

CADIS: Consistent Adjoint Driven Importance Sampling

SCALE 6.3.1

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

Learning goals

- Introduce the Consistent Adjoint Driven Importance Sampling (CADIS) variance reduction technique
 - Identify the input syntax and where additional input is required in the Monaco/MAVRIC input file to use the CADIS variance reduction technique
 - Identify the additional output files created when using the CADIS variance reduction technique
- Identify the MAVRIC utilities and how they can be used when performing shielding analysis
- In-class example: analyze dose rates near a cask

Variance reduction with CADIS



Variance reduction

- Changing the sampling routines to optimize the simulation to get more particles to do *something*
- Traditional methods:
 - Rely on knowing the expected results
 - Require trial and error
 - Multiple methods may work against each other
 - May require several iterations
 - Typically involve many steps
 - May require specialized Monte Carlo (MC) codes

Variance reduction

- What is desired:
 - Tell MC code what tally/tallies to calculate
 - The MC code figures out the best way to do that
- Importance map (weight windows)
 - If we know how “important” a given particle is as a function of $(\vec{r}, E, \vec{\Omega})$, then weight window target values can be assigned, using roulette and splitting to control particle weight
 - Importance is the solution to the adjoint equation, using the tally location as the spatial component of the adjoint source and the tally response function as the energy component of the adjoint source

A bit of review

- Forward transport equation

$$H\Phi(\vec{r}, E, \hat{\Omega}) = Q(\vec{r}, E, \hat{\Omega})$$

$$H = \hat{\Omega} \cdot \vec{\nabla} + \Sigma(\vec{r}, E) - \int dE' \int d\hat{\Omega}' \Sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \rightarrow \hat{\Omega})$$

- Solve for flux, then compute detector response

$$R = \int_{V_d} \int_E \sigma_d(\vec{r}, E) \phi(\vec{r}, E) dE dV$$

- Adjoint transport equation

$$H^+\Phi^+(\vec{r}, E, \hat{\Omega}) = Q^+(\vec{r}, E, \hat{\Omega})$$

$$H^+ = -\hat{\Omega} \cdot \vec{\nabla} + \Sigma(\vec{r}, E) - \int dE' \int d\hat{\Omega}' \Sigma_s(\vec{r}, E \rightarrow E', \hat{\Omega} \rightarrow \hat{\Omega}')$$

- Solve for adjoint flux using $q^+(\vec{r}, E) = \sigma_d(\vec{r}, E)$, then compute

$$R = \int_{V_s} \int_E q(\vec{r}, E) \phi^+(\vec{r}, E) dE dV$$

Where;

H : transport operator

Φ : scalar flux

$(\vec{r}, E, \hat{\Omega})$: spatial, energy, and angular components

$\sigma_d(\vec{r}, E)$: detector response

$q(\vec{r}, E)$: forward source

$q^+(\vec{r}, E)$: adjoint source

$\phi(\vec{r}, E)$: forward flux

$\phi^+(\vec{r}, E)$: adjoint flux

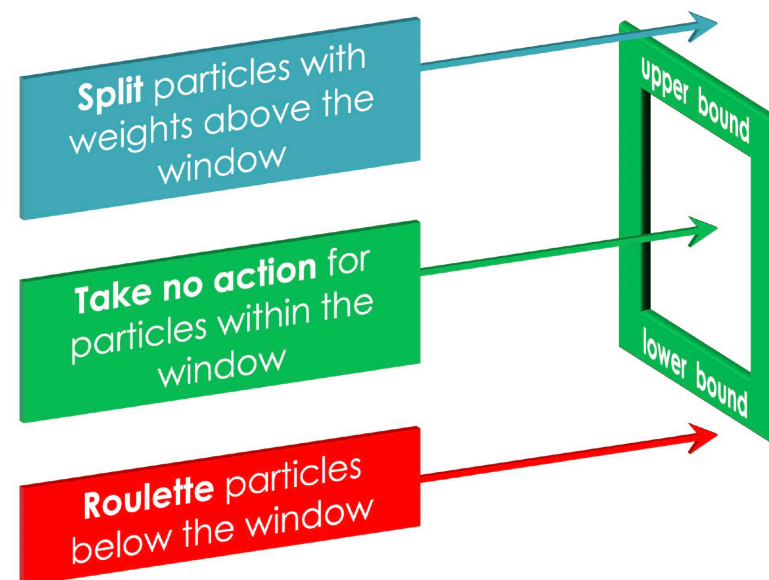
V_d : detector volume

V_s : source volume

Using importances

- If the true importances are known, then there is no need for the MC calculation
- If the approximate importances are known, then weight windows can be used for a MC game:

$$\bar{w}(\vec{r}, E) = \frac{c}{\phi^+(\vec{r}, E)}$$



CADIS methodology

Consistent Adjoint Driven Importance Sampling

- Biased source and importance map work together

Ali Haghghat and John C. Wagner, "Monte Carlo Variance Reduction with Deterministic Importance Functions," Progress in Nuclear Energy, 42(1), 25–53, (2003).

- Solve the adjoint problem using the detector response function as the adjoint source

$$q^+(\vec{r}, E) = \sigma_d(\vec{r}, E)$$

- Weight window targets are inversely proportional to the adjoint flux (i.e., measure of importance of the particles to the response)

$$\bar{w}(\vec{r}, E) = \frac{c}{\phi^+(\vec{r}, E)}$$

CADIS methodology (cont.)

