent cross sections using flux spectrum from CENTRM**

Features of CENTRM

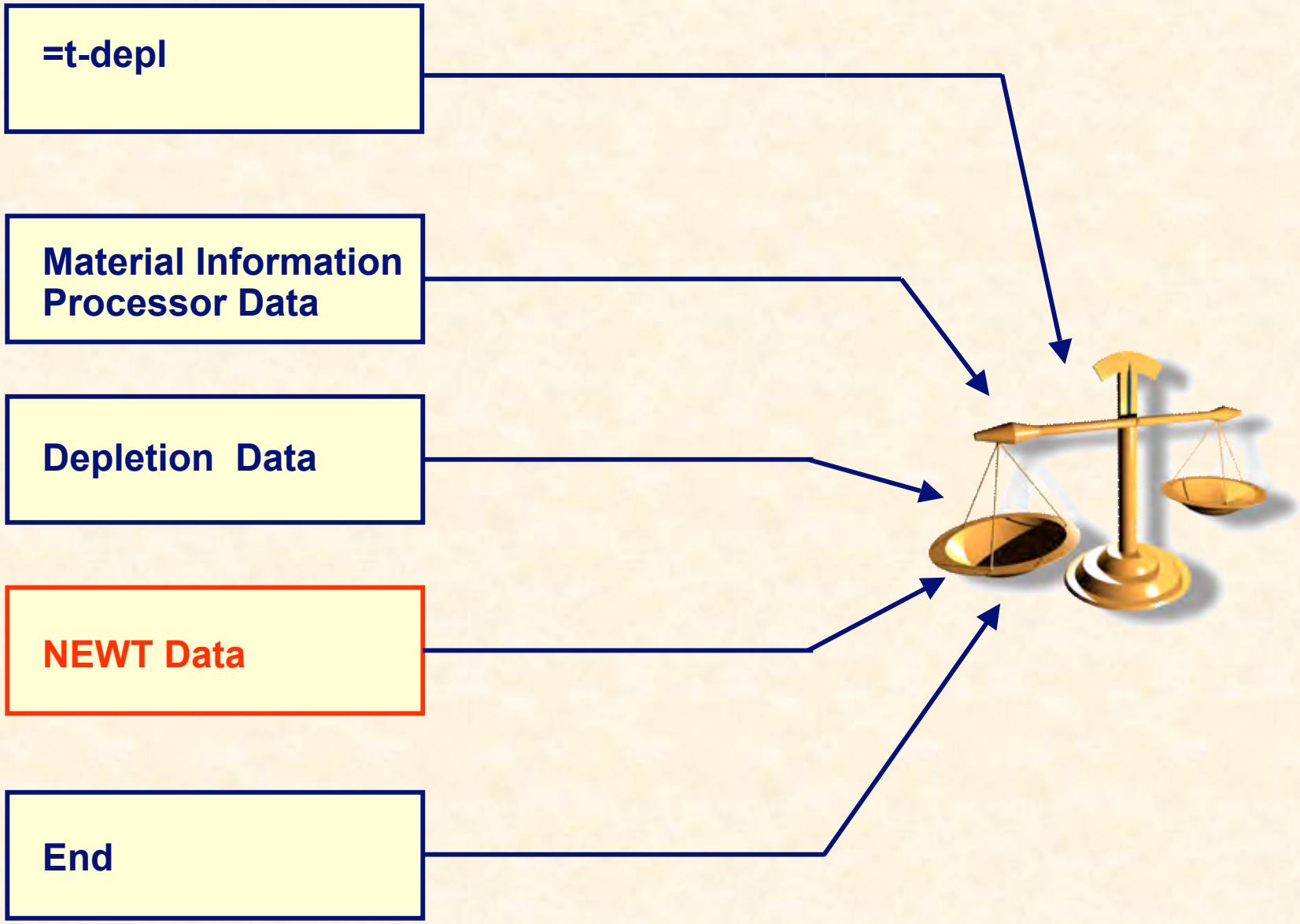
- Solves 1-D discrete ordinates transport equation using pointwise XS data to obtain continuous-energy angular flux spectra
- Computes full energy-range spectrum (not just in resolved range)
  - pointwise in resonance energy range
  - multigroup in fast and thermal ranges
  - pointwise is an option in thermal ranges
- Pointwise data pre-processed directly from ENDF/B specs (no assumption of SLBW)

Features of CENTRM (Cont.)

- Generates problem-specific energy mesh that accounts for resonance overlap
- Includes general space and temperature variation in materials
- Point cross sections are automatically temperature-corrected.
- Computes anisotropic scatter source for s-wave elastic & inelastic reactions using pointwise flux and analytical scatter kernel
- Includes pointwise thermal spectrum calculation

Benefits of CENTRM

- CENTRM/PMC expands traditional resolved resonance self-shielding capabilities in NITAWL
- Removes several approximations in NITAWL
  - Accurate calculation of slowing-down sources with resonance interference effects
  - Includes resonance overlap effects
  - Accurate treatment of clad, moderator flux spectrum
  - Includes impact of scatter-anisotropy and leakage on PW spectrum
- Treats cases with multiple absorber-zones
  - e.g., fissionable lump in fissionable solution


=t-depl
Material Information Processor Data
Depletion Data
NEWT Data
End

Transport Solution

- NEWT performs the transport physics analysis for a specified 2-D configuration
  - similar in function to XSDRNPM (1-D), and KENO (3-D)
  - NEWT is a two-dimensional arbitrary-geometry discrete ordinates transport solver (more later)
- KENO V.a and KENO-VI are optional transport solvers for depletion.

Transport Solution (continued)

- Cross sections are created for each pin type
- With CENTRM, cross sections may be generated for subregions (e.g., rings) within a cell.
- The transport solution couples all cross sections in generating a detailed multidimensional solution
  - NEWT is fully converged everywhere
  - KENO has stochastic and spatially-varying uncertainties.
- The average spatial flux is used to independently deplete each spatial region

Transport Solution

- The transport solution performs two functions:
  - Spatial flux solution used by TRITON to calculate power distributions:
    - ORIGEN-s may deplete using powers or fluxes
    - Powers must be scaled according to user input
  - Multigroup cross section collapse to update ORIGEN three-group cross sections.

Transport Flux Weighting

- Within the TRITON sequence, NEWT automatically performs a 3-group collapse to thermal/resonance/fast energies expected by COUPLE
  - COUPLE is a utility to convert AMPEX-format cross section to update ORIGEN data
  - Cross sections are collapsed based on the average flux in each mixture
  - Collapsed cross sections are created for each depletion mixture
  - A loop over COUPLE/ORIGEN-S is performed, processing each mixture one at a time, for all mixtures being depleted

Transport Flux Weighting (cont'd)

- The NEWT transport solution uses the flux distribution to calculate the power generation in each mixture.
  - fluxes are used to calculate fission and n-γ reaction rates, and obtain power generation rates in all mixtures
  - TRITON obtains powers and fluxes from NEWT
    - Powers are used to normalize fluxes for ORIGEN-S
    - NEWT uses the same flux-to-power conversions as ORIGEN-S for consistency
    - Depletion by flux is was default in SCALE 5, but depletion by power was added and made default in SCALE 5.1.

Transport Flux Weighting (cont'd)

- User-requested flux collapse is also available - if requested, two independent group collapses are performed
  - required 3-group collapse
  - user-specified collapse
- This feature is most commonly used in generating reactor physics parameters in a few-group structure.
- Weighted microscopic cross sections are saved in AMPX Working Library Format
- Homogenized macroscopic cross sections, if requested, are saved in same format

Library Updates

- COUPLE reads 3-group working format libraries and updates the ORIGEN-S library database.
- Only those nuclides used in the transport calculation are updated on the ORIGEN-S library
  - By default, in addition to input trace quantities of nuclides a set of 66 additional nuclides are automatically added to all fuel mixtures.
  - Optionally, another 150 nuclides may be added.
    - Additional nuclides are necessary to accurately predict eigenvalues at higher burnups
    - Incurs increased computational effort

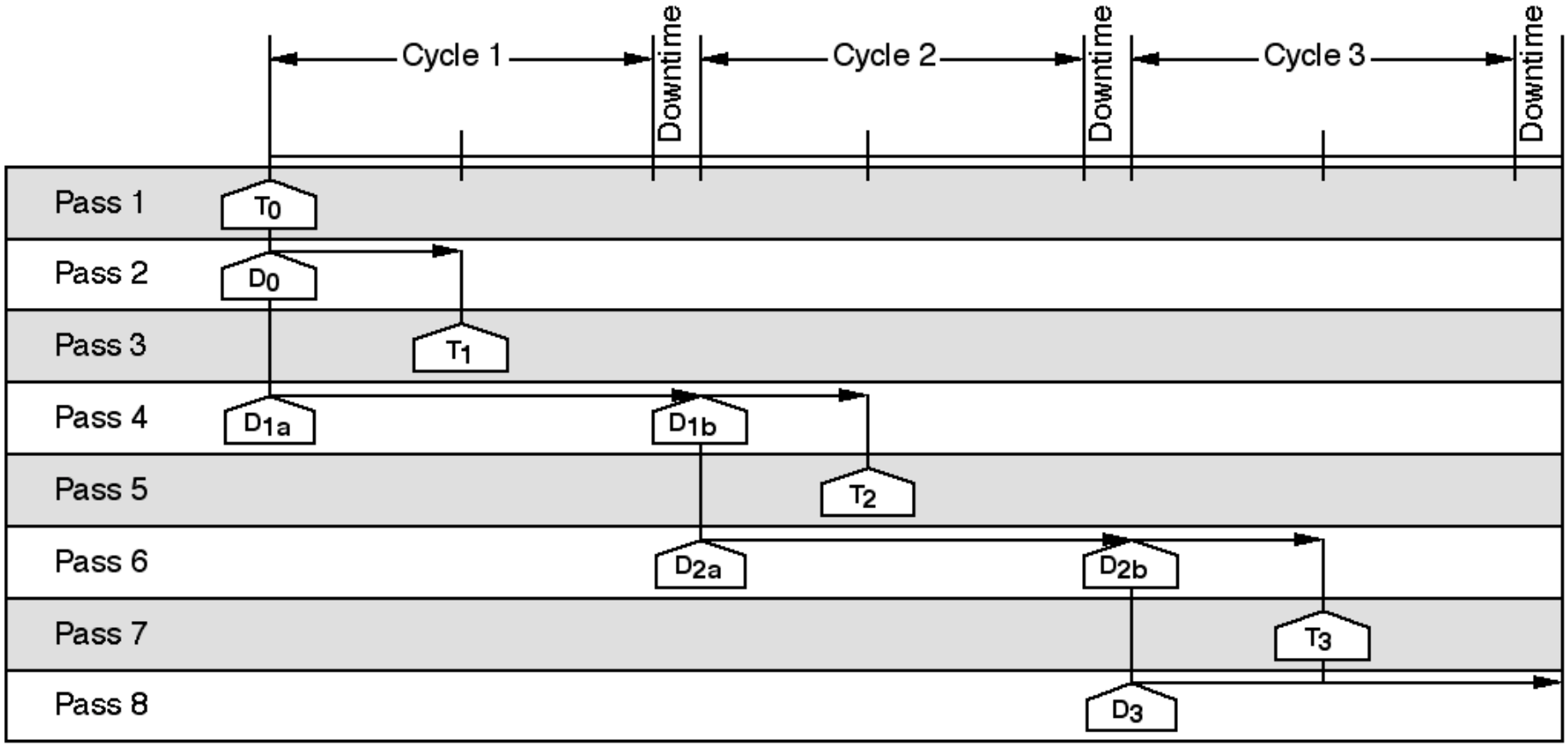
Library Updates (cont'd)

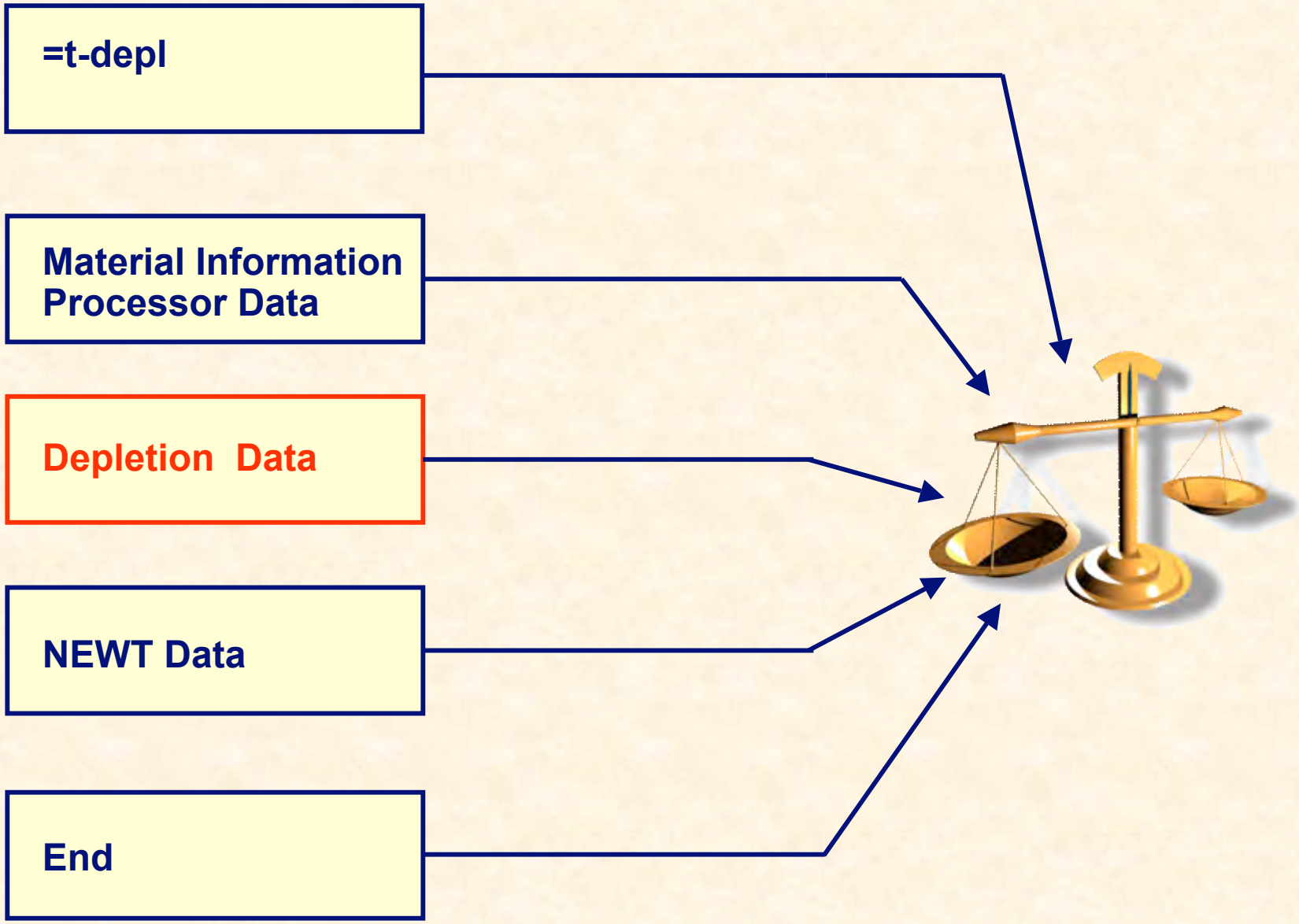
- TRITON creates burnup-dependent cross-section libraries that are used by ORIGEN-S for depletion calculations
  - Multigroup cross sections change with time as inventories vary.
  - Changes are generally second-order so cross-section updates are not needed on a fine temporal basis.
  - Cross-section updates are required more often when properties change rapidly, e.g., Gd depletion.

Library Updates (cont'd)

- TRITON creates burnup-dependent cross-section libraries that are used by ORIGEN-S for depletion calculations
  - Library updates are NOT timesteps - TRITON calculates ORIGEN-S timesteps based on burnup length.
  - Libraries are only used for depletion calculations - decay does not depend on cross sections.
- TRITON uses a predictor-corrector approach for depletion calculations.

t-depl Predictor/Corrector Approach




=t-depl
Material Information Processor Data
Depletion Data
NEWT Data
End

Input Format for the t-depl Sequence

```
=t-depl                parm=AAAA (optional)
Title or description of up to 80 characters
x-sect_lib_name
read alias             (optional)
[List of aliases]  ← MUST PRECEDE read comp BLOCK
end alias
read comp
[List of mixture specifications]
end comp
read celldata
[unit cell specifications]
end celldata
```

Input Format for the t-depl Sequence

```
end celldata
read keyword1
[keyword 1 type data]
end keyword1
read keyword2
[keyword 1 type data]
end keyword2
[additional keyword blocks as needed]
read model
[NEWT/KENO model data]
end model
end
```

Input Format for the t-depl Sequence

- **t-depl supports the following keyword blocks**
  - ***alias*** **- variable substitution definitions** *(optional)*
  - ***burn*** **- power and decay history** *(required)*
  - ***depletion*** **- mixtures to be depleted, normalization instructions** *(required)*
  - ***timetable*** **- time-dependent concentration/temperature changes** *(optional)*
  - ***keep_data*** **- user specification of which code outputs should be retained** *(optional)*
  - ***branch*** **- branch calculation instructions** *(optional)*
  - ***opus*** **- post processing commands** *(optional)*
  - ***model*** **- NEWT input** *(required)*

BURN/BURNDATA data block *(required)*

```
read burndata
power=P1 burn=B1 down=D1 nlib=N1 end
power=P2 burn=B2 down=D2 nlib=N2 end
power=P3 burn=B3 down=D3 nlib=N3 end
...
end burndata
```

- **P = average specific power in the basis mixture(s), in megawatts per metric tonne of heavy metal (MW/MTHM),**
- **B = burn length at power P, in days,**
- **D = down or decay time (zero power) following the burn period (days)** ***(optional, default=0.0)*****,**
- **N = number of cross-section libraries produced per cycle** ***(optional, default=1)*****.**

DEPLETION data block *(required)*

- General format:

`read depletion` M_1 M_2 M_3… M_n `end depletion`

- Only mixture numbers (assigned in standard composition input) listed here will be depleted.
- Minimum of one mixture, limit of 255 mixtures.
- Generally only fuels, absorbers or targets are depleted - clad, moderator, and other structure generally need not be depleted (clad must be depleted for activation analysis).
- TRITON normalizes system to a total mass of 1 metric ton of *heavy metal* ($Z \geq 90$) over all depletion mixtures.

DEPLETION data block

- *depletion* block also allows specification of *basis* to which input power is applied.
- By default, input power is applied to entire system (ALL mixtures, depleted or not)
  - power in individual mixtures is distributed according to flux- and energy-producing reactions
  - non-fuel regions contribute to system power (n-γ reactions)
  - appropriate if the average power in a fuel assembly is known.