- We want source particles born with a weight matching the weight window targets

$$w_0(\vec{r}, E) = \frac{q(\vec{r}, E)}{\hat{q}(\vec{r}, E)} = \bar{w}(\vec{r}, E)$$

- So, the biased source needs to be

$$\hat{q}(\vec{r}, E) = \frac{q(\vec{r}, E)}{\bar{w}(\vec{r}, E)} = \frac{1}{c} q(\vec{r}, E) \phi^+(\vec{r}, E)$$

- Since the biased source is a probability density function (pdf), solve for c

$$c = \int_{V_S} \int_E q(\vec{r}, E) \phi^+(\vec{r}, E) dE dV$$

Which is the
estimate of
response: R

CADIS methodology – Summary

- Define the adjoint source

$$q^+(\vec{r}, E) = \sigma_d(\vec{r}, E)$$

- Solve for the adjoint flux $\phi^+(\vec{r}, E)$
- Estimate the response R

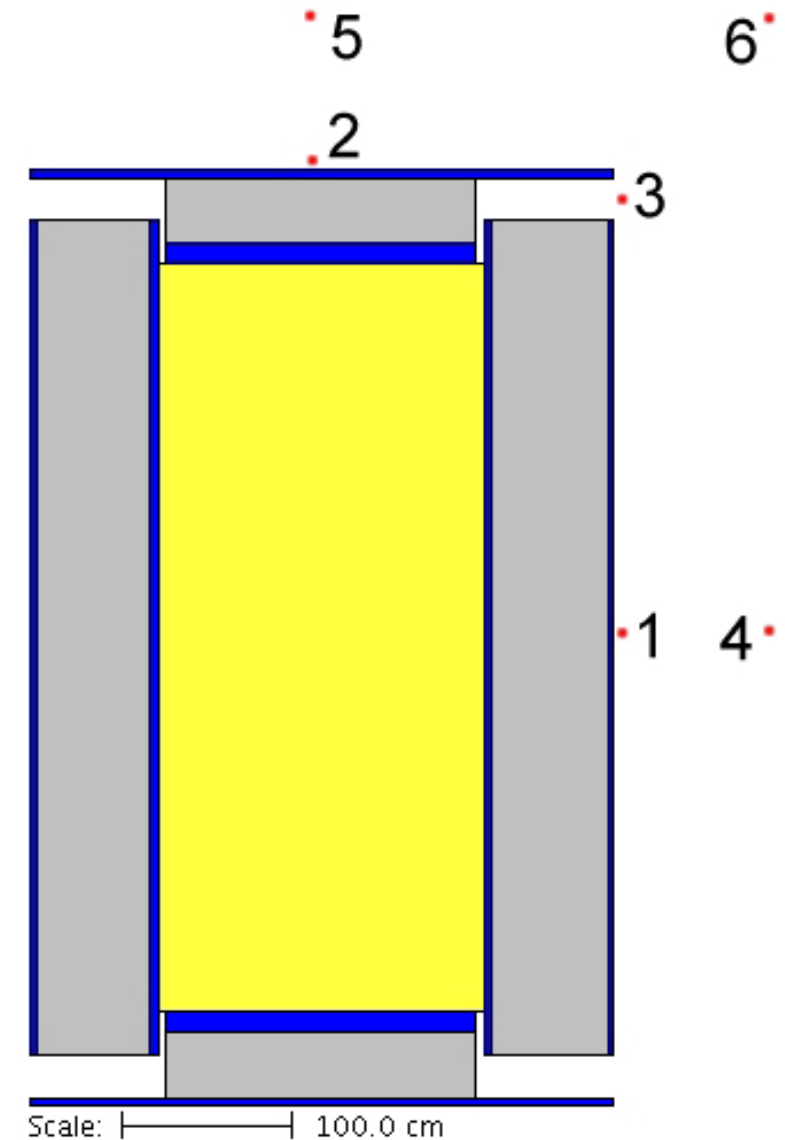
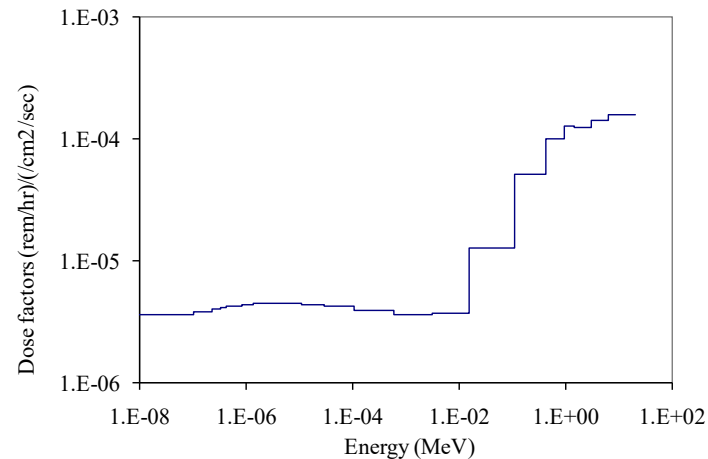
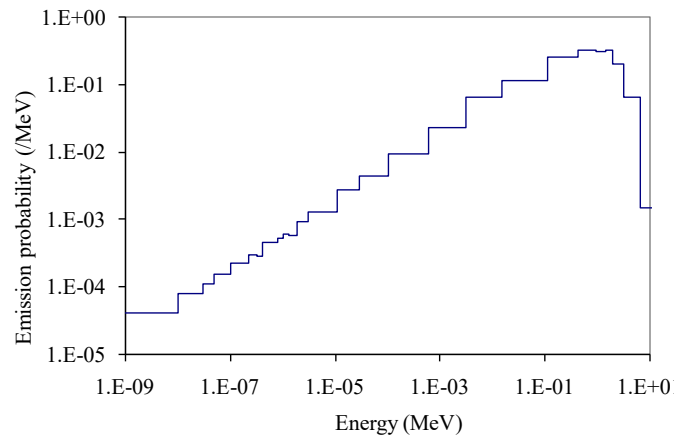
$$R = \int_{V_S} \int_E q(\vec{r}, E) \phi^+(\vec{r}, E) dE dV$$