DEPLETION data block

- Power can be normalized to one mixture or set of mixtures by setting a negative sign (-) in front of the mixture number:

```
read depletion -M1 M2 M3… Mn end depletion
```

  - this would normalize powers such that mixture M_1 will have a specific power of P MW/MTHM (P is specified in burn block).
  - Powers in other mixtures will vary from that of M_1 according to relative power generation in each mixture.

TIMETABLE data block

```
read timetable
temperature 3
121.0 615
121.01 685
322.5 685
352.5 610
738.75 610 end
density 3 2 5010 5011
0.0 1.0
121.0 0.1
121.01 1.25
322.5 0.13
352.5 0.98
738.75 0.1 end
end timetable
```

- Timetable curves are used to determine temperatures & densities at the time of each transport calculation
- Linear interpolation is used between points
- Standard comp specifications are used unless altered be a timetable entry
- Select nuclide densities can be varied for boron letdown, mixture exchange.

TIMETABLE data block

- **Example**

Cycle	Boron concentration (ppm)		Temperature (K)
	BOC	EOC	
1	**1000**	**100**	**615**
2	**1250**	**130**	**685**
3	**980**	**100**	**610**

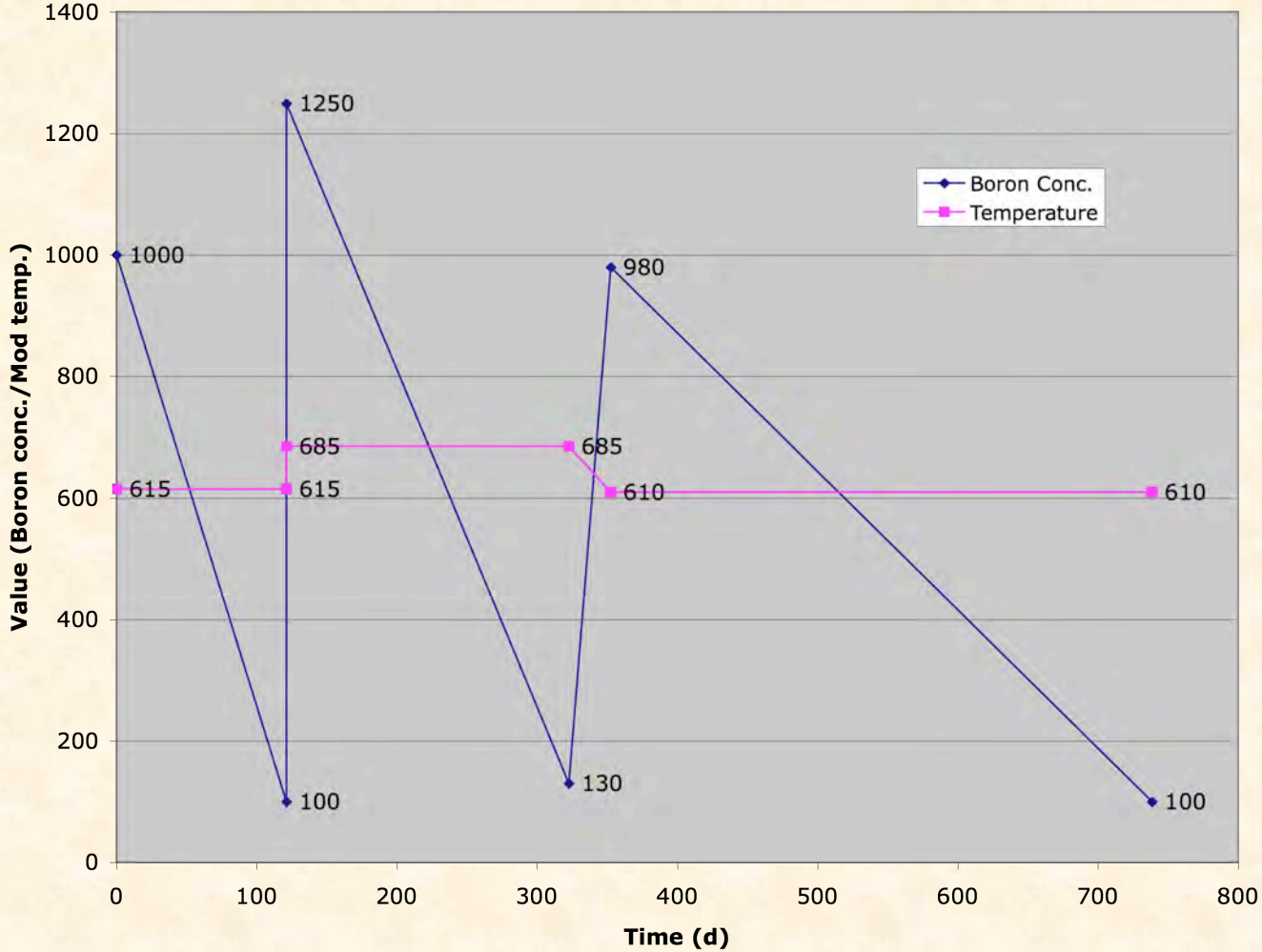
```
read timetable
temperature 3
0.0  615     121.0 615
121.01 685   322.5 685
352.5 610    738.75 610 end
density 3 2 5010 5011
0.0 1.0          121.0 0.1
121.01 1.25      322.5 0.13
352.5 0.98       738.75 0.1 end
end timetable
```

Cycle 1: burn from 0 to 121d, no down time

Cycle 2: burn from 121 to 322.5d, down from 322.5 to 352.5d.

Cycle 3: burn from 352.5d to 738.75d, down time unimportant.

TIMETABLE data block



ALIAS data block

- **Mixture lists can be specified in multiple forms**
  - **separated by spaces and/or commas**
  - **a series can be specified as** `first-last`
  - **any combination of the above**
- **Example:**

```
read alias
$fuel 1–3 10 end        [Mixtures 1, 2, 3, and 10]
$clad 4,5,6,11 end      [Mixtures 4, 5, 6, and 11]
$mod 7 8–9 12 end       [Mixtures 7, 8, 9, and 12]
end alias
```

- **Aliases are then used in place of mixture numbers.**
- **A pre-processor parses input, expands all aliases, and rewrites a modified input.**

BRANCH data block *(optional)*

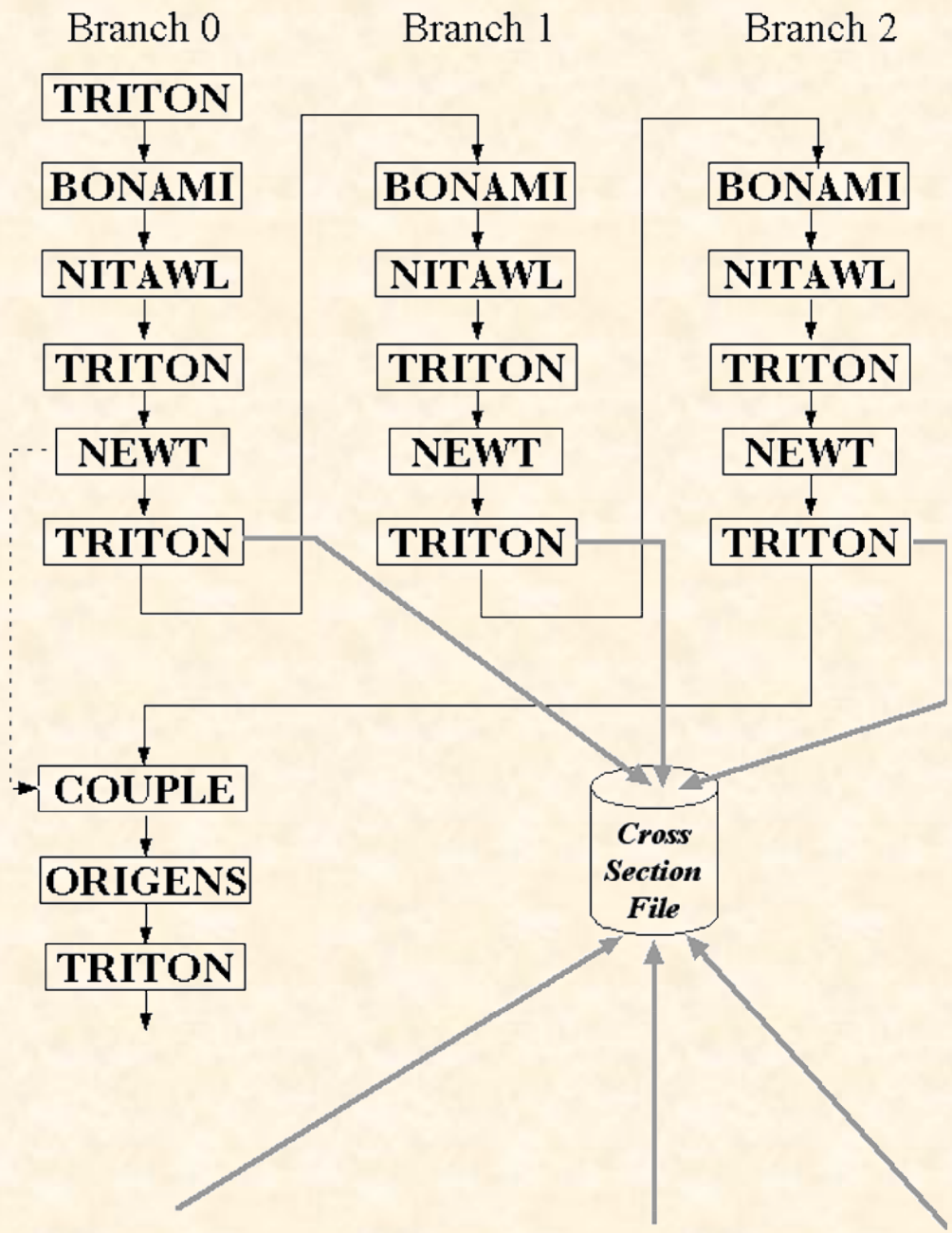
- Branch calculations are supported in TRITON for lattice physics analysis (t-depl only)
- Branch calculations allow for a nominal condition transport calculation followed by perturbation of one or more of the following parameters:
  - fuel temperature
  - moderator temperature
  - moderator density
  - soluble boron concentration
  - control rod insertion

BRANCH data block

- Branch capabilities were added to TRITON to support lattice physics analyses in an ongoing project for the US Nuclear Regulatory Commission.
- Branch calculations yield cross sections computed for one or more perturbed states for each depletion step.
- Cross sections are saved in a single database file for a user-specified collapsing scheme.
- Cross sections may be subsequently extracted from this file and converted for use by core simulation codes.

BRANCH data block

- **Cross sections are computed for the nominal and each perturbed state.**



BRANCH data block

- The *branch* data block consists of two parts:
  - The first part defines mixture associations to be used in branch calculations
    - associates mixture numbers with keywords "fuel" or "mod"
    - Also defines mixtures present in the transport model during control rod insertion and withdrawal.
  - The second part defines the actual branch states.
- TRITON performs repeated NEWT calculations for nominal conditions followed by each perturbed state

BRANCH data block

- For N branches, N+1 transport calculations are performed. Nominal or base conditions are considered "branch 0"
- Depletion calculations are all based on branch 0 conditions
- Other branches yield perturbed cross sections, but do not affect the depletion calculation.
- Cross sections are collapsed and homogenized according to input specified within the NEWT model (collapse and homog data blocks)

BRANCH data block

```
read branch
define deftype   I1 I2 … In end
…
tf=fuelT tm=modT dm=modD sb=boronconc cr=inout end
…
end branch
```

deftype	=	'fuel,' 'mod,' 'crout', or 'crin'
I_i	=	list of mixtures associated with type definition *deftype*.
fuelT	=	branch fuel temperature (K)
modT	=	branch moderator temperature (K)
modD	=	branch moderator density (g/cm^3)
boronconc	=	soluble boron concentrations (ppm)
inout	=	control rod state (out=0, in=1)

BRANCH Data Block Example

- **6 branch specifications - 6 calculations:**
  1 base, 5 branches

```
read branch
define fuel 11 12 end
define mod 13 14  end
define crout 20 21 end
define crin  30 31 end
dm=.75 cr=0 sb=655 tf=900 tm=650 end
dm=.80 cr=0 sb=655 end
dm=.80 cr=1 sb=655 end
sb=20 dm=.8 end
sb=1300 dm=.8 end
tf=559 end
end branch
```

fuel branches are applied to 11 & 12

mod branches are applied to 13 & 14

mixtures 20 & 21 are present for C/R out

mixtures 30 & 31 are present for C/R in

0) initial (base) conditions

1) mod den changed to 0.8

2) C/R in, mod den changed to 0.8

3) soluble B reduced to 20ppm, mod den 0.8

4) soluble B increased to 1300ppm, mod den 0.8

5) fuel temp changed to 559, mod den to base value.

READ MODEL Block

- The transport model input (NEWT, KENO V.a, or KENO-VI) is read with TRITON input.
- Each module's input is embedded between *READ MODEL* and *END MODEL* blocks.

```
read model
Two-pin depletion
read parm
   prtflux=no  drawit=yes echo=yes
   xnlib=4 run=yes collapse=yes prtmxsec=no prtbroad=yes
   sn=4 inners=2 outers=800 epsilon=1e-4 prtmxtab=yes
  solnmode=buck restart=yes
end parm
read materials
 1    1   !  3.3% enriched uo2 ! end
 2    1   !  zirc cladding     ! end
 3    1   !  water             ! end
end materials
end model
```

How am I doing so far?

While testing out a new product,
Mark gets some unexpected feedback.

UT-BATTELLE

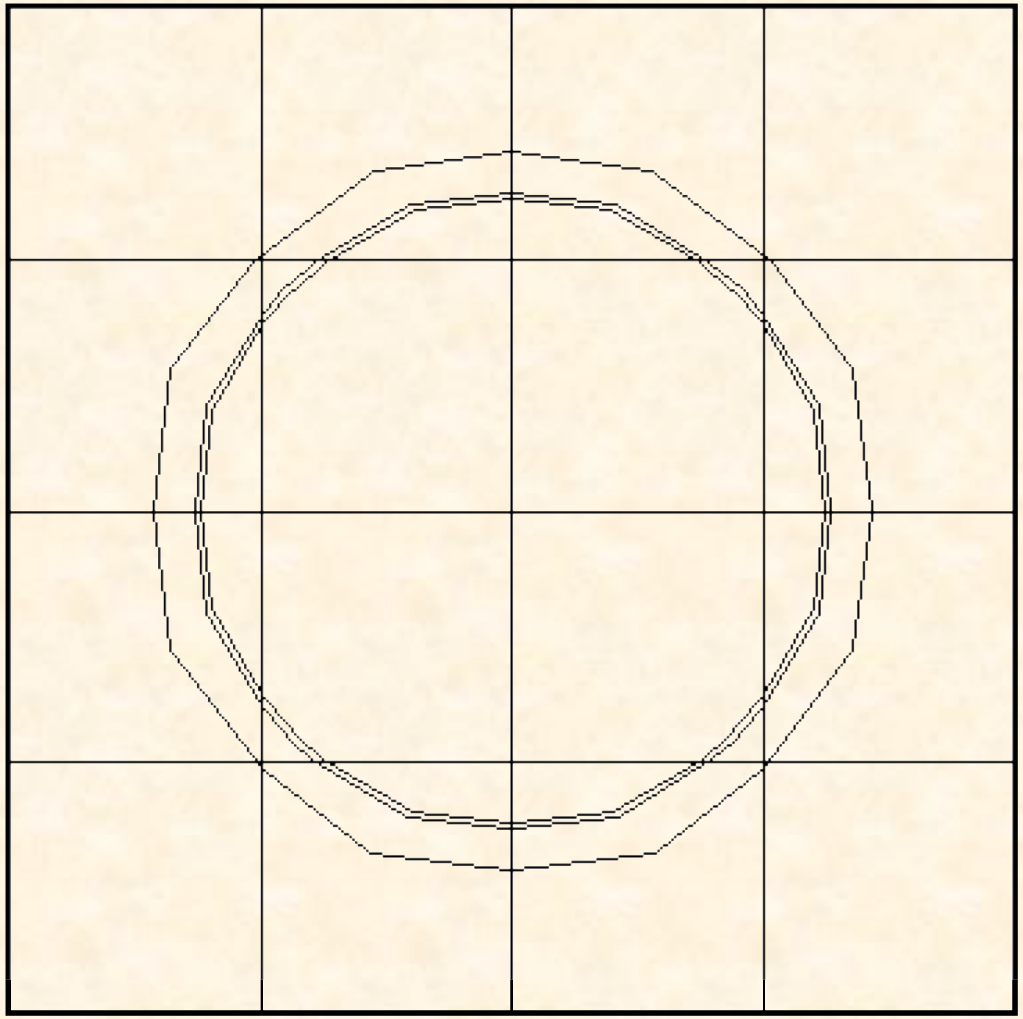
NEWT: Theory and Capabilities

OAK RIDGE NATIONAL LABORATORY
U. S. DEPARTMENT OF ENERGY

NEWT - Introduction

- NEWT provides a discrete ordinates solution for arbitrary, non-orthogonal configurations
  - Discrete ordinates methods are the most rigorous approximation to the transport equation
  - Direct solution of angular fluxes allows capture of important anisotropic effects
- NEWT uses Extended Step Characteristic discretization:
  - Allows cells in the form of arbitrary polygons
  - Able to closely approximate any curved surface (with volume conserved)
  - Able to exactly represent any straight-sided body

NEWT - Introduction



Theory - Extended Step Characteristic Discretization

Characteristic form of the transport equation

$$\frac{d\psi(s)}{ds} + \sigma_t(s)\psi(\mathrm{s}) = Q(s)$$

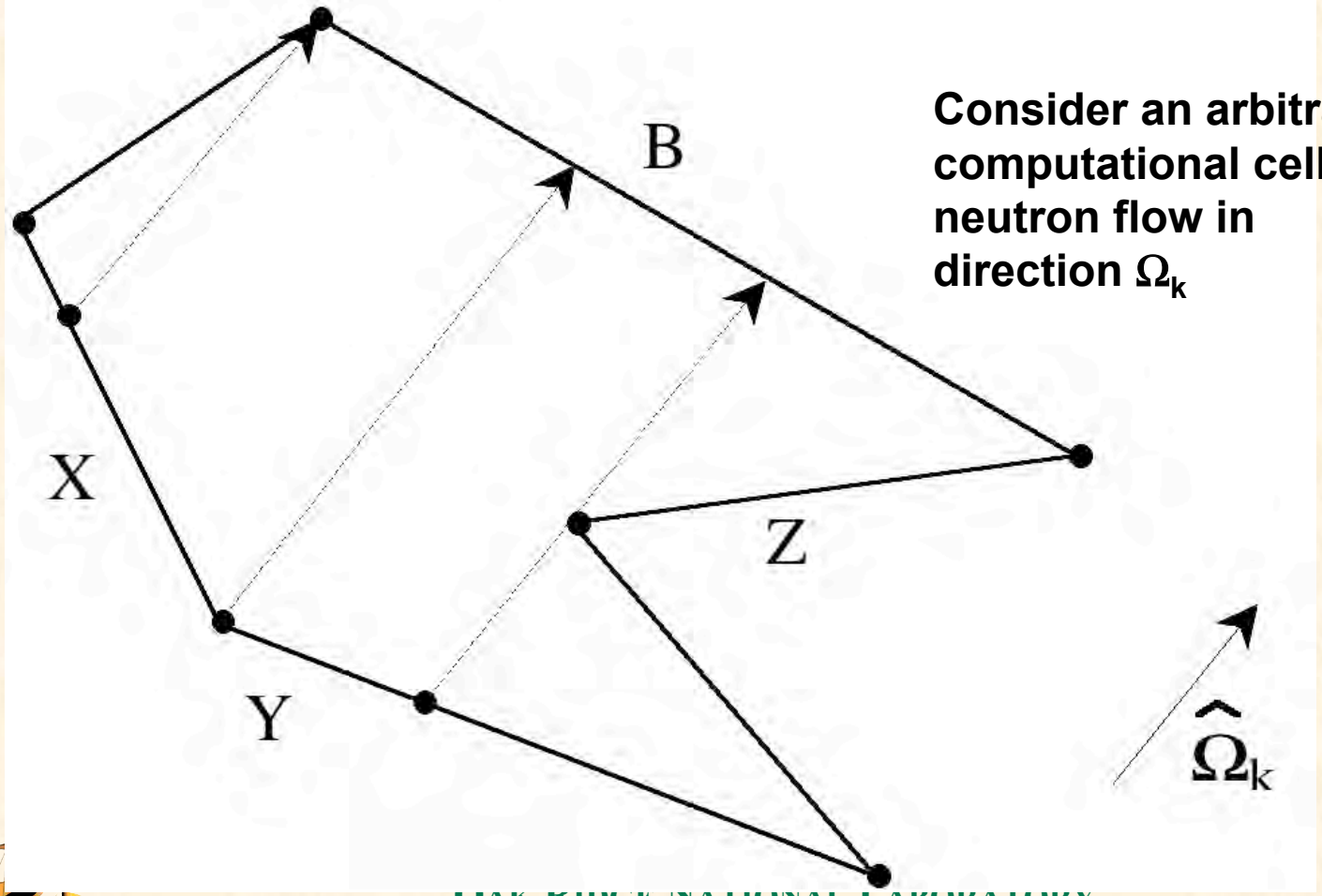
Which has the solution:

$$\psi(s) = \psi_0 e^{-\sigma_t s} + e^{-\sigma_t s}\int_0^s Q e^{\sigma_t s'} ds'$$

Assuming a constant source (Q) and properties (σ_t) the integral can be evaluated, giving:

$$\psi(s) = \psi_0 e^{-\sigma_t s} + \frac{Q}{\sigma_t}\left(1 - e^{-\sigma_t s}\right)$$

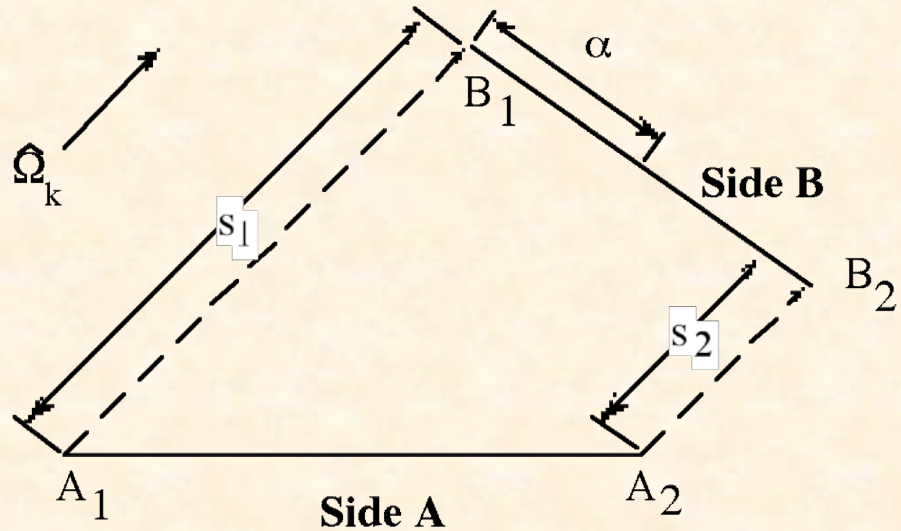
Theory - Extended Step Characteristic Discretization



Consider an arbitrary computational cell and neutron flow in direction Ω_k

Theory - Extended Step Characteristic Discretization

$$\overline{\psi}_B = \frac{\int_0^L \psi(s(\alpha))\,d\alpha}{\int_0^L d\alpha}$$

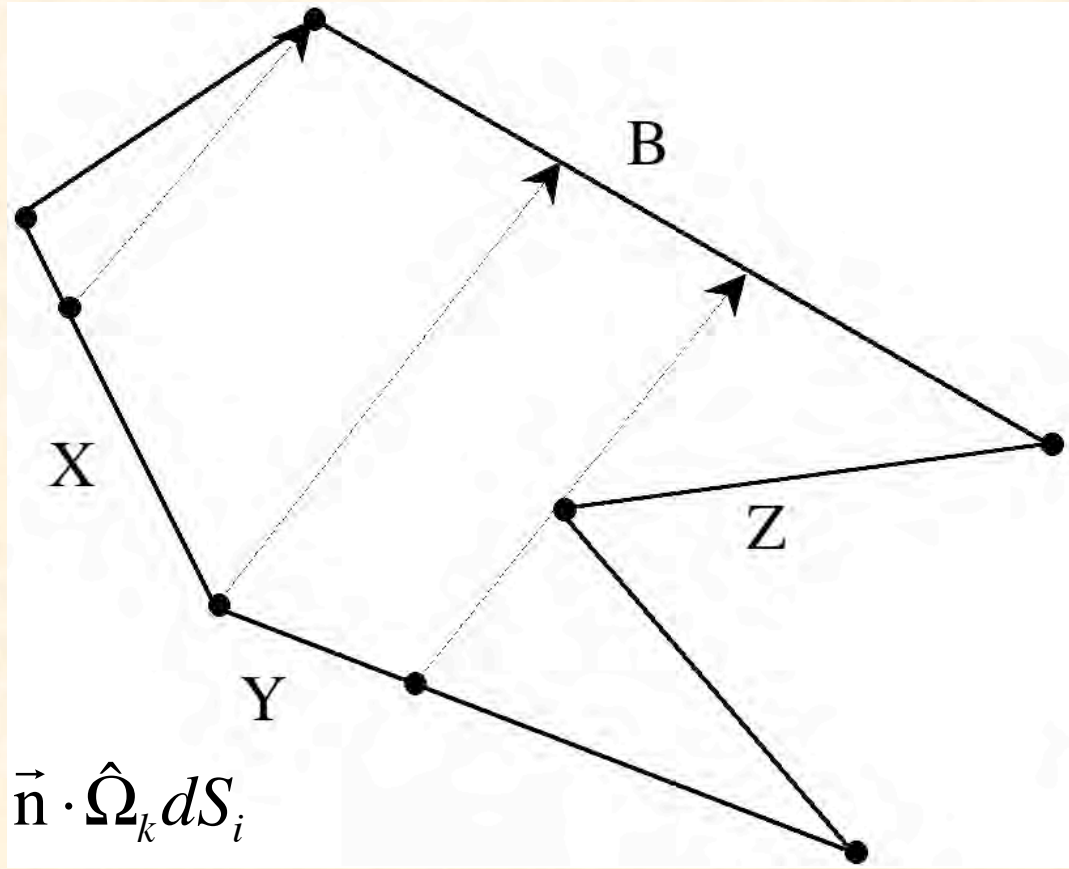


$$\overline{\psi}_B = \frac{(\overline{\psi}_A - Q/\sigma_t)}{\sigma_t(s_2 - s_1)}\left[e^{-\sigma_t s_1} - e^{-\sigma_t s_2}\right] + Q/\sigma_t \qquad s_1 \neq s_2$$

$$\overline{\psi}_B = (\overline{\psi}_A - Q/\sigma_t)e^{-\sigma_t s_1} + Q/\sigma_t \qquad s_1 = s_2$$

Theory - Extended Step Characteristic Discretization

$$\overline{\psi}_B = \frac{1}{L_B}\sum_{i=1}^{n}\overline{\psi}_i \ell_i$$



$$\overline{\psi}_{cell} = \frac{Q}{\sigma_t} - \frac{1}{\sigma_t V}\sum_{i=1}^{n}\overline{\psi}_i \int_{S_i} \vec{n}\cdot\hat{\Omega}_k dS_i$$

Theory - Solution method

- Grid generator first converts curved surfaces (cylinders) to line segments, then places bodies on an underlying grid.
- NEWT analyzes each line segment for each angle to:
  - Determine required solution sweep order
  - Calculate side-to-side transport factors
  - Save all non-zero factors in a database

Theory - Solution method

- Iterative solution uses cross-section, source, and transport factors to solve all line segment angular fluxes according to direction-dependent sweep order.
- Balance eqn. used for cell-averaged angular flux, numerical integration gives scalar flux.
- Cells are solved iteratively in an inner loop for each energy group, based on a guessed source Q.
- An outer loop iterates on energy groups to update the source as energy groups converge.

Features and Capabilities

- NEWT uses the discrete ordinates method, allowing detailed angular and spatial resolution of fluxes.
  - Arbitrary order of scattering (P_n approximation)
  - Arbitrary quadrature order
  - Automated grid generation (fast and easy)
  - Multigroup energy solution
  - A rigorous and highly accurate deterministic solution for non-trivial configurations

Features and Capabilities

- Although the characteristic form of the transport eqn. is integrated in the NEWT derivation, NEWT is <u>not</u> a Method of Charactistics code.
- Ray tracing is done only within a cell, to obtain angular fluxes on boundaries and the cell average.
- Rays are not uniformly distributed - only used where needed.

Features and Capabilities

- Current solution modes
  - Eigenvalue
  - Critical buckling
  - Eigenvalue with B1 Critical Spectrum
- Postscript-based graphics
- Cross-section collapse
- 4 Boundary conditions
  - Reflective
  - White
  - Vacuum
  - Periodic
- Convergence control
- Comparable speed to other 2D S_n codes

Features and Capabilities

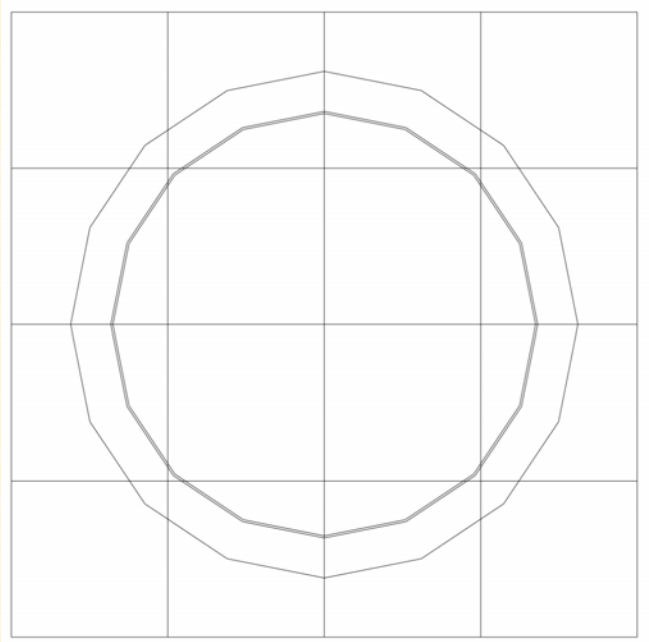
- Runs within SCALE
  - Standalone SCALE module or as part of the TRITON sequence
  - Uses AMPX-format cross-sections
  - Keyword-based input
- Geometry specification *very similar* to that of KENO-VI (2D instead of 3D)
- "Old" (SCALE 5.0) geometry package still supported
- Grid generation is completely automated!
- Spatial discretization is accomplished by merging bodies with an underlying rectangular grid.

Features and Capabilities

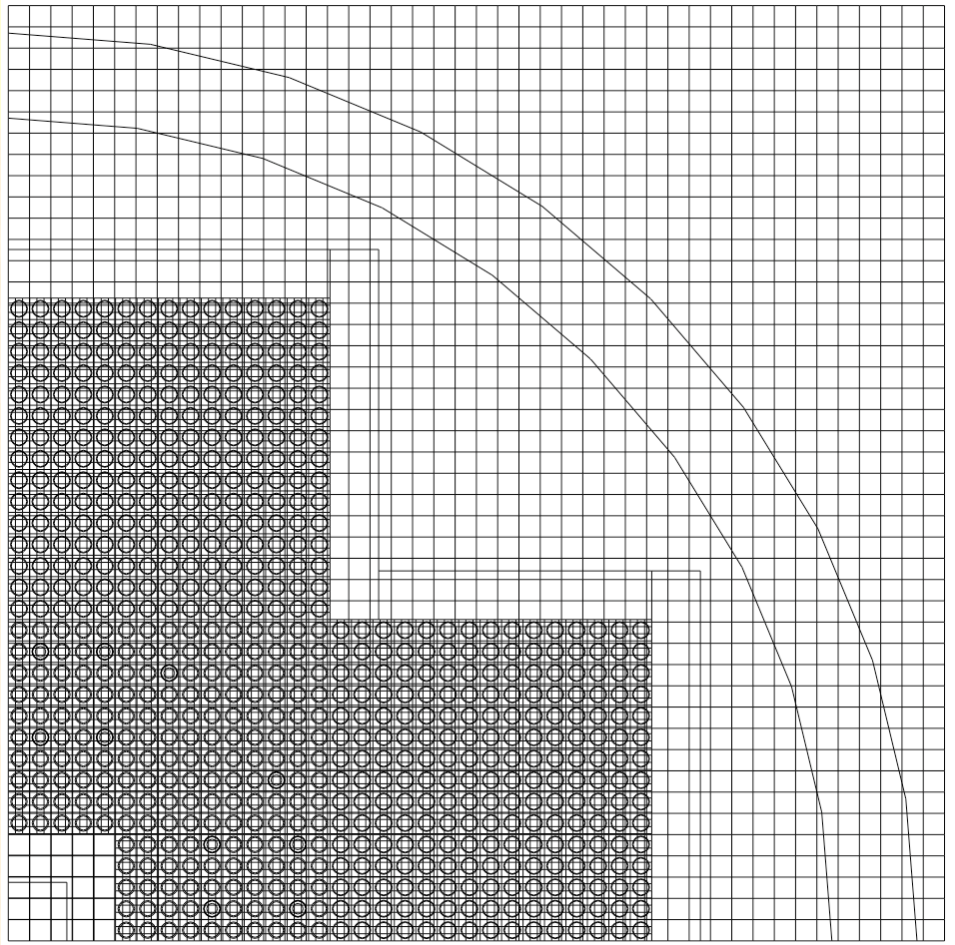
- Creates auxiliary files for other applications:
  - Restart
  - Geometry
  - Flux file for depletion (TRITON)
  - Flux weighted cross sections
    - By nuclide in each mixture
    - Homogenized for multiple mixtures
    - Outputs other data required for diffusion codes
  - Data used by SCALE sensitivity modules