- Construct weight windows and biased source

$$\bar{w}(\vec{r}, E) = \frac{R}{\phi^+(\vec{r}, E)} \quad \hat{q}(\vec{r}, E) = \frac{1}{R} q(\vec{r}, E) \phi^+(\vec{r}, E)$$

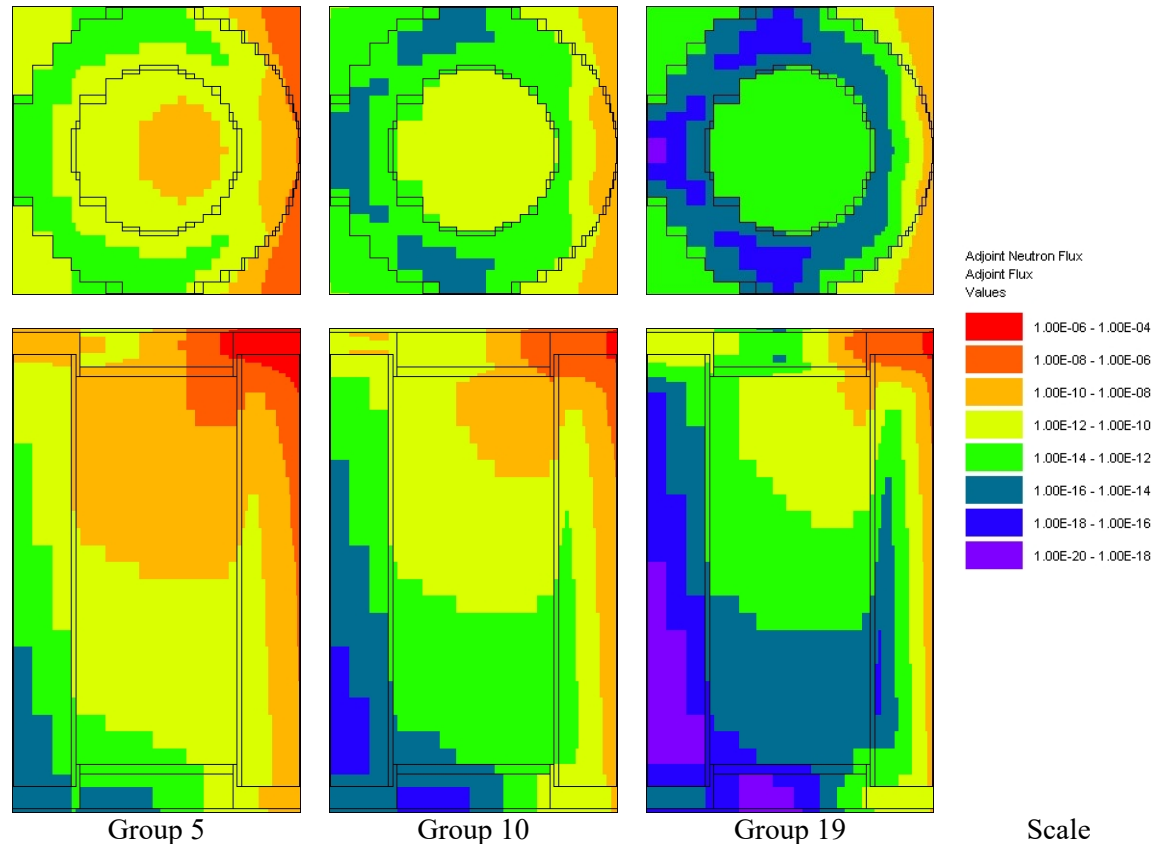
Example application – spent fuel cask

Calculate dose rate at six different detector locations



Example application – spent fuel cask (cont.)

- For detector 3: compute adjoint fluxes

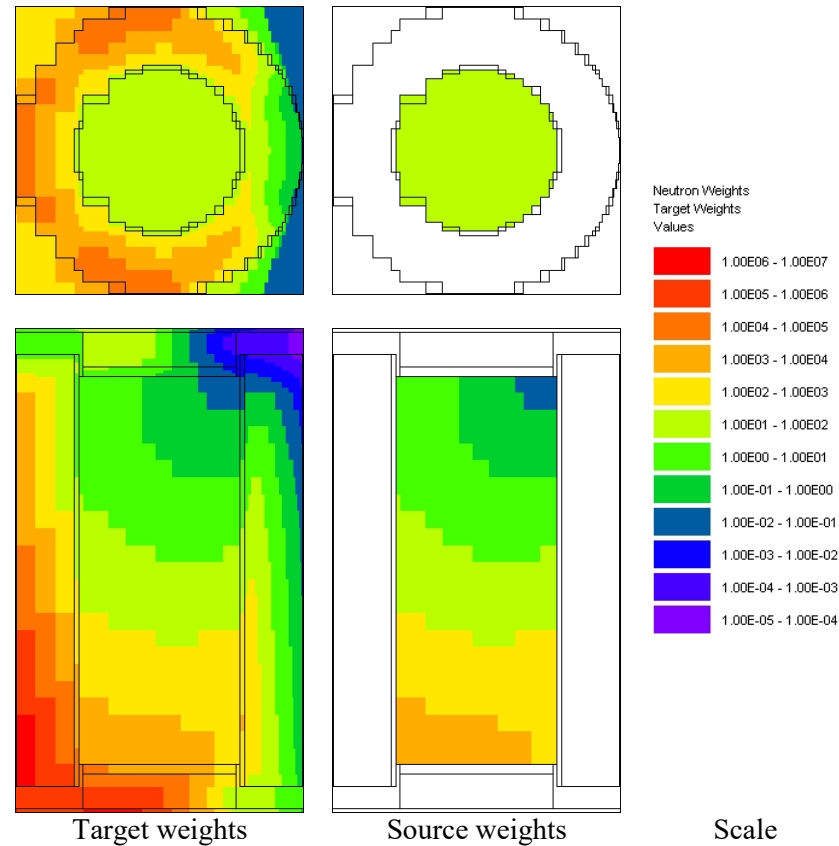


Horizontal cross-sectional views at the cask midplane showing adjoint neutron flux solution

Adjoint neutron fluxes ($\text{n/cm}^2/\text{s}$) calculated by Denovo for groups 5 (0.9–1.4 MeV), 10 (0.58–3.0 keV), and 19 (0.8–1 eV)

Example application – spent fuel cask (cont.)

- Compute target weights, biased source weights

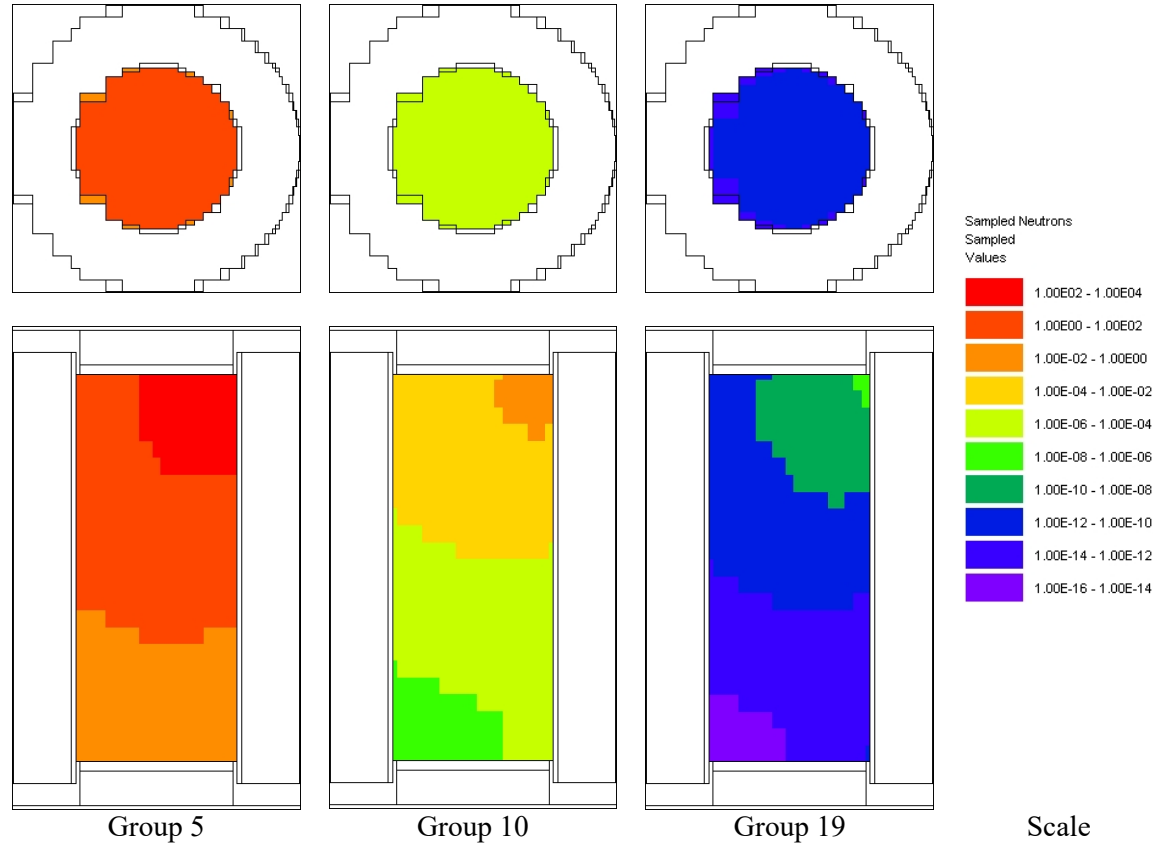


Horizontal cross-sectional views at the cask midplane showing neutron weights

Neutron target weights from the importance map and source weights (at birth) for neutron group 5 (0.9 to 1.4 MeV)

Example application – spent fuel cask (cont.)

- Biased source



Biased source sampling probability (neutrons/cm³) for neutron groups 5 (0.9–1.4 MeV), 10 (0.58–3.0 keV), and 19 (0.8–1 eV)

Example application – spent fuel cask (cont.)