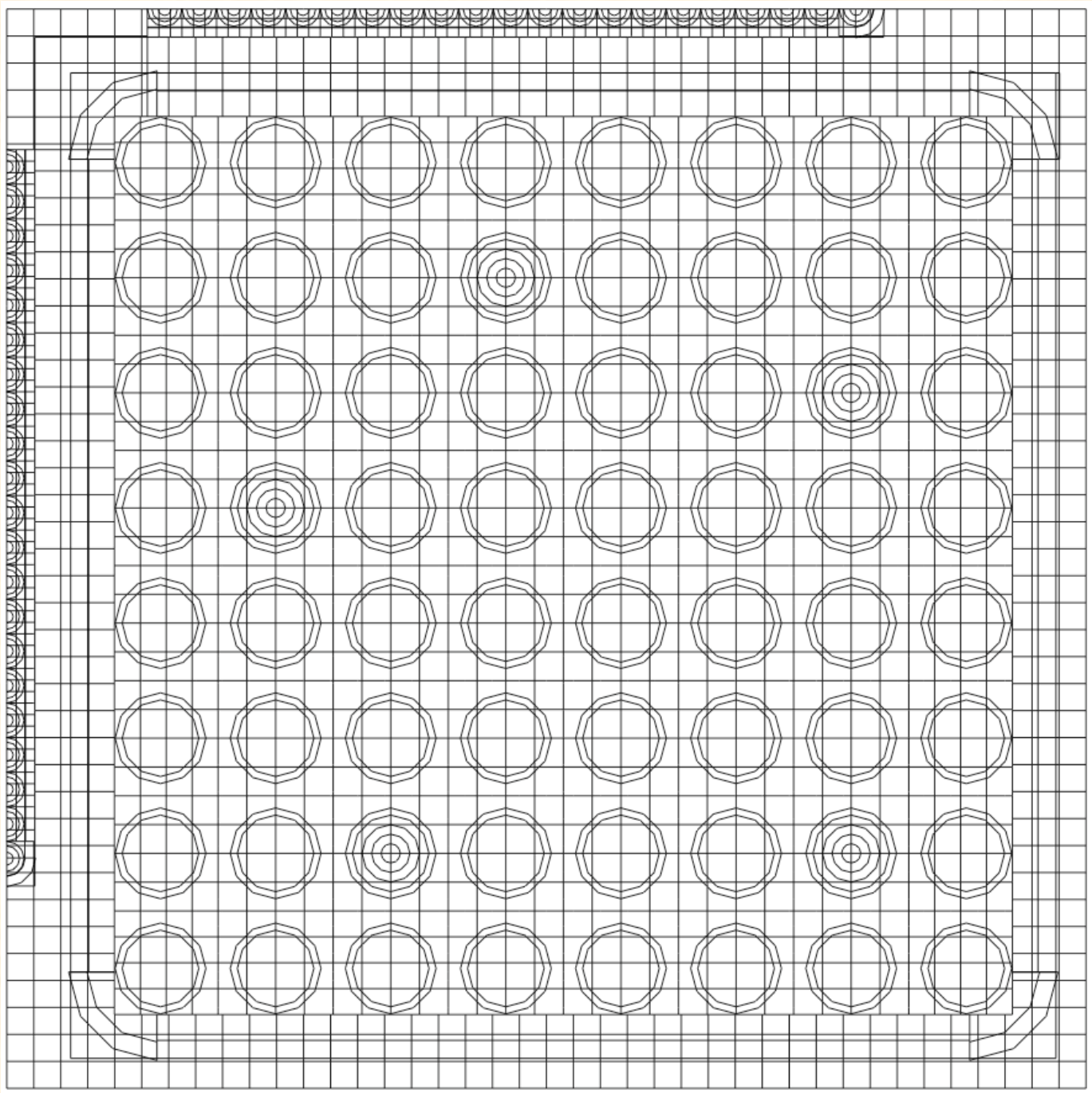
NEWT Models

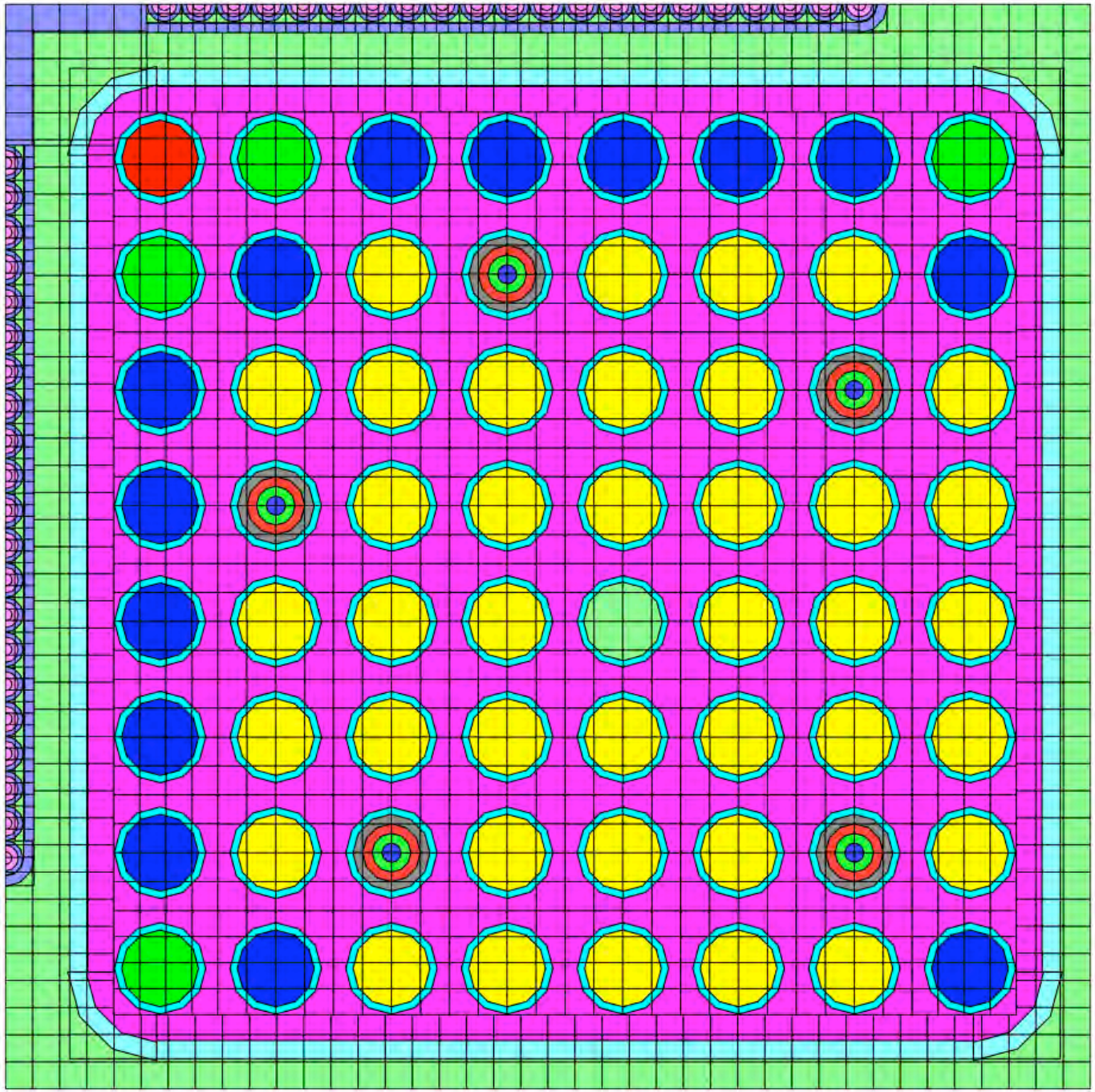
VENUS-2 MOX benchmark (1/4 core)



Simple pin cell





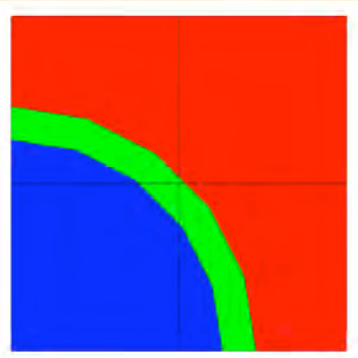


NEWT Input

- Keyword-based input
- Data is input in data blocks:
- Basic model must include these data blocks:
  - control parameters
  - mixture identification
  - geometry description
- Additional optional data blocks are used for special operations

NEWT Input - Example

```
=newt
Infinite lattice PWR pin cell
read parm
collapse=yes outers=100 epsilon=1.0e-5 kguess=1.34 prtxsec=yes
   prtmxsec=yes prtbroad=yes
end parm
read materials
  3  2  'moderator' end
  2  1  'clad'  end
  1  1  'fuel' end
end materials
read geom
global unit 1
cylinder 10  0.41000 com='fuel' chord +x=0 chord +y=0 sides=20
cylinder 20  0.4750  com='clad' chord +x=0 chord +y=0 sides=20
cuboid 30 0.65635 0.0 0.65635 0.0 com='moderator'
media 1 1 10
media 2 1 20 -10
media 3 1 30 -20
boundary 30 2 2
end geom
read bounds all=reflect end bounds
end
```

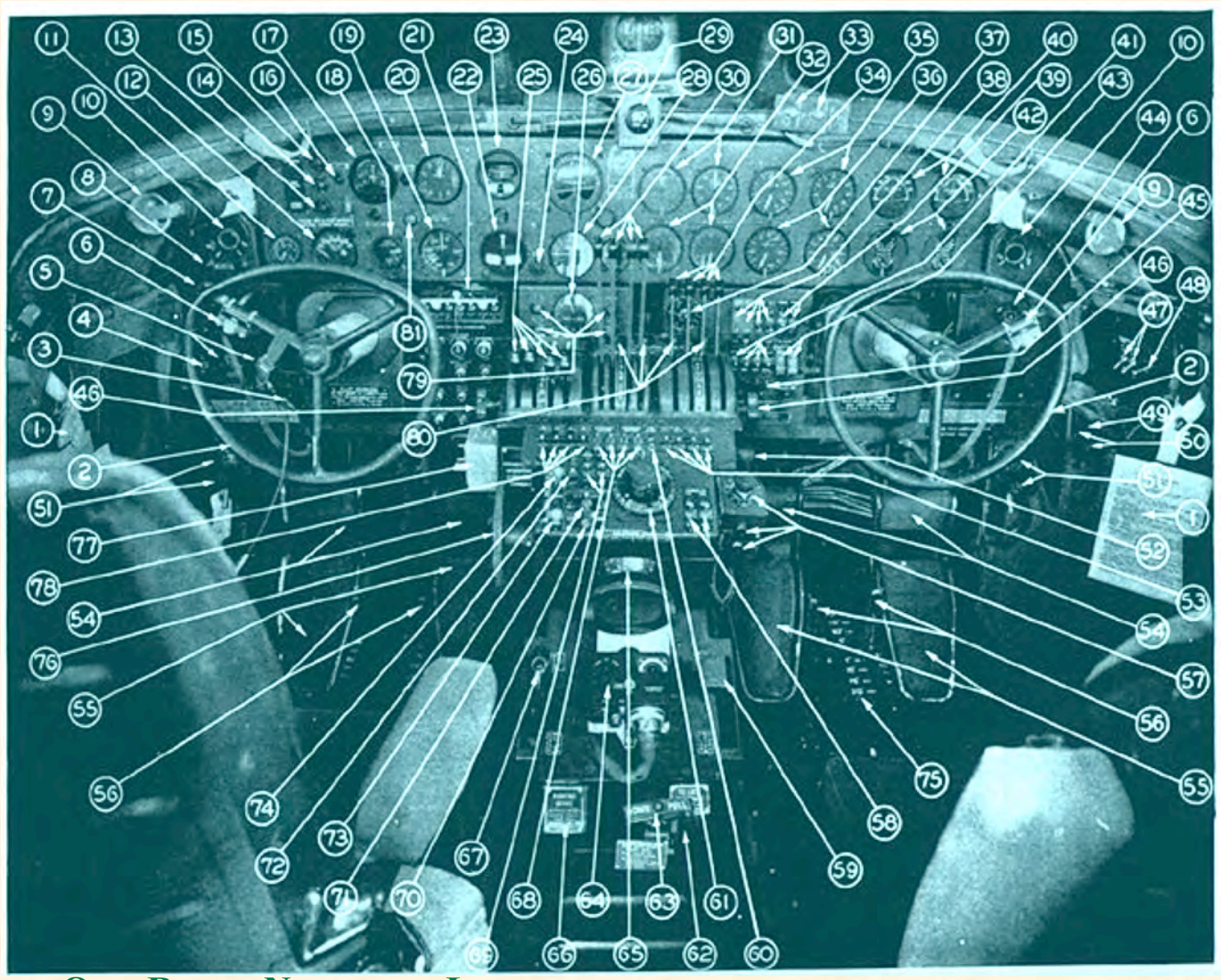


Data Blocks

READ PARAM	problem control data	END PARAM
READ MATERIAL	material data	END MATERIAL
READ GEOMETRY	model geometry specifications	END GEOMETRY
READ COLLAPSE	energy collapse data	END COLLAPSE
READ BOUNDS	boundary conditions	END BOUNDS
READ HOMOG	spatial homogenization data	END HOMOG
READ MIXT	mixing table data	END MIXT
READ ARRAY	arrays	END ARRAY
READ ADF	discontinuity factor data	END ADF
READ FLUXPLANE	flux and current edit	END FLUXPLANE

Parameter Block

So many controls, so little time!



Parameter Block

accel=(***yes***/no) - source acceleration

adjoint=(yes/**no**) - perform adjoint calculation

bf=*real number* - buckling factor **(1.420892)**

cmfd=(yes/**no**) - coarse-mesh finite-difference accelerator

combine=(**yes**/no) - allow automatic combining of small cells into larger neighbors

collapse=(yes/**no**) - perform flux-weighted cross-section collapse

drawit=(yes/**no**) - create PostScript geometry/material plots

echo=(yes/**no**) - echo outer iteration edits to standard error

epseigen=*real* - convergence criterion for k **(0.0001)**

epsinner=*real* - spatial convergence criterion for inner iters. **(0.0001)**

epsouter=*real* - spatial convergence criterion for outer iters. **(0.0001)**

epsilon=*real* - combined convergence (all of above) **(0.0001)**

Parameter Block (continued)

forward=(**yes**/no) - perform forward calculation (use forward= or adjoint=)

height=*real* - Height (transverse dimen.) used for buckling calculations **(0.0)**

hmoglib=*integer* - unit number in which homogenized cross sections are saved **(35)**

hybrid=(yes/**no**) - use combined ESC and diamond-differencing for solution.

inners=*integer* - max number of inner iterations per energy group **(4)**

kguess=*real* - initial guess for eigenvalue **(1.0)**

mixtab=*integer* - unit number from which mixing table data is read **(92)**

outers=*integer* - max number of outer iterations before halting calc. **(200)**

premix=(yes/**no**) - ignore mixing table - cross sections are premixed.

prtbroad=(yes/**no**/1d) - print broad-group (collapsed) cross-sections. 1d will not print scattering cross sections.

prtflux=(yes/**no**) - print fine-group flux edit PostScript file.

Parameter Block (continued)

prtmxsec=(yes/**no**/1d**) - print mixed (macroscopic) cross-sections. 1d will not print scattering cross sections.**

prtmxtab=(yes/no/1d**) - print input mixing table.**

prtxsec=(yes/**no**/1d**) - print microscopic cross-sections read from library. 1d will not print scattering cross sections.**

restart=(yes/**no**) **- if file *restart_newt* exists, restart=yes will read file and load fluxes prior to beginning iterations**

run=(yes/no**) - Perform iterations after initial checking and setup.**

savrest=(yes/no**) - write fluxes to file *restart_newt* after calculation ends**

solntype=(keff/buck/b1**) - Eigenvalue/critical buckling/B1 critical spectrum**

sn=(2/4/**6**/8/10/12/14/16**) - Sn quadrature order**

timed=(yes/**no**) **- print iteration timing and cpu usage data**

wtdlib=*integer* - unit number in which collapsed cross sections are saved (30)

Parameter Block (continued)

xcmfd=*integer* - number of fine-grid cells per coarse cell in x (1)

xycmfd=*integer* - number of fine-grid cells per coarse cell in x and y (1)

xnlib=*integer* - unit number from which input cross sections are read (4)

ycmfd=*integer* - number of fine-grid cells per coarse cell in y (1)

Example Parameter Block

```
read parm
   solntype=keff adjoint=no run=yes prtflux=no
   prtbroad=yes mixtab=92 xnlib=4 wtdlib=30
   collapse=yes accel=yes sn=6 inners=4 outers=100
   epsinner=1.0e-5 epsouter=1.0e-5 epseigen=1.0e-4
   kguess=1.34 restart=no prtxsec=yes prtmxsec=yes
   prtmxtab=yes
end parm

read parm collapse=yes outers=100
epsilon=1.0e-5 epseigen=1.0e-4 kguess=1.34
prtxsec=yes prtmxsec=yes prtbroad=yes end parm
```

Material Block

- **Required.**
- **Descriptions are optional.**
- **Each mix used in the calculation must be listed.**
- **Format:**

```
read materials
First Mixture ID      Pn order   'opt. description'  end
Second Mixture ID Pn order   'opt. description'  end
Third Mixture ID      Pn order   'opt. description'  end
…
end materials
```

Note: do not use a '/' inside a description
- FORTRAN issue

Example Material Block

```
read materials
  3  2  'water' end
  1  1  '3.0 enriched fuel' end
  2  1  end
end materials
```

Collapse Block

- Contains instructions to collapse the cross-sections to a reduced group structure
- For each fine-group energy group, a broad group assignment is made.
- Broad group numbers must be contiguous (constant or increasing)
- Cross sections are collapsed using the converged flux in each mixture to collapse each nuclide in each mixture.
- FIDO-type repeat factors are permitted

Example Collapse Block

- **Collapse 44 group cross sections to 3 groups (44 entries required):**

```
read collapse
1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
end collapse
```

- **Condensed version of same operation (9+17+18=44):**

```
read coll 9r1 17r2 18r3 end coll
```

or

```
read coll 9*1 17*2 18*3 end coll
```

Geometry Block

- 2D SCALE Generalized Geometry Package (2DSGGP)
- Patterned after KENO-VI input, with three exceptions
  - two-dimensional (x-y plane)
  - allows specification of underlying grid structure,
  - approximates curved surfaces (e.g., cylinders) as polygon, with control over number of sides (default is 12 sides).

Geometry Block

- **Used to specify the underlying grid(s), bodies, array and hole placement**
- **Geometry is defined in terms of *units***
- **Many units may be defined, but the global domain is defined by a single *global* unit**
- **Within each unit one may place other structure in the form of:**
  - `shapes`
  - `arrays`
  - `holes`

Geometry Block - *unit*

- Basic building blocks of model
- One (global) unit is always required
- Other units may be defined, and are placed in the global unit by arrays or holes
- Keyword *unit* is followed by a unique unit identification number

  Example: `unit 10`
- Each unit contains:
  1. shapes, arrays and/or holes
  2. media specifications
  3. boundary definition

Geometry Block - Unit: shapes

- Shapes are predefined bodies:
  - cuboid
  - cylinder
  - hexprism
  - rhexprism
- Example:

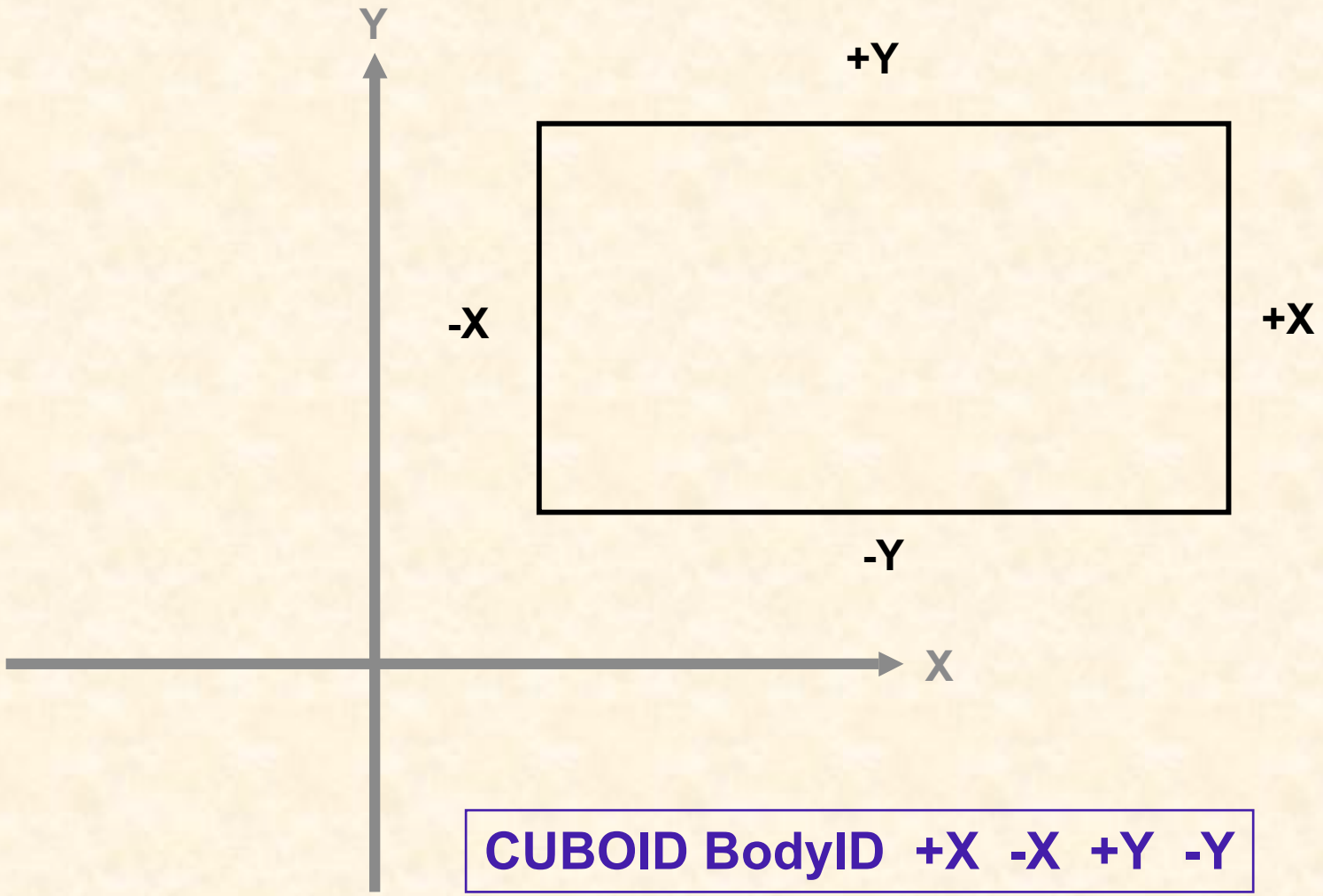
```
unit 20
  cuboid    11   1.   -1.   1.   -1.
  cylinder  12   0.8
  cylinder  13   0.6
```

- Order of shapes is not important
- Each shape must be assigned a unique ID within each unit.