- Computational speed up is significant (Six separate calculations)

Results

detector	Analog Monaco 5164 min	MAVRIC 543 min
1	5.54E-04 ± 24.0%	7.07E-04 ± 0.7%
2	6.97E-03 ± 5.9%	7.77E-03 ± 0.5%
3	1.55E-02 ± 2.0%	1.54E-02 ± 0.4%
4	4.57E-04 ± 4.7%	4.27E-04 ± 0.7%
5	1.36E-02 ± 0.9%	1.35E-02 ± 0.2%
6	2.91E-03 ± 1.2%	2.91E-03 ± 0.2%

$$FOM = \frac{1}{T\sigma^2}$$

Speed up

detector	Monaco	MAVRIC
1	1	11179
2	1	1324
3	1	238
4	1	429
5	1	193
6	1	342

CADIS summary

- Provides consistent relationships for calculating source and transport biasing parameters
- Eliminates the incompatibility between source and transport biasing that has been problematic in other approaches
- Large speedup for source/detector problems
- Described in more detail in:
 - John Wagner, Alireza Haghighat, “Automated Variance Reduction of Monte Carlo Shielding Calculations Using the Discrete Ordinates Adjoint Function,” *Nuclear Science & Engineering*, vol. 128, p.p. 186–208, 1998
 - John Wagner, Alireza Haghighat, “Monte Carlo Variance Reduction with Deterministic Importance Functions,” *Progress of Nuclear Energy*, vol. 42(1), 2003

MAVRIC input using importance maps



MAVRIC input: block format

```
=mavric      parm=keyword  
Some title for this problem  
v7.1-28n19g
```

```
read compositon  
...  
end composition  
read celldata  
...  
end celldata
```

```
read geometry  
...  
end geometry  
read array  
...  
end array  
read volume  
...  
end volume  
read plot  
...  
end plot
```

```
read definitions  
...  
end definitions  
read sources  
...  
end sources  
read tallies  
...  
end tallies  
  
read parameters  
...  
end parameters  
read biasing  
...  
end biasing  
read importanceMap  
...  
end importanceMap  
  
end data  
end
```

Cross section libraries used with Denovo and Monaco

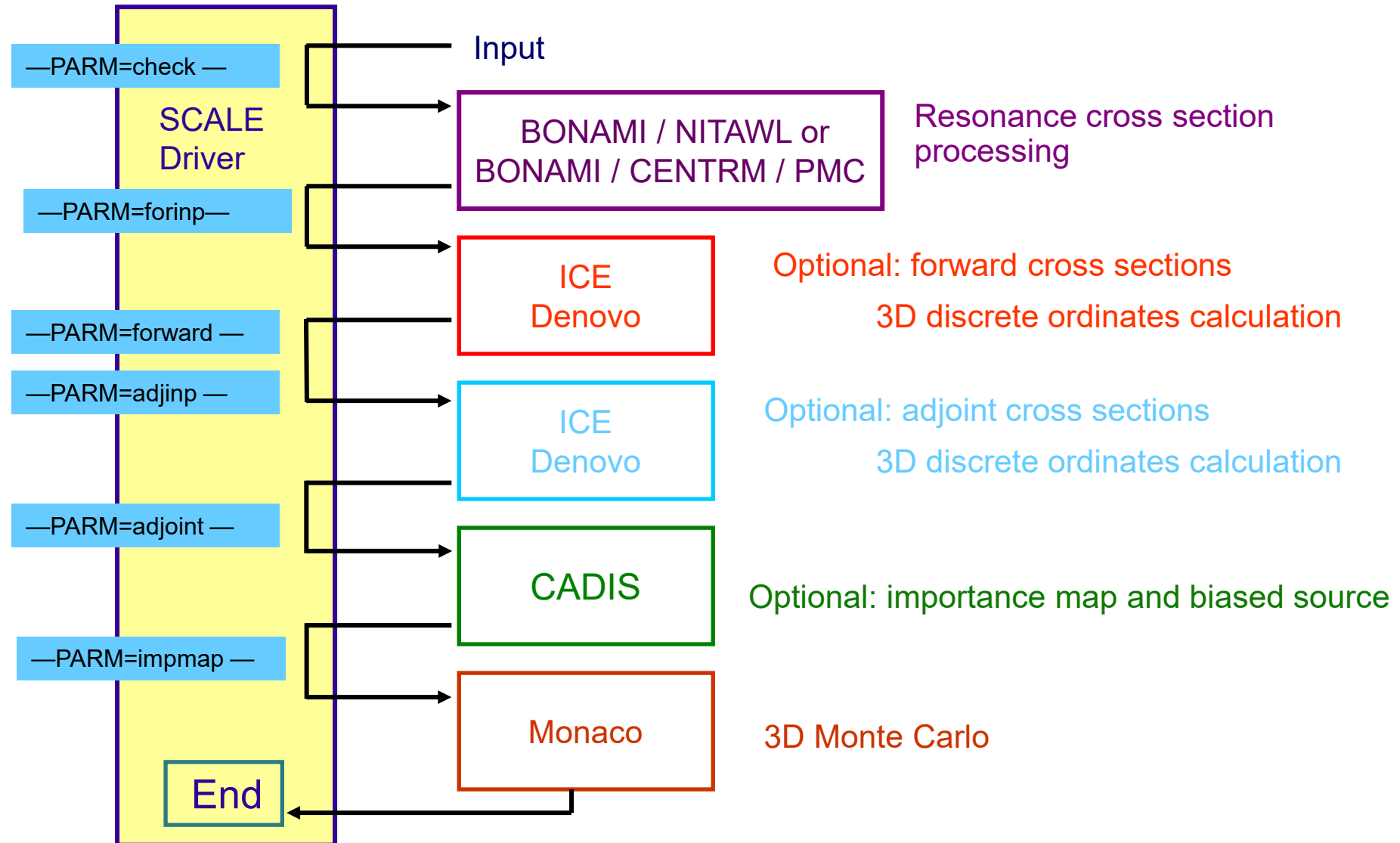
- It is fastest to use different cross section libraries for the **Denovo** importance map (adjoint) calculation and the **Monaco** Monte Carlo calculation
 - Use a coarse group library for Denovo, *v7.1-28n19g*
 - Use a fine group library for Monaco, *v7.1-200n47g*
 - Use a continuous-energy library for Monaco, *ce_v7.1_endf.xml*

```
=mavric      parm=keyword
Some title for this problem
v7.1-28n19g      ←Denovo library

...

read parameters
...
library="v7.1-200n47g"      ←Monaco MG library
OR
ceLibrary="ce_v7.1_endf.xml" ←Monaco CE library
...
end parameters
```