2D-SGGP Geometry Input

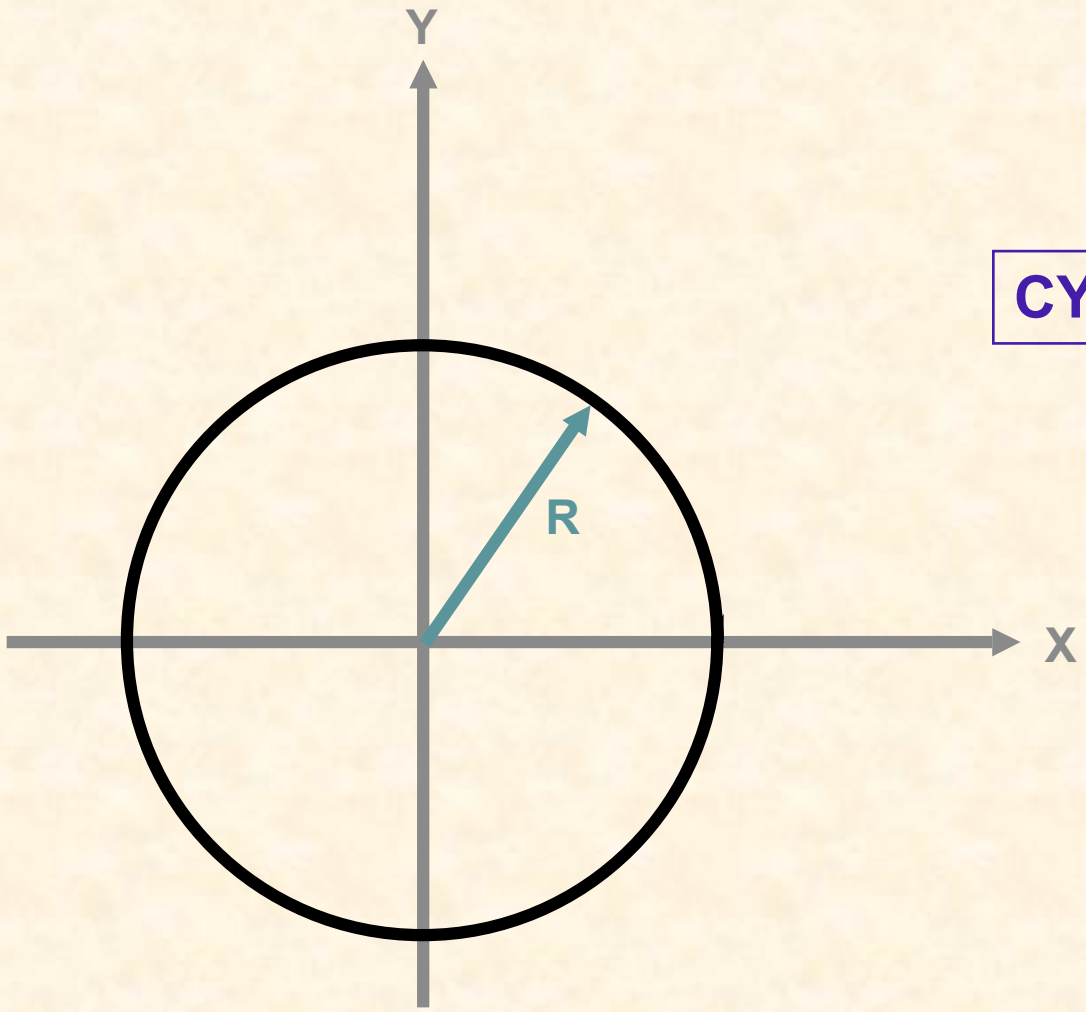
SGGP Geometry Keyword	Geometry Record Input
CUBOID	BodyID, +X, -X, +Y, -Y Positions
CYLINDER	BodyID, Radius
HEXPRISM, RHEXPRISM	BodyID, Inner Radius

Cuboid



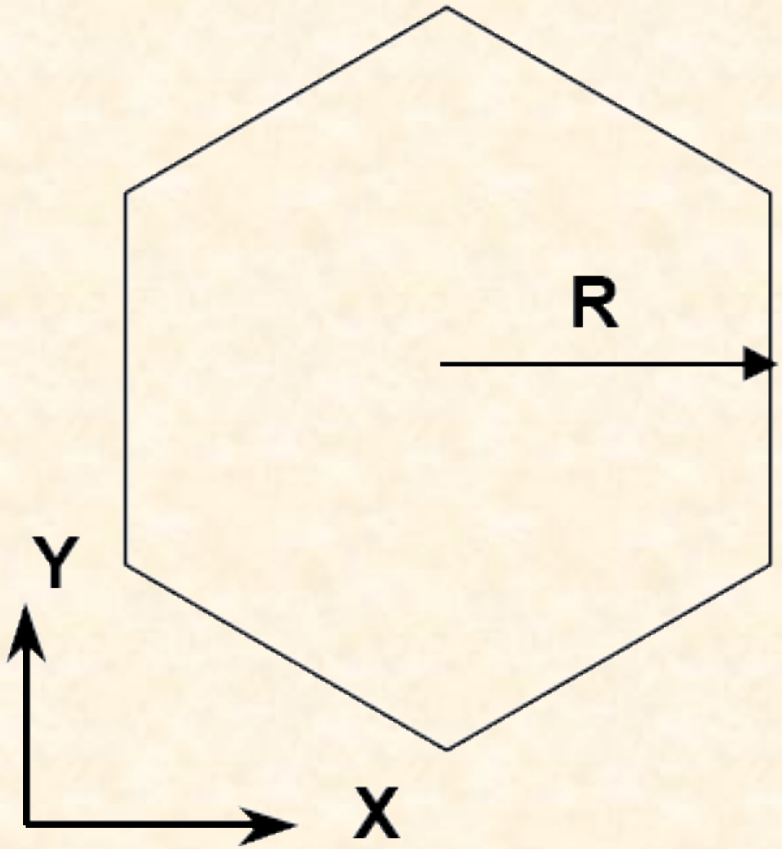
CUBOID BodyID +X -X +Y -Y

Cylinder



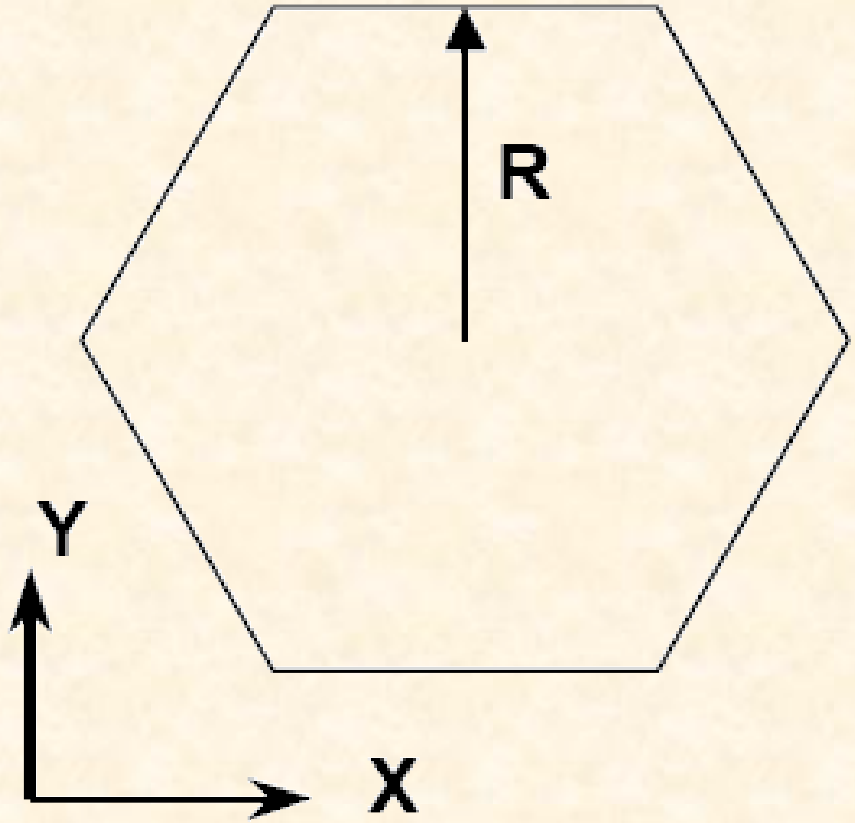
CYLINDER BodyID R

Hexagonal Prism



HEXPRISM BodyID R

Rotated Hexprism



RHEXPRISM BodyID R

Geometry Block - shape modifiers

- Every shape description may be followed by modifiers that affect that shape:
  - origin (moves origin of hexagons and cylinders)
  - chord (removes portion of hexagons and cylinders)
  - rotate (rotates body around origin - all shapes)
  - com (comment - all shapes)
  - sides (number of sides to approximate cylinder)

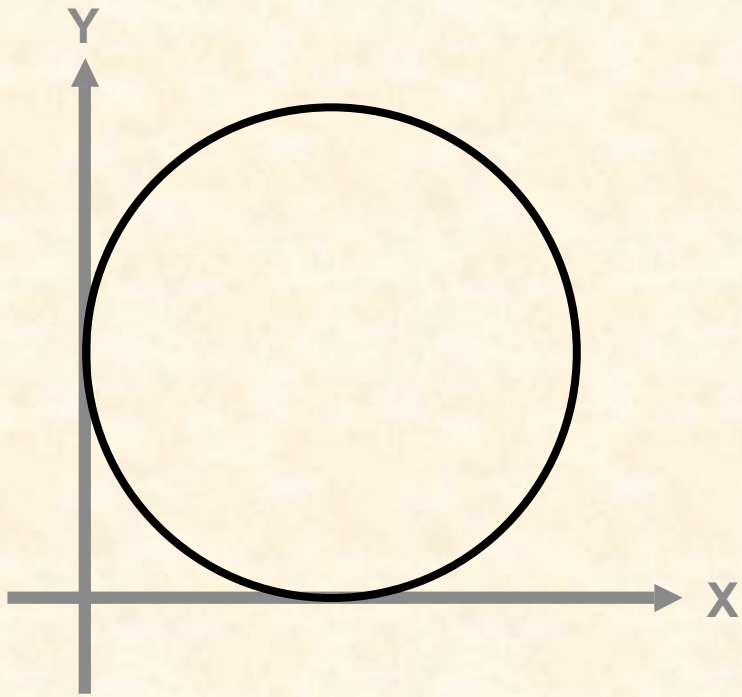
Geometry Modification Keywords

Keyword	Input Values
CHORD	+X=, -X=, +Y=, -Y= *(only one)*
ORIGIN	X=, Y= *(either or both)*
ROTATE	A1=
None	SIDES=

ORIGIN Data

- Translates the origin of a volume to a new specified position. Only non-zero data need to be entered
- Applies to the Shape or Hole Record that immediately precedes it
- Contains the following components
  - ORIGIN keyword
  - Subordinate keywords (X=, Y=)

ORIGIN Example

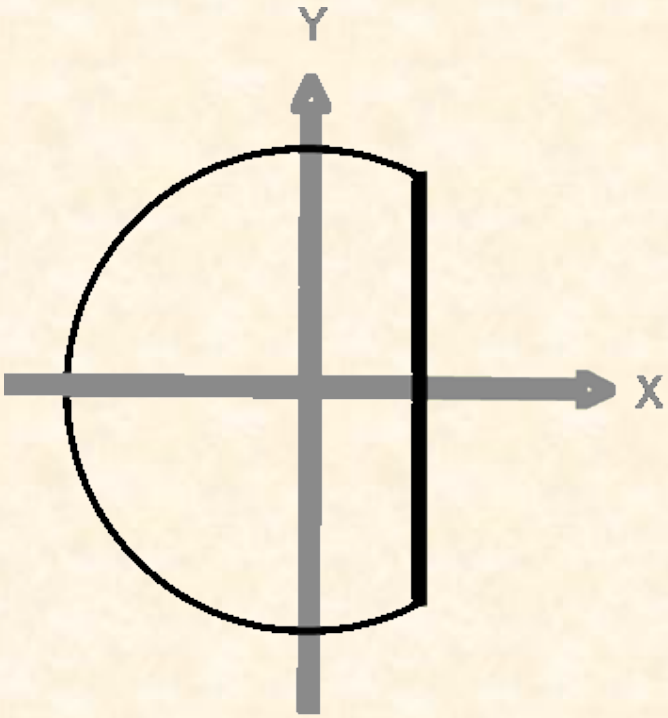


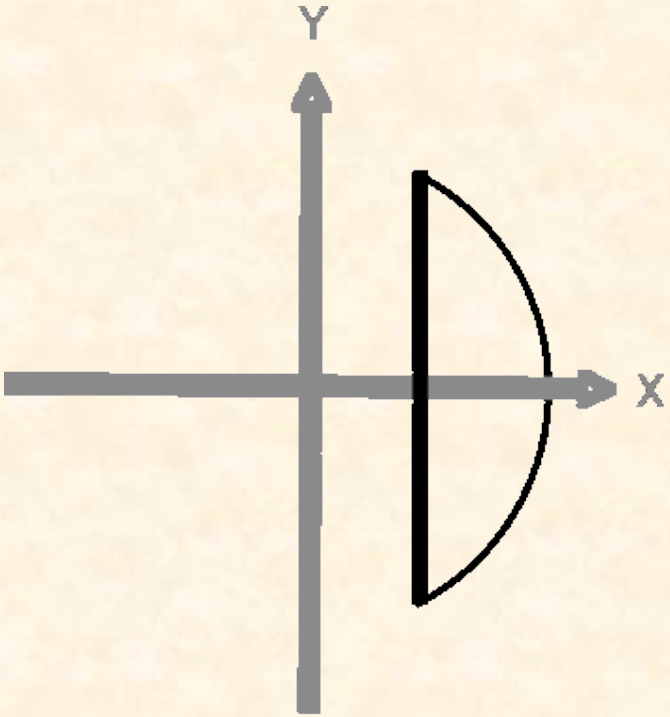
CYLINDER 1 10.0 ORIGIN X=10.0 Y=10.0

CHORD Data

- Truncates a geometric volume with a plane perpendicular to a major axis
- Applies to the Shape Record that immediately precedes it
- May not be used for CUBOIDS
- Contains the following components
  - CHORD keyword
  - A single subordinate keyword
    (+X=, -X=, +Y=, -Y=)

Chord Examples

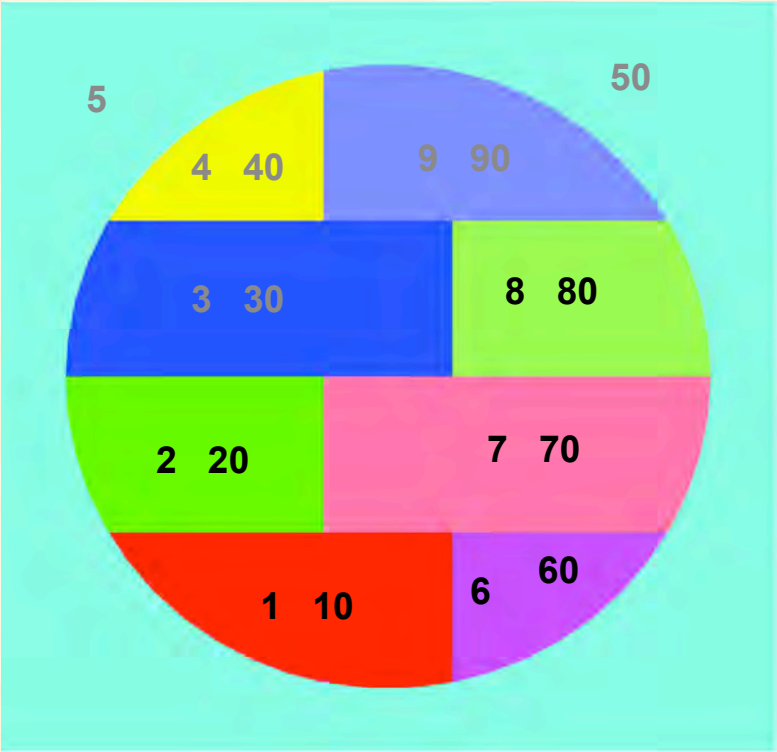




CYLINDER 1 10.0 CHORD -X=5.0

CYLINDER 2 10.0 CHORD +X=5.0

Cylinder Cut in 8 Regions Using Chords

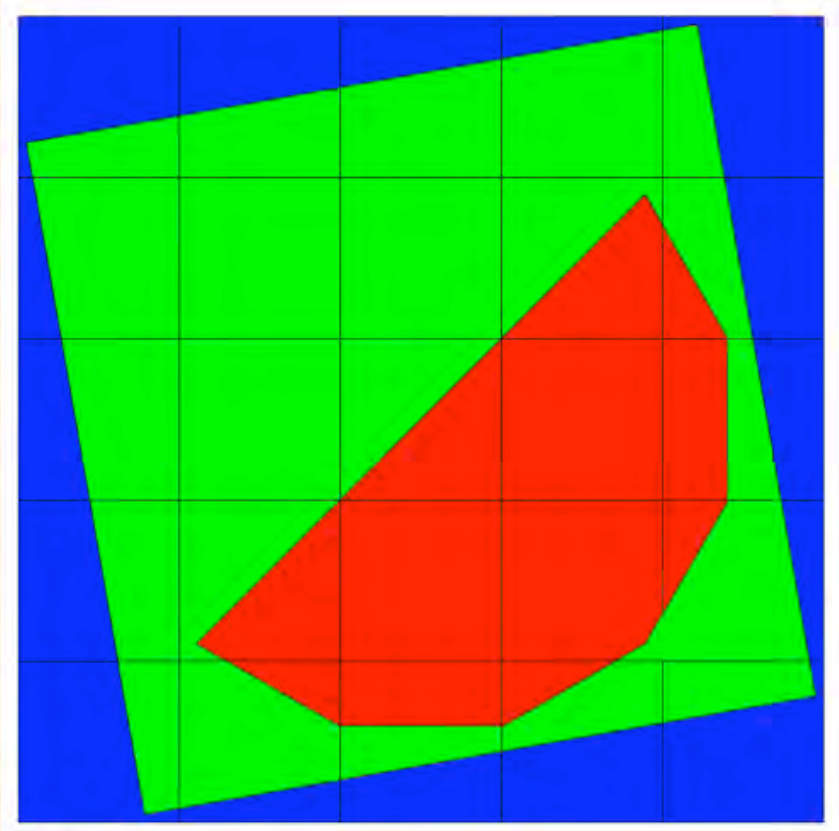



```
global unit 1
cylinder  10  10.0  chord -y=-5.0 chord -x=2.0 sides=50
cylinder  20  10.0  chord +y=-5.0 chord -y=0.0
          chord -x=-2.0 sides=50
cylinder  30  10.0  chord +y=0.0 chord -y=5.0
          chord -x=2.0 sides=50
cylinder  40  10.0  chord +y=5.0 chord -x=-2.0 sides=50
cylinder  60  10.0  chord -y=-5.0 chord +x=2.0 sides=50
cylinder  70  10.0  chord +y=-5.0 chord -y=0.0
          chord +x=-2.0 sides=50
cylinder  80  10.0  chord +y=0.0 chord -y=5.0
          chord +x=2.0  sides=50
cylinder  90  10.0  chord +y=5.0 chord +x=-2.0 sides=50
cuboid  50  4p12.0
media  5  1  50 -10 -20 -30 -40 -60 -70 -80 -90
media  1  1  10
media  2  1  20
media  3  1  30
media  4  1  40
media  6  1  60
media  7  1  70
media  8  1  80
media  9  1  90
boundary  50 12 12
```

ROTATE Data

- Rotates a volume in a counter-clockwise directions about the origin
- Rotate operation occurs <u>before</u> chord operation
- Applies to the Shape or HOLE Record that immediately precedes it
- Contains the following components
  - ROTATE keyword
  - Subordinate keyword (A1=)
- Bodies are rotated, but NEWT's grid lines are NOT rotated

ROTATE Example



```
read geom
unit 1
cylinder  10  10.0 chord +x=0.0
media 1 1 10
boundary 10

global unit 2
cuboid  20  4p11.0 rotate a1=10
cuboid 30 4p13.0
hole 1 rotate a1=-45
media  2  1  20
media  3  1  30 -20
boundary  30 5  5
end geom
```

Geometry Body Examples

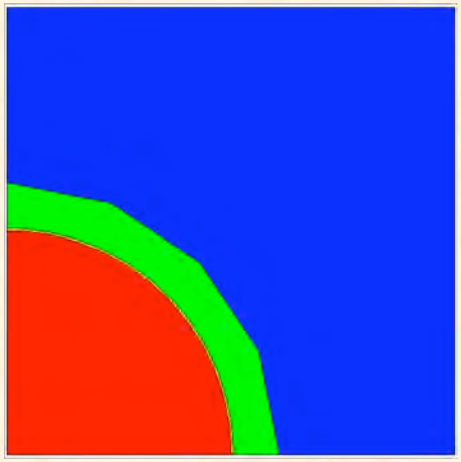
```
cuboid 3 1.0 0.0 2.0 0.0 rotate al=45
cylinder 2 5.0 origin x=2.5 y=2.5 sides=16

hexprism 13 1.0 rotate a1=90 origin y=1.0
hexprism 13 1.0 rotate a1=30 origin y=1.0
-OR-  rhexprism 13 1.0 origin y=1.0
```

Geometry Block - shapes within a unit

- **Example:**

```
global unit 1
   cylinder 10 0.5 sides=50 chord +x=0 chord +y=0 com='fuel'
   cylinder 20 0.505 sides=50 chord +x=0 chord +y=0 com='gas gap'
   cylinder 30 0.600 sides=16 chord +x=0 chord +y=0 com='clad'
   cuboid 40 1.0 0.0 1.0 0.0 com='moderator'
```



Geometry Block - Holes

- Holes are used to place a single unit within another unit.
- Similar to an array, but places only one unit
- Multiple holes may be placed within a unit
- Useful when cells are repeated but not in regular pattern
- Format:

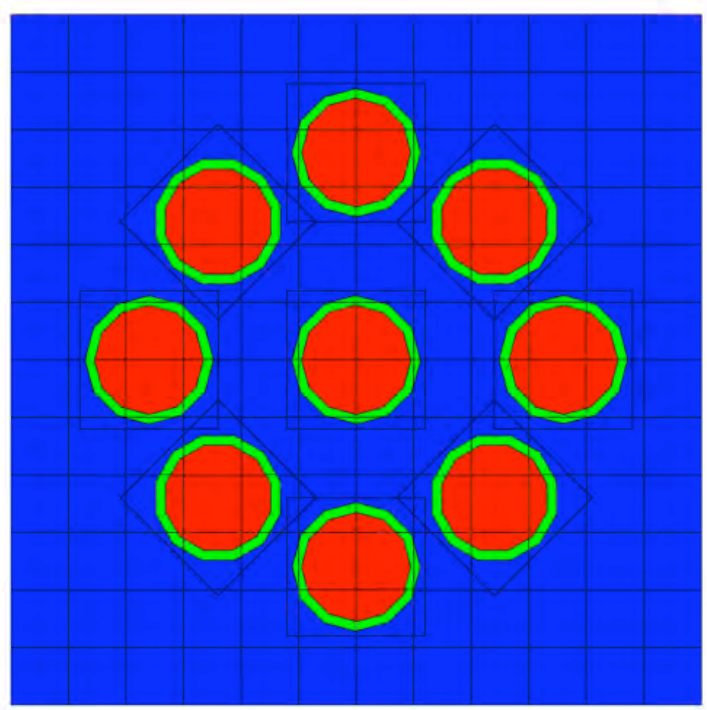
```
hole unitid [origin x= y=] [rotate a1=]
```

- origin places (0,0) of unit at a new location

Geometry Block - Holes

```
unit 1
cylinder 30 .3860
cylinder 40 .4550
cuboid   50 4p0.5
…

global unit 2
hole 1
hole 1 origin x=1. y=1. rotate a1=45
hole 1 origin x=-1. y=1. rotate a1=45
hole 1 origin x=-1. y=-1. rotate a1=45
hole 1 origin x=1. y=-1. rotate a1=45
hole 1 origin x=0 y=1.5
hole 1 origin x=0 y=-1.5
hole 1 origin x=1.5 y=0
hole 1 origin x=-1.5 y=0
cuboid 10 4p2.5
…
```



Geometry Block - Arrays

- Arrays are used to place a one or more units within another unit in a regular pattern
- Multiple arrays may be placed within a unit; arrays may be placed in arrays and in holes
- Unlike holes, arrays cannot be rotated (unless placed inside a hole
- Arrays are placed within a shape.

Geometry Block - Arrays

- **Format:**

`array` *`arrayid bodyid`* `place` *`ix iy`* x_{ix} y_{iy}

- **Arrays are defined in the *Array* data block. This command is used to place an array in space.**

- **Data following the keyword PLACE used to position the array in a region consists of:**
  - **the (I, J) position of the unit in the array used to establish the location of the array**
  - **the (X, Y) coordinates of the surrounding unit, containing the origin of the unit used to position the array**

Geometry Block - Unit:media

- Shape input is used to define bodies within unit
- Media input follows shape specifications within a unit
- Media input is used to define mixtures associated with bodies
- Format:

```
media materialid bias_id region_def_vector
```

- The *bias_id* entry is not (currently) used, but is kept for consistency with KENO, and is required. Value is arbitrary.

Geometry Block - Unit:media

- **Region definition vector**
  - **describes the location of a given mixture (media) within a model**
  - **provides list of shapes (by shape ID number) for which the media is either *inside* or *outside***
  - **If a media is *inside* a body, the body is listed with a positive number**
  - **if a media is outside a body, the body is listed with a negative number**

Geometry Block - Unit:media

- **Example:**

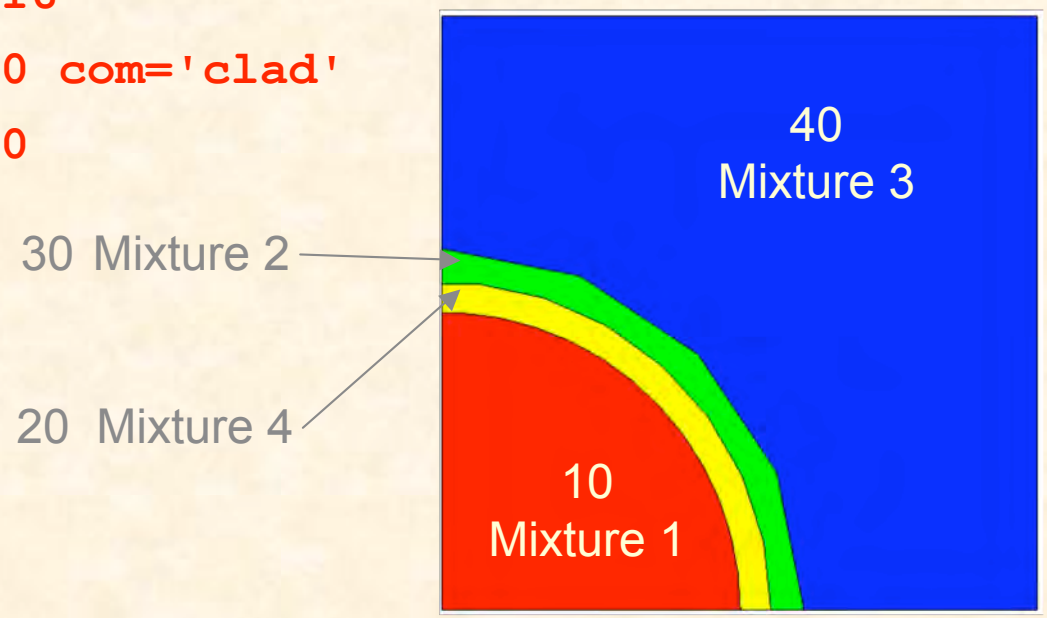
```
media 1 0 10
media 4 0 20 -10
media 2 0 30 -20
media 3 0 40 -30
```

  - mixture 1 is inside body 10
  - mixture 4 is inside body 20 but outside body 10
  - mixture 2 is inside body 30 but outside body 20
  - mixture 3 is inside body 40 but outside body 30

Geometry Block - Unit:media

- **Example:**

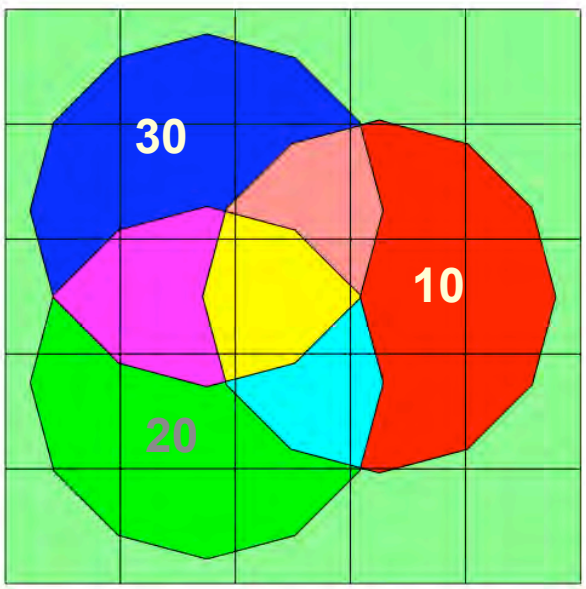
```
global unit 1
cylinder 10 0.5 sides=50 chord +x=0 chord +y=0 com='fuel'
cylinder 20 0.505 sides=50 chord +x=0 chord +y=0 com='gas gap'
cylinder 30 0.600 sides=16
    chord +x=0 chord +y=0 com='clad'
cuboid 40 1.0 0.0 1.0 0.0
    com='moderator'
media 1 0 10
media 4 0 20 -10
media 2 0 30 -20
media 3 0 40 -30
```



Geometry Block - Unit:media

- **Media specifications use a logical AND to define regions. This allows intersections and specification of intersecting regions**

```
global unit 1
cylinder 10 9.0 origin x=4.5
cylinder 20 9.0 origin y=-4.5 x=-4.5
cylinder 30 9.0 origin y=4.5 x=-4.5
cuboid 40 4p15
media 1 0 10 -20 -30
media 2 0 20 -10 -30
media 3 0 30 -10 -20
media 4 0 10 20 30
media 5 0 10 20 -30
media 6 0 20 30 -10
media 7 0 10 30 -20
media 8 0 40 -10 -20 -30
```



Geometry Block - Unit: boundary

- Boundary definition follows all shape and media input
- Only one boundary specification may be input
- Boundary definition designates outermost body of unit (i.e., unit boundary)
- Boundary definition also provides the underlying grid for each unit
- Format:

```
boundary bodyid x_grid y_grid
```

Geometry Block - Unit: boundary

- **Example:**

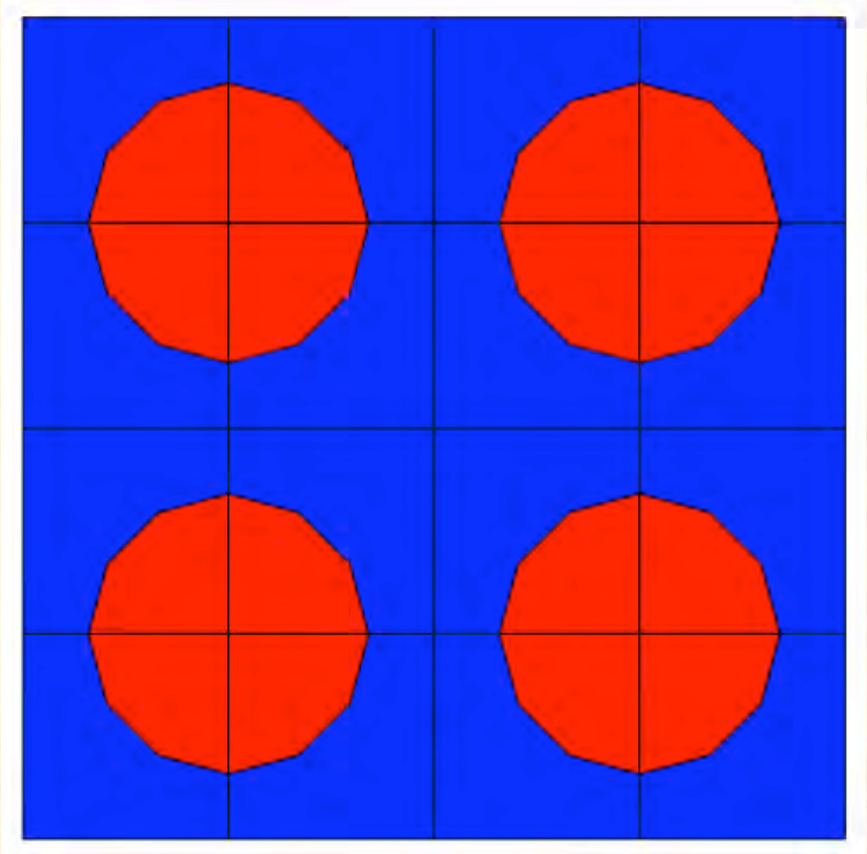
```
boundary 40 5 5
```

  - Body 40 is the outermost body of this unit
  - The unit has an underlying 5x5 rectangular grid

- **Global unit MUST have a base grid defined**
- **Other units may have a local grid defined, but if omitted the global grid will be used.**
- **This is best shown by examples.**

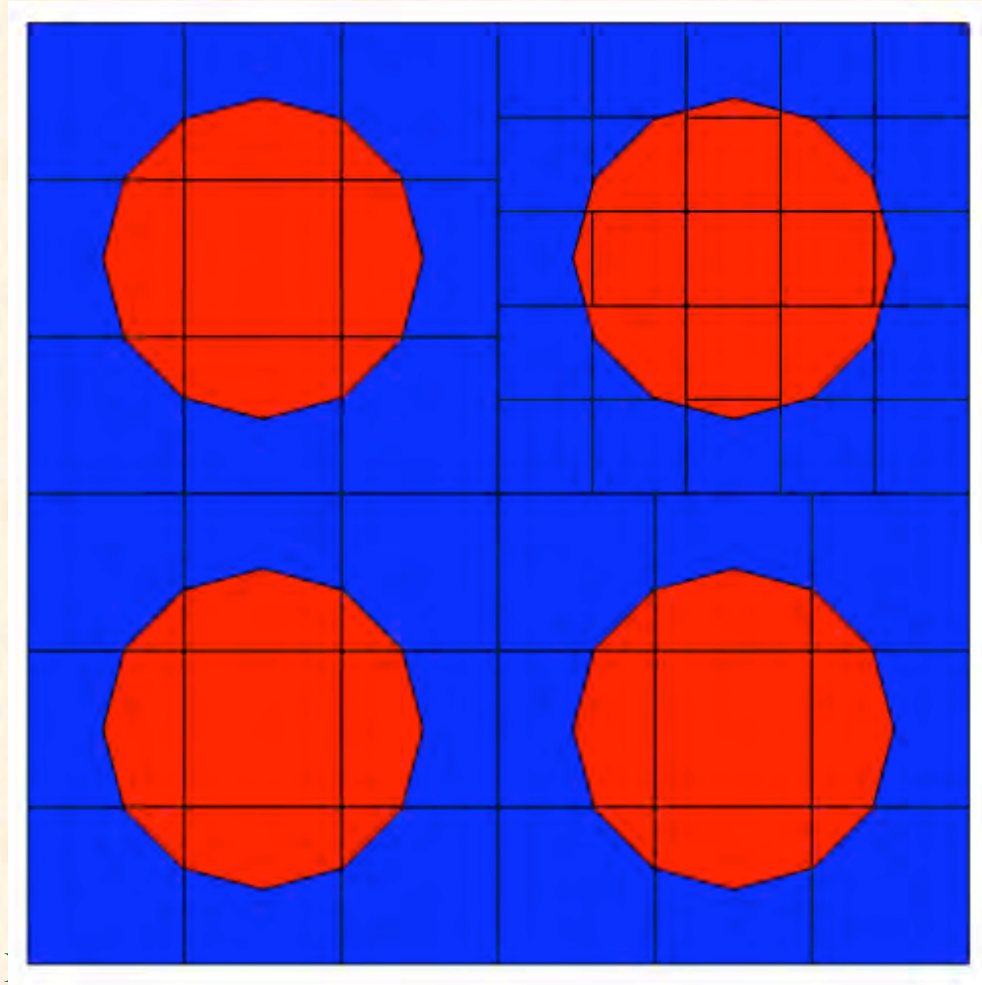
Geometry Block - Full unit description

```
read geom
global unit 1
cylinder 10 0.5
cylinder 20 0.5 origin x=1.5
cylinder 30 0.5 origin y=1.5
cylinder 40 0.5
   origin x=1.5 y=1.5
cuboid 50 2.25 -0.75 2.25 -0.75
media 1 0 10
media 1 0 20
media 1 0 30
media 1 0 40
media 3 0 50 -10 -20 -30 -40
boundary 50 4 4
end geom
```