MAVRIC sequence workflow



PARM=keyword

- PARM=CHECK
 - Checks over input, looks for simple mistakes
- PARM=FORINP
 - Produces forward Denovo input: `i_xkba`
- PARM=FORWARD
 - Runs forward Denovo, produces forward flux file: `outputName.forward.dff`
- PARM=ADJINP
 - Produces adjoint Denovo input: `i_xkba`
- PARM=ADJOINT
 - Runs adjoint Denovo, produces adjoint flux file: `outputName.adjoint.dff`
- PARM=IMPMAP
 - Creates biased source and importance map, produces
 - *.**mim** – Monaco importance map
 - *.**msm** – Monaco source mesh

MAVRIC simulations using CADIS

- User must provide:
 - All the information required for an analog calculation
 - Materials
 - Geometry
 - Source
 - Tallies
 - Parameters
 - Plus,
 - a grid geometry for the discrete ordinates
 - a description of the adjoint source (based on the tally you wish to optimize)
- Everything else is automatic!

Files generated with CADIS

- Mesh importance map
 - *outputName.mim*
- Mesh source map (biased source)
 - *outputName.msm*
- Or
 - *outputName.sid.msm* for each source
- For multiple sources, each source is biased, and the biased strength of each source is computed

MAVRIC utilities

mt2ascii	Convert a mesh tally into an ASCII text file.
mt2msl	Convert a mesh tally into a mesh source lite.
mt2msm	Convert a mesh tally into a mesh source.
mt2silo	Convert a mesh tally file into a Silo file for VisIt.
mt2vtk	Convert one dataset of one family in a mesh tally to VTK format.
mtAdder	Add several Monaco mesh tally files together into one mesh tally.
mtAverager	Average several Monaco mesh tally files into one mesh tally.
mtBinOp	Binary operation of mesh tally files: sum, difference, product and ratio.
mtDisp	Display the basics of a mesh tally file.
mtExpand	Expand a space-only mesh from a mesh tally with an energy function.
mtFilter	Perform various filters on a mesh tally file.
mtInv	Invert all of the values in a mesh tally.
mtMask	Keep only or remove specified voxels of a mesh tally based on geometry.
mtMinMax	Find the location/value of the min or max of each realmesh in a mesh tally.
mtMultiply	Multiply a mesh tally by a constant factor.
mtPull	Pull values from certain voxels out of a mesh tally file.
mtRefine	Subdivide the mesh into smaller meshes for better visualization.
mtResp	Apply a response function to one family of a mesh tally file.
mtSplit	Split off part of a mesh tally file into a separate mesh tally file.

MAVRIC utilities (cont.)

mtBinOp - Binary operation of mesh tally files: sum, difference, product and ratio.

Intended use: Apply simple math to the results stored in mesh tally files.

Input: The first mesh tally, the operator: add (or sum, +), subtract (or difference, -), multiply (or product, x, *) and divide (or ratio, ÷, " /"), the second mesh tally name and then name of the resulting mesh tally file.

Output: A new mesh tally file.

Notes: Uncertainties are propagated assuming the two mesh tallies are uncorrelated, which may not always be a good assumption. Mesh tallies need to have the same grid structure and number of families and groups. Dataset names in the results are inherited from the first mesh tally and may not make sense after the operation. When using the / (slash) for division, enclose it in quotes ("/").

Example:

```
=mtBinOp
"neutron.3dmap"      ! first operand
divide              ! operation
"total.3dmap"        ! second operand
"ratio.3dmap"        ! output file name
end
```

MAVRIC utilities (cont.)

dff2dso	Convert a Denovo flux file into a Denovo spatial output file.
dff2mai	Convert a Denovo flux file into a mesh angular information file.
dff2mim	Invert a Denovo flux file and store as a mesh importance map.
dff2msl	Convert a Denovo flux file into a mesh source lite.
dffBinOp	Binary operation of Denovo flux files: sum, difference, product and ratio.
dffDisp	Display the basics of a Denovo flux file.
dffExpand	Expand a space-only Denovo flux file by an energy function.
dffFilter	Perform various filters on a Denovo flux file.
dffFix	Fix the zero and negative values in a Denovo flux file.
dffInt	Integrate a single particle type from a Denovo flux file.
dffInv	Invert the values in a Denovo flux file.
dffMult	Multiply a Denovo flux file by a constant factor.
dffPull	Pull fluxes from certain voxels out of a Denovo flux file.
dffResp	Apply a response function to scalar fluxes in a Denovo flux file.
dffSplit	Split off a single particle type from a Denovo flux file.