Geometry Block - Full unit description

```
read geom
unit 1
cylinder 10 0.5
cuboid 20 0.75 -0.75 0.75 -0.75
media 1 0 10
media 3 0 20 -10
boundary 20 5 5
unit 2
cylinder 10 0.5
cuboid 20 0.75 -0.75 0.75 -0.75
media 1 0 10
media 3 0 20 -10
boundary 20
global unit 3
rectangle 10 3.0 0.0 3.0 0.0
array 1 10  place 1 1 0.75 0.75
media 3 0 10
boundary 10 6 6
end geom
```



Bounds Block

- **Bounds block is used to define boundary conditions.**
- **Four boundaries exist:**

+x=	(right side)
-x=	(left side)
+y=	(top side)
-y=	(bottom side)

- **Combined Boundary Conditions:**

+fc=, +xy=	(+x and +y)
-fc=, -xy=	(-x and -y)
all=, xyf=	(all four surfaces)
xfc=	(+x and -x)
yfc=	(+y and -y)

Bounds Block

- Four boundary conditions are supported
  - vacuum
  - reflective
  - white
  - periodic
- Use as few or as many characters as desired:

  +x=v +x=vac +x=vacuum

  -xy=r -xy=ref -xy=refl
- Default BC, if not specified, is reflective

Bounds Block

- Bounds block always comes after geom block
- Examples:

```
read bounds all=vac end bounds

read bounds
xfc=periodic
yfc=refl
end bounds

read bounds
+x=vac +y=vac -x=ref -y=ref  end bounds
```

PostScript Plot Files

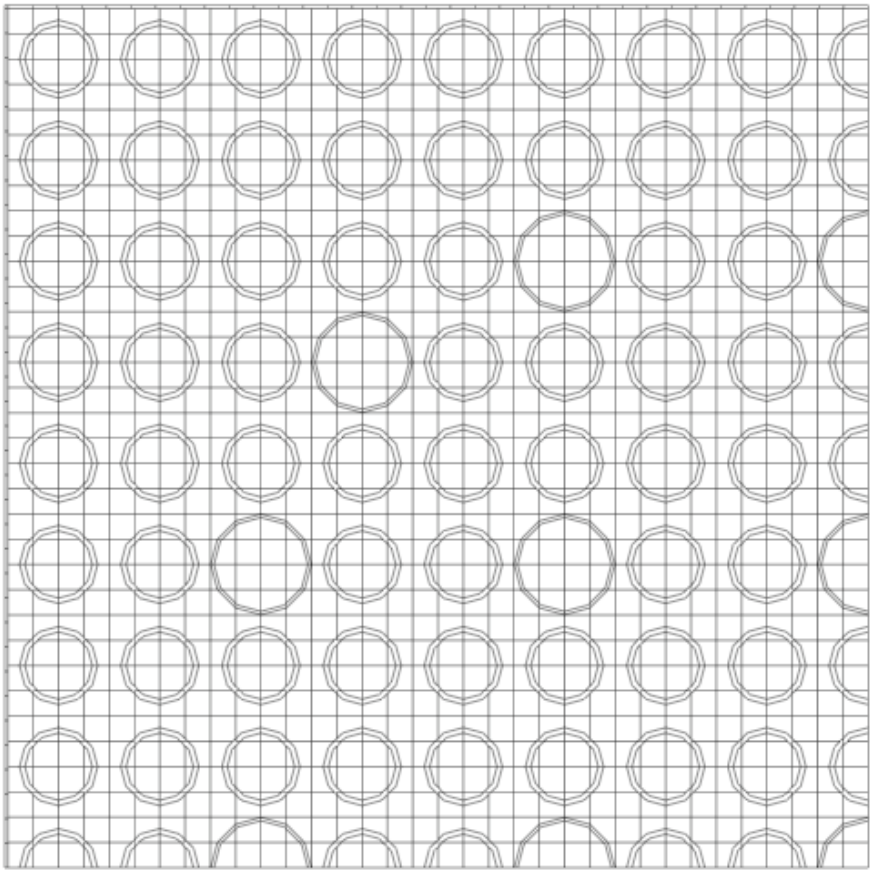
- **NEWT provides simple PostScript plot files for input review and output analysis.**
- **Generation of PostScript plots is enabled in NEWT's *read param* data block.**
  - ***drawit=yes* will generate two plot files:**

    ***jobname*.newtgrid.ps**

    ***jobname*.newtmatl.ps**
    - **These file are automatically copied back to your home directory**
  - ***prtflux=yes* generates the file cleo.ps in the working directory. This file is not automatically copied back.**

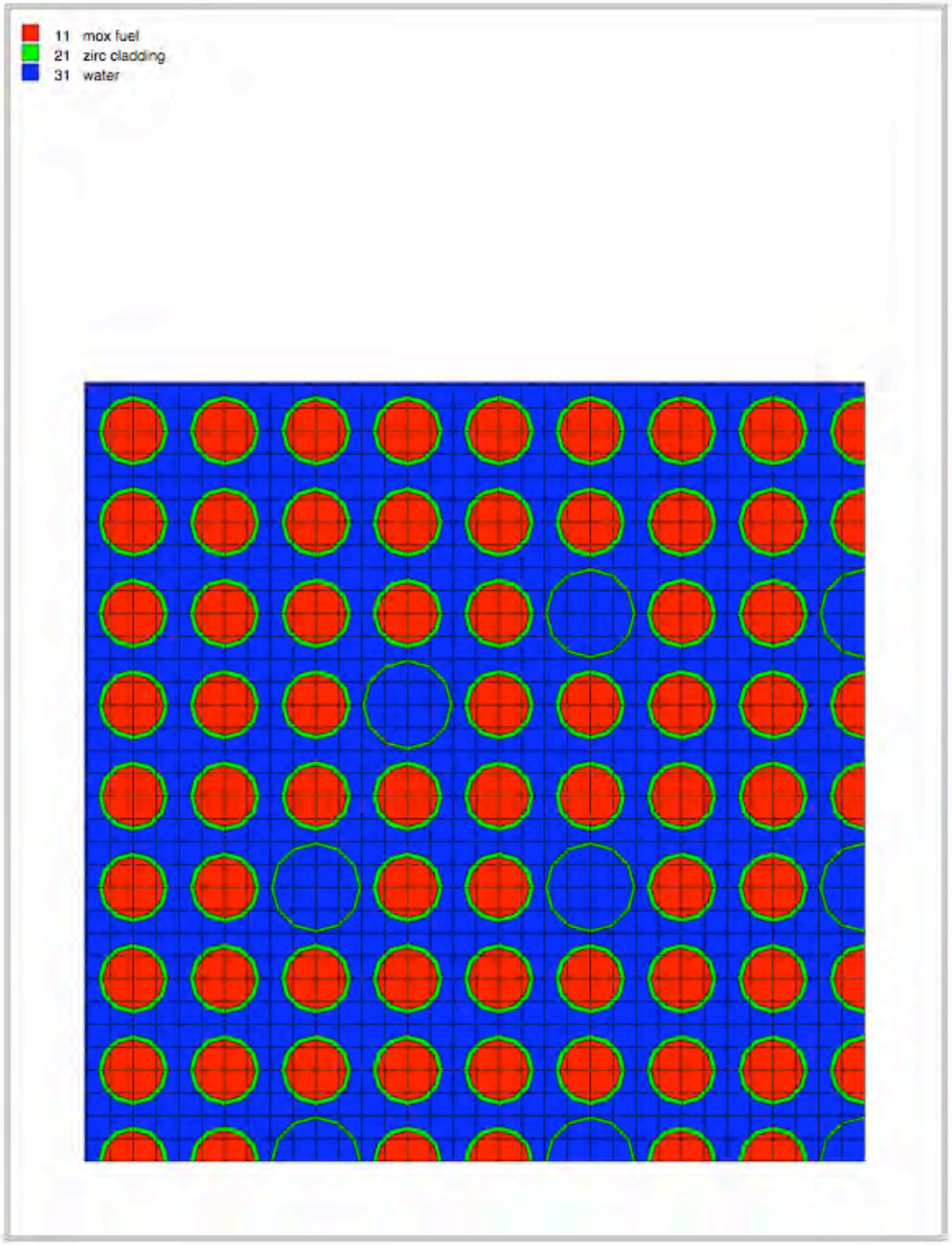
newtgrid.ps

Generated first, if a consistent geometry model is read.



newtmatl.ps

Generated second, after media placement operations are completed.

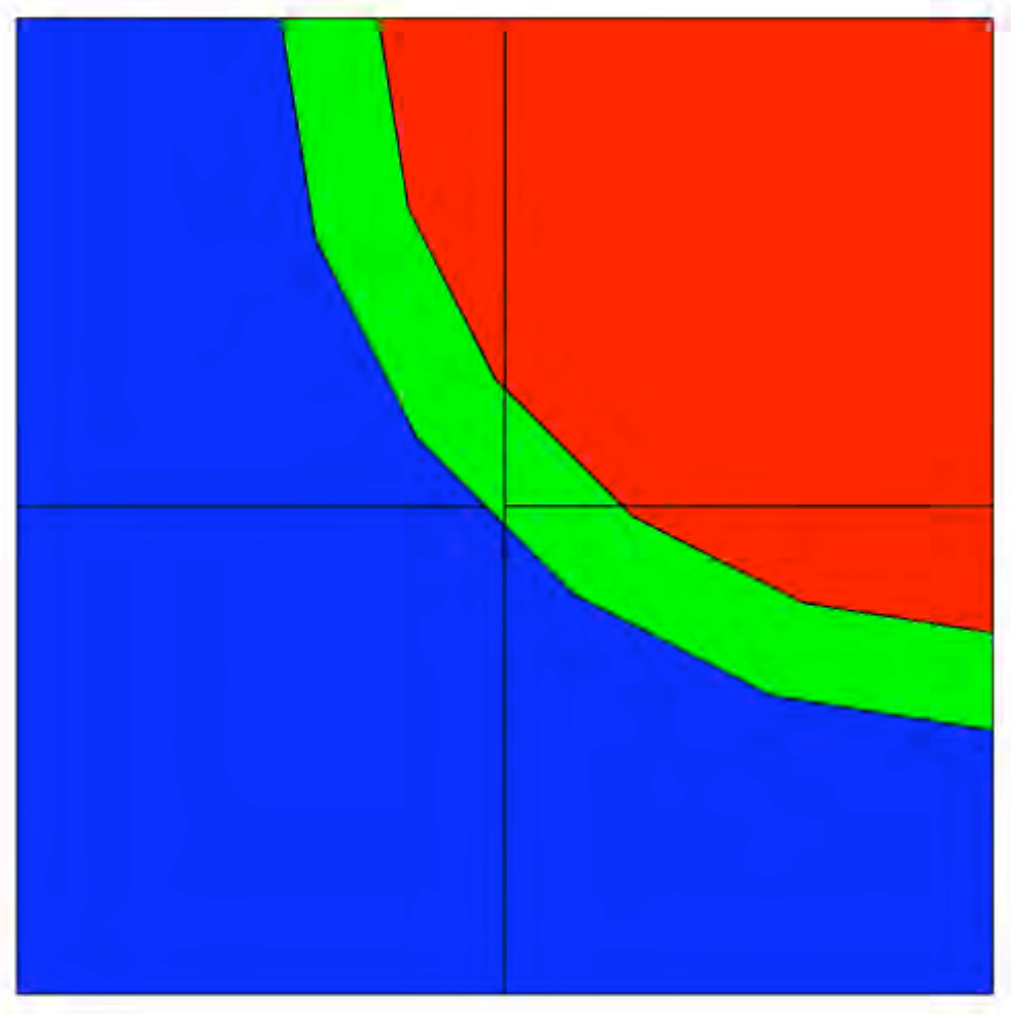


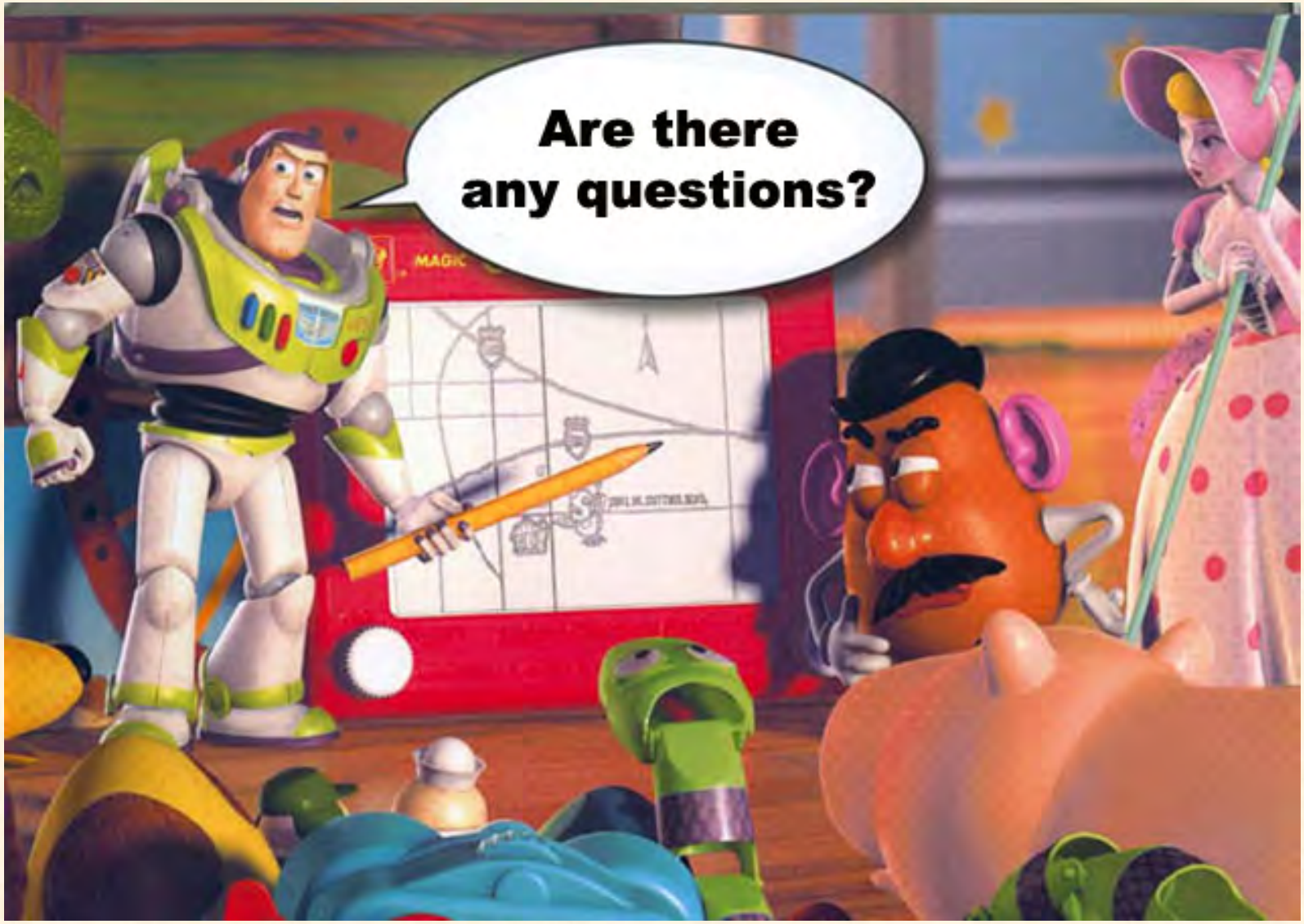
OAK RIDG
U. S. DEPAR

NEWT Input - Examples

```
=newt
Infinite lattice PWR pin cell
read materials    No parameter block – all defaults used
  3  2  end       Collapsed cross sections written to ft30f001
  2  1  end
  1  1  end
end materials
read collapse 7r1 2 3 2r4 5 6 7 8 8 8r9 14r10 6r11 10r12
13 7r14 11r15 12r16 30r17 16r18 2r19 6r20 3r21 6r22 14r23
27r24 10r25 5r26 27 28 29 2r30 31 32 33 2r34 2r35 3r36
2r37 38 39 40 41 42 3r43 9r44 end collapse
read geom
global unit 1
cylinder 10 .410 chord -x=0 chord -y=0 sides=20
cylinder 20 .475 chord -x=0 chord -y=0 sides=20
cuboid 30 0.0 -0.65635 0.0 -0.65635
media 1 1 10
media 2 1 20 -10
media 3 1 30 -20
boundary 30  2 2
end geom
read bounds all=refle  end bounds
end
```

NEWT Input - Examples




Are there any questions?