MAVRIC utilities (cont.)

dsi2asc	Convert a Denovo simple input (*.dsi) from binary to ascii.
dsiDisp	Display the basics of a Denovo simple input file.
dso2msl	Use a Denovo spatial output to create a mesh source lite.
dsoDisp	Display the basics of a Denovo spatial output file.
mim2wwinp	Convert a mesh importance map into an MCNP weight window file.
mimDisp	Display the basics of a mesh importance map (*.mim) file.
mimNorm	Normalize a mesh importance map to a given location/energy.
msmDisp	Display the basics of a mesh source map (*.msm) file.

MAVRIC utilities (cont.)

```
=mtAverager
```

```
3
```

```
'c:\documents\meshFilename_1.3dmap'      800000
```

```
'c:\documents\meshFilename_2.3dmap'      900000
```

```
'c:\documents\meshFilename_3.3dmap'      800000
```

```
'c:\documents\meshFilename.ave.3dmap'
```

```
end
```

```
=mtSplit
```

```
'c:\documents\meshFilename.mt1.3dmap'    ! the mesh tally
```

```
2                                          ! second family - responses
```

```
1                                          ! first response - photon dose rate
```

```
'photonDoseRate.3dmap'                  ! output file name
```

```
end
```

```
=mtMultiply
```

```
'photonDoseRate.3dmap'                  ! input file name
```

```
1000.0                                  ! convert rem/hr to mrem/hr
```

```
'c:\documents\mremPhoton.3dmap'         ! output file name
```

```
end
```

In-class example: dose rates near a storage cask

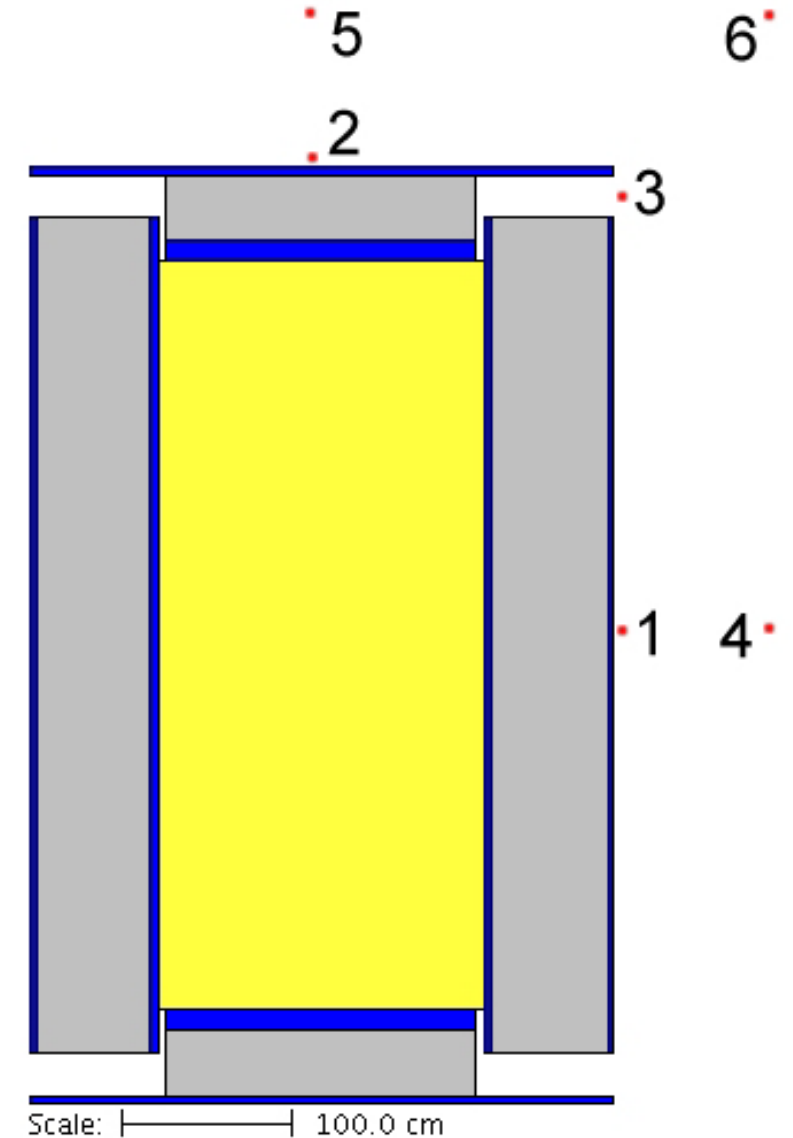
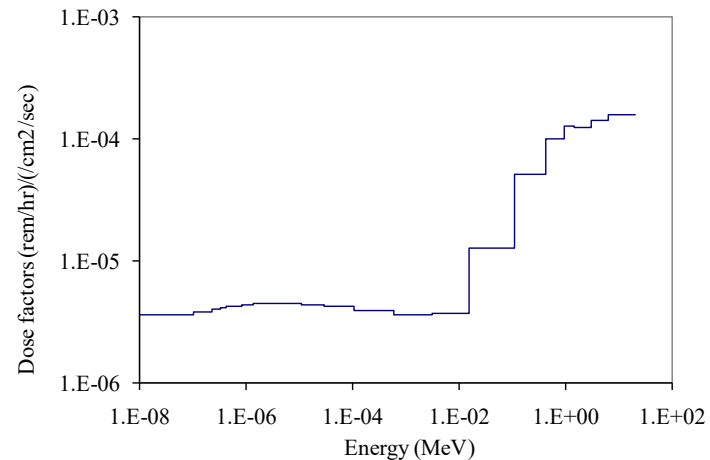
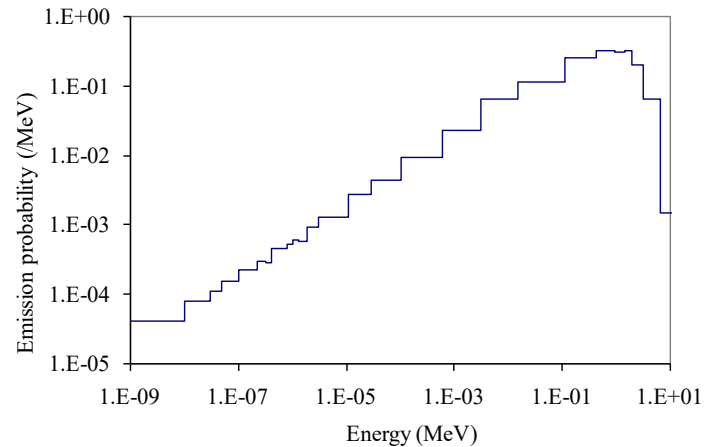