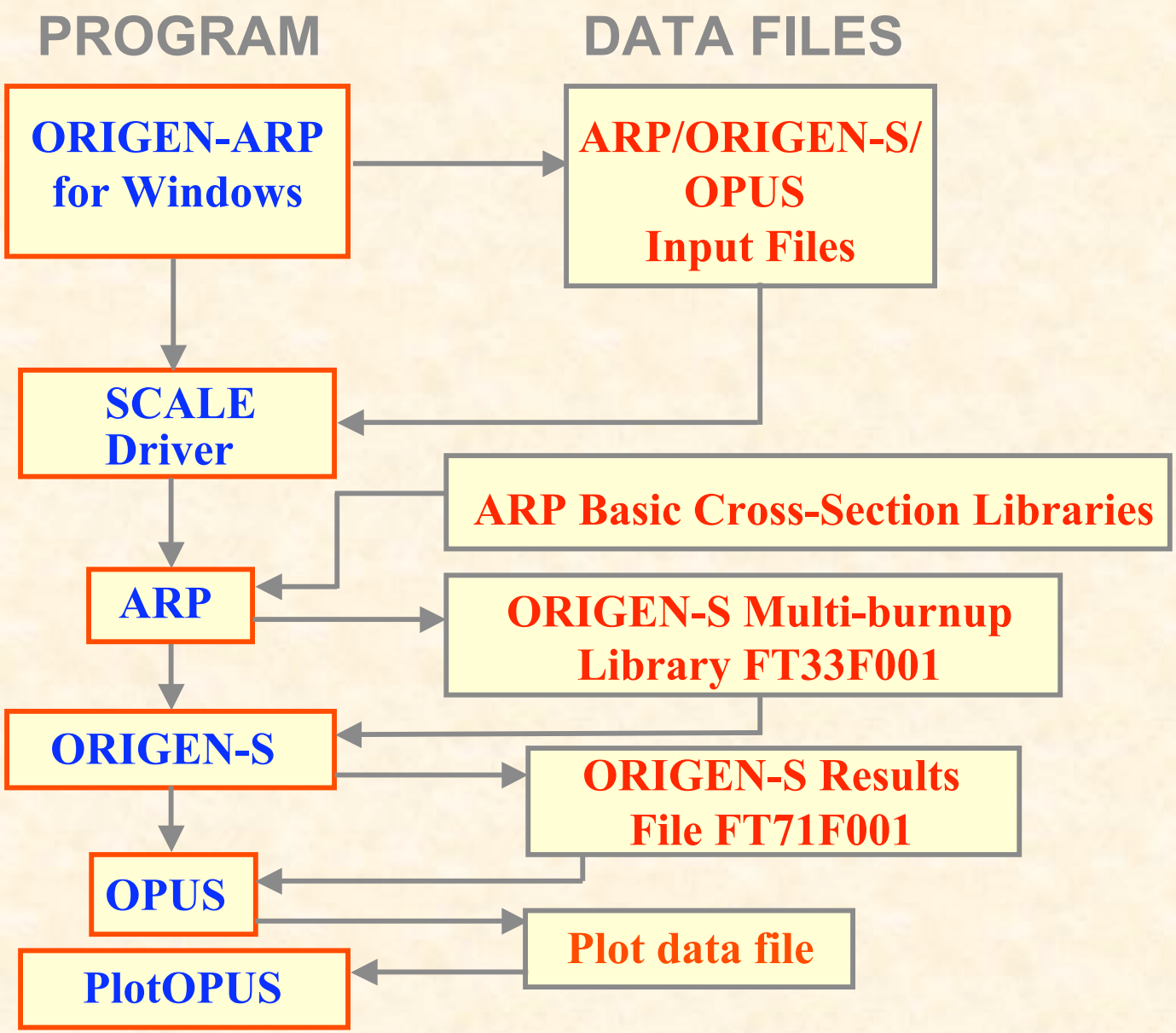
Applications with TRITON

TRITON Capabilities

- Generation of libraries for ORIGEN-ARP
- Restart to simulate fuel shuffling
- Generation of broad group master libraries
- Cross sections for nodal analysis
- Forward and adjoint solutions for S/U calculations
- Faster CENTRM processing and mixed methods
- Reduced cross section processing for large problems

ORIGEN-ARP

- Fast, Easy, and Accurate Fuel Depletion, Decay, and Source Terms
- Allows easy setup of ORIGEN-S depletion and decay cases
- Interactive Windows program with on-line help
- Generates input files for ARP, ORIGEN-S, and OPUS on PC or workstation
- Executes code sequence on PC
- Runs very fast – in a few seconds


PROGRAM
DATA FILES
ORIGEN-ARP for Windows
ARP/ORIGEN-S/ OPUS Input Files
SCALE Driver
ARP Basic Cross-Section Libraries
ARP
ORIGEN-S Multi-burnup Library FT33F001
ORIGEN-S
ORIGEN-S Results File FT71F001
OPUS
Plot data file
PlotOPUS

ORIGEN-ARP Methodology

- Generates ORIGEN-S cross-section libraries by interpolation of existing cross section data
- Interpolation parameters for uranium fuels are burnup, enrichment, and moderator density (optional)
- MOX interpolation parameters are total Pu content, 239Pu concentration, burnup, and moderator density
- Easy-to-use Windows program offers 2 input options:
  - Origen Express form
  - Detail Input forms
- Multi-burnup libraries are generated for different assembly designs
- Libraries generated by TRITON - one library for each initial fuel enrichment and moderator density combination

ORIGEN-ARP Libraries

- UO2 LWR Fuel types
  - 7 x 7 BWR, 8 × 8 BWR
  - 14 × 14 PWR, 15 × 15 PWR, 17 × 17 PWR
- Enrichments - 1.5 to 5.0 wt %
- Burnups - 0 to 60 GWd/MTU
- CANDU reactor (natural UO_2)
- MAGNOX (natural U metal)
- Advanced Gas-Cooled Reactor (AGR)
- Russian VVER 440, VVER 1000
- MOX fuels (PWR and BWR assemblies)
  - 8x8-2, 9x9-9, 9x9-1, 10x10-9
  - 14x14, 15x15, 16x16, 17x17, 18x18
  - Variable moderator density for BWR designs

Generation of ARP Cross Sections

```
=t-depl
Westinghouse 17x17 PWR pin cell
238groupndf5
read comp
uo2   1 den=10.25 1 900 92235 3.101 92238 96.899  end
wtptzirc 2 6.44 5 40000 97.91 26000 0.5
         50116 0.318 50118 0.5406 50120 0.7314 1 633 end
h2o    3 den=0.72    1  576 end
wtpt-bormod 3 0.72   1 5000 100 500.0–6 576 end
end comp
read celldata
latticecell squarepitch pitch=1.26 3 fuelr=0.3922 1
            cladr=0.4572  2 end
end celldata
read depletion 1  end depletion
read burndata
power=25.0 burn=0.01     down=0   nlib=1  end
power=25.0 burn=1500.0   down=0   nlib=20 end
end burndata
read keep_output
newt opus
end keep_output
```

Generation of ARP Cross Sections

```
read model
Westinghouse 17x17 PWR pin cell
read materials
 1  1  ! fuel !  end
 2  1  ! clad !  end
 3  2  ! water !  end
end materials
read geom
global unit 1
com='regular fuel rod'
cylinder 10 .3922 chord +x=0 chord +y=0
cylinder 30 .4572 chord +x=0 chord +y=0
cuboid   40  0.63 0.0 0.63 0.0
media 1 1 10
media 2 1 30 -10
media 3 1 40 -30
boundary 40 10 10
end geom
read bounds all=refl end bounds
end model
end
=shell
cp ft33f001* $RTNDIR/
end
```

Restart to Simulate Fuel Shuffling, etc.

- At the end of a depletion calculation, TRITON saves the concentrations of all nuclides (for which master library x-secs exist) in a preformatted file.
- The file can be automatically or manually included in a subsequent calculation.
- Can be used for any kind of change - the results on one calculation are readily available and formatted for the next calculation.

TRITON Restart

```
=t-depl
Depletion with BPR
44group
read comp
uo2 1 den=9.550 1 743  92234 0.023 92235 2.561 92236 0.013 92238 97.403 end
wtptzirc  2 6.44 4 40000 97.91 26000 0.5 50116 0.86 50120 0.73 1.0 620  end
h2o 3 den=0.7544 1 559 end
wtptbor 3 0.7544 1 5000 100  652.5e-6 559 end
wtptbpr 4 2.081 6 8016 53.58 11000 2.82 13027 1.758
                  14000 37.63 19000 0.33  5000 3.882 1 559 end
ss304 6 1 559.0 end
end comp
read celldata
   latticecell squarepitch pitch=1.43 3 fueld=0.9484 1 cladd=1.0719 2 end
end celldata
read depletion
  1 4
end depletion
read burndata
  power=20.86 burn=243.5 down=104.0  nlib=4 end
end burndata
read model
Depletion with BPR
read parm echo=yes sn=4 collapse=yes epsilon=1e-3 cmfd=yes xycmfd=3 end parm
```

TRITON Restart

```
read geom
  unit   1
  com='fuel rod cell'
  cylinder  1 1 0.4742
  cylinder  2 1 0.53595
  cuboid    3 1 4p0.715
  unit   2
  com='BRP/water hole location'
  cylinder  3 1 0.28385
  cylinder  6 1 0.30035
  cylinder  4 1 0.50865
  cylinder  6 1 0.55755
  cuboid    3 1 4p0.715
global unit 10
array 1 1 place 1 1 0 0
end geom
read array
ara=1 nux=5 nuy=5 fill
1 1 1 1 1
1 1 1 1 1
1 1 2 1 1
1 1 1 1 1
1 1 1 1 1 end fill
end array
read bounds  all=reflect end bounds
end model
end
```

TRITON Restart

```
=t-depl
Depletion restart
44group
read comp
< stdcmp_mix0001
wtptzirc  2 6.44 4 40000 97.91 26000 0.5 50116 0.86 50120 0.73 1.0 620  end
h2o 3 den=0.7544 1 559 end
end comp
read celldata
   latticecell squarepitch pitch=1.43 3 fueld=0.9484 1 cladd=1.0719 2 end
end celldata
read depletion
  1
end depletion
read burndata
  power=19.57 burn=156.0 down=39.0  nlib=1 end
  power=18.11 burn=156.0 down=3936.0  nlib=1 end
end burndata
read model
Depletion restart
read parm echo=yes sn=4 collapse=yes epsilon=1e-3 cmfd=yes xycmfd=3 end parm
```

TRITON Restart

```
read geom
  unit   1
  com='fuel rod cell'
  cylinder  1 1 0.4742
  cylinder  2 1 0.53595
  cuboid    3 1 4p0.715
  unit   2
  com='water hole location'
  cuboid    3 1 4p0.715
global unit 10
array 1 1 place 1 1 0 0
end geom
read array
ara=1 nux=5 nuy=5 fill
1 1 1 1 1
1 1 1 1 1
1 1 2 1 1
1 1 1 1 1
1 1 1 1 1 end fill
end array
read bounds  all=reflect end bounds
end model
end
```

Generation of Broad Group Master Libraries

- SCALE 44-group ENDF/B-V library was collapsed from the 238-group library using an LWR spectrum.
  - This library was often misused and abused
  - SCALE 5.1 will include an ENDF/B-VI 238-group library, but no collapsed 44-group library.
- TRITON now provides the ability to create a new, <u>appropriately-weighted</u> broad-group library from a fine-group master
- Works only for t-newt calculations

Generation of Broad Group Master Libraries

- To invoke, run a T-NEWT calculation as:

```
=t-newt     parm=weight
or
=t-newt     parm=(weight=N)
```

- If "N" is specified, weighting is performed using the NEWT-calculated flux in mixture "N"
- If N is not specified, weighting is performed using the system-averaged flux
- A master format library named *newxnlib* is created in the temporary directory.

Generation of Broad Group Master Libraries

```
=t-newt   parm=(weight=30)
Magnox Unit Cell
238GROUPNDF5
read comp
wtptFuel 10  17.2  4 92235 0.711 92234 0.0054 13027 0.5 92238 98.7836 1 650 end
wtptMgClad      20  1.65  2 12000 99.5 40000 0.5 1 650 end
u-234    30 0 0.0000500e-05 549 end
u-235    30 0 0.0072000e-05 549 end
u-238    30 0 0.9927500e-05 549 end
b-10     30 0 .199e-10 549 end
b-11     30 0 .801e-10 549 end
li-6     30 0 .075e-10 549 end
li-7     30 0 .925e-10 549 end
c-12     30 den=1.70 1 549 end
wtptCO2         40  0.137  2 8016 72.7 6000 27.3 1 600 end
end comp
read celldata
latticecell squarepitch pitch=20 30 fuelr=1.45 10 gapr=2.55 40 cladr=2.6 20 end
end celldata
read model
Magnox Unit Cell
read parm drawit=yes epsilon=.0001 cmfd=yes xycmfd=3 end parm
```

Generation of Broad Group Master Libraries

```
read collapse
7r1 2 3 2r4 5 6 7 2r8 8r9 14r10 6r11 10r12 13
7r14 11r15 12r16 30r17 16r18 2r19 6r20 3r21 6r22 14r23
27r24 10r25 5r26 27 28 29 2r30 31 32 33 2r34 2r35 3r36
2r37 38 39 40 41 42 3r43 9r44
end collapse
read materials
  10      1 !  fuel! end
  20      1 !  cladding! end
  30      1 !  moderator! end
  40      1 !  coolant! end
end materials
```

Generation of Broad Group Master Libraries

```
read geom
global unit 1
cylinder 1 2.9 sides=16
cylinder 2 3.0 sides=16
cylinder 3 5.0 sides=16
cuboid   4 4p10.0
media 10 1 1
media 20 1 2 -1
media 40 1 3 -2
media 30 1 4 -3
boundary 4 9 9
end geom
read bounds all=ref end bounds
end model
end
=shell
cp newxnlib $RTNDIR/magnox44
end
```

Cross sections for nodal analysis

- T-DEPL (only) generates a database of cross sections any time branch calculations are requested.
  - Archive of cross sections and data for each reference case and branch, for each depletion step.
  - Saved in user-specified collapsed and homogenized format.
  - Other data (D, ADFs, pin form factors, neutron velocity, etc. is also saved)
- The database is a a simple binary file intended for later processing.

Cross sections for nodal analysis

- TRITON can produce pin-power edits for comparison to other results
- The TRANSFORMER utility is currently used to process this file and generate cross sections for PARCS.
- ORNL hopes to eventually extend TRANSFORMER to support other nodal codes
- The format of the binary file is available and other translators may be independently developed.

Cross sections for nodal analysis

```
=t-depl
Infinite lattice depletion model for a single pincell with branches.
44groupndf
read comp
uo2  1 den=10.045  1  841
   92234 0.022 92235 2.453 92236 0.011 92238 97.514   end
wtptzirc 4 6.44 4 40000 97.91 26000 0.5 50116 0.86 50120 0.73 1.0 620  end
h2o  5 den=0.7573  1  557 end
end comp
read celldata
latticecell squarepitch   pitch=1.4732  5 fuelr=0.47250 1 cladr=0.5588  4  end
end celldata
read depletion -1 end depletion
read burndata
power=21.220 burn=400  down=0     nlib=2 end
end burndata
read branch
define fuel 1 end
define mod 5 end
tf=841 tm=557 dm=.7573 sb=0.0 cr=0 end
tf=1000 end
tf=1000 tm=650 dm=.680 end
end branch
```

Cross sections for nodal analysis

```
read model
Infinite-lattice pin model (one-fourth)
read parm collapse=yes epsilon=5e-3 epseigen=1e-3 end parm
read materials
  1  1   !  3.0 enriched fuel! end
  4  1   !  clad ! end
  5  1   !  water ! end
end materials
read collapse 23r1 21r2 end collapse
read homog
500 homog_region 1 4 5 end
end homog
read adf 1 500 n=0.7366 w=0.7366 end adf
read geom
global unit 1
cylinder 10 .47815
cylinder 20 .55880
cuboid   30 0.7366 0.0 0.7366 0.0
media 1 1 10
media 4 1 20 -10
media 5 1 30 -20
boundary 30 3 3
end geom
read bounds all=refl end bounds
end model
end
```

Faster CENTRM Processing

- TRITON allows one to "mix and match" CENTRM cross section processing

- Two-region can be used globally, but more rigorous S_n solution can be applied when needed.

```
latticecell squarepitch pitch=1.44 41 fuelr=0.49 12 cladr=0.56 21 end

latticecell squarepitch pitch=1.44 42 fuelr=0.49 13 cladr=0.56 22 end

multiregion cylindrical right_bdy=white end
90 0.21913470 91 0.30990320 92 0.37955240 93 0.43826930
94 0.49000000 26 0.56000000 46 0.81243300 end zone
    centrmdata npxs=1 nfst=0 nthr=0 nmf6=0 end centrmdata
```

Reduced Cross Section Processing

- TRITON can be used to deplete every fuel pin in an assembly
  - Symmetry generally allows depletion of an assembly subset
- Each fuel pin is individually resonance-processed
  - Rigorous but slow
- An alternate approach has been developed for TRITON depletion to reduce this computational penalty.

Reduced Cross Section Processing

- Macroscopic cross sections for depleted materials are more sensitive to number densities that cross sections.
- It is a reasonable approximation to generate a single set of cross sections for similar rods by using an averaged set of isotopics.
- Local number densities in each rod are retained and combined with average cross sections.
- User determines which fuel mixtures are "similar."

Reduced Cross Section Processing

- Macroscopic cross sections for depleted materials are more sensitive to number densities that cross sections.
- It is a reasonable approximation to generate a single set of cross sections for similar rods by using an averaged set of isotopics.
- Local number densities in each rod are retained and combined with average cross sections.
- User determines which fuel mixtures are "similar."

Reduced Cross Section Processing

```
=t-depl
BWR-like depletion model with lattice physics calculations
xn44
read alias
 $fuel23 1,2 end
 $fuel36 3–6 end
 $2fuels 10,11 end
 $2clads 401–402 end
 $2mods 201–202 end
 $allfuels 1–6 end
end alias
read comp
' 2.3 w/o
u-234 10 0 4.7008e-06 900 end
u-235 10 0 5.2968e-04 900 end
u-238 10 0 2.2208e-02 900 end
o-16  10 0 4.5491e-02 900 end
' 3.6 w/o
u-234 11 0 7.5720e-06 900 end
u-235 11 0 8.2904e-04 900 end
u-238 11 0 2.1907e-02 900 end
o-16  11 0 4.5497e-02 900 end
zirc4  $2clads den=6.53 1 559 end
h2o $2mods  den=0.457 1.0000   559 end
end comp
```

Reduced Cross Section Processing

```
read celldata
latticecell squarepitch pitch=1.63 $2mods
                        fuelr=0.529 $2fuels cladr=0.615 $2clads end
end celldata
read depletion $fuel23 $fuel36 $fuelg end
               assign 10 $fuel23 end
               assign 11 $fuel36 end
end depletion
...
read model
BWR-like model with lattice physics calculations
read parm
  collapse=yes epsilon=1e-3 cmfd=yes xycmfd=3
end parm
read materials
 $allfuels   1  ! fuel pin ! end
 401  2  !Zirc4! end
 201  2  !H2O(void)! end
 399  2  !H2O(solid)! end
end materials
...
read array
ara=1 nux=3 nuy=3 pinpow=yes fill   1   2   3
                                    2   4   5
                                    3   5   6  end fill
end array
```

```
$fuel23 1,2 end
$fuel36 3–6 end
```

Congratulations! You've made it to the end of this tutorial!