Learning goals

- The students and instructor will work through an example problem together, which will allow them to apply what they have learned about running Monaco/MAVRIC using the CADIS variance reduction technique

Spent Fuel Shipping Cask Example in the [MAVRIC manual](#)

Calculate dose rate at six different detector locations

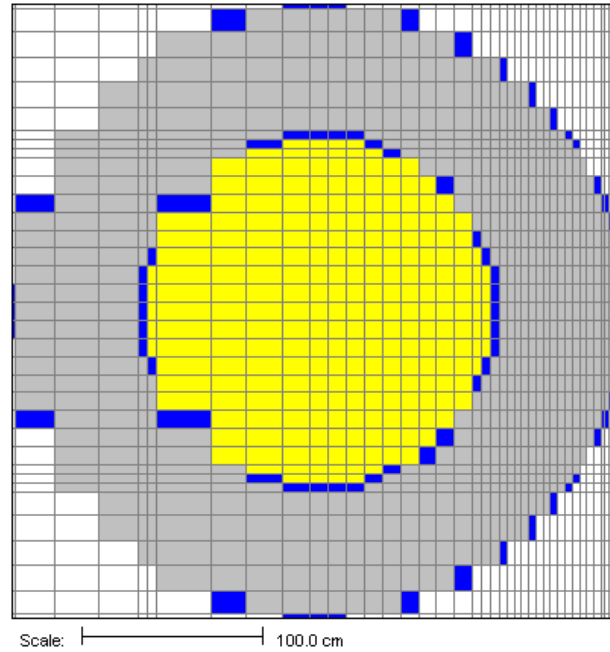
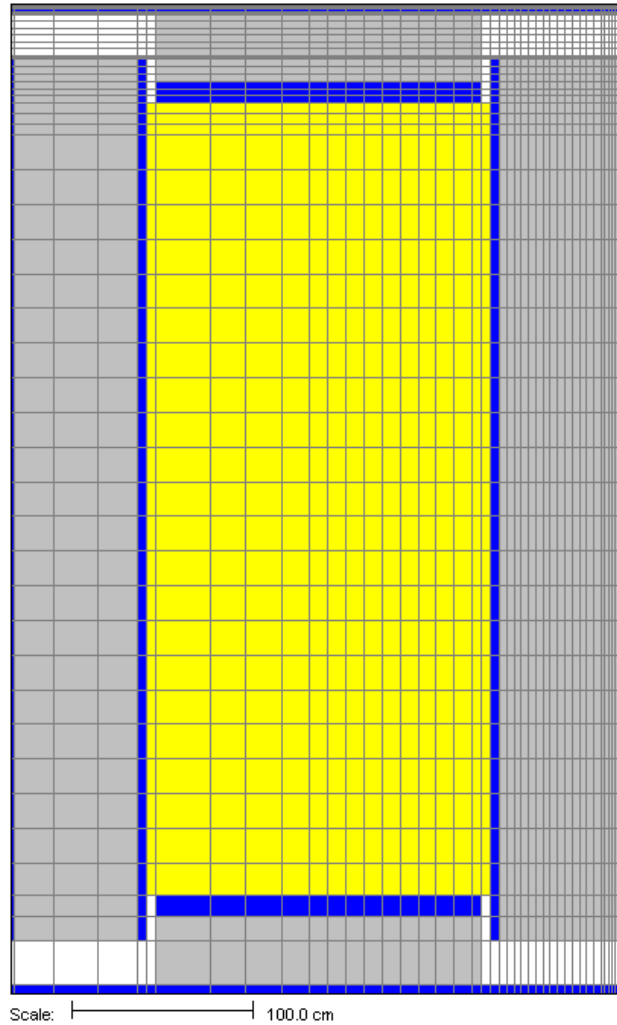


In-Class Example Problem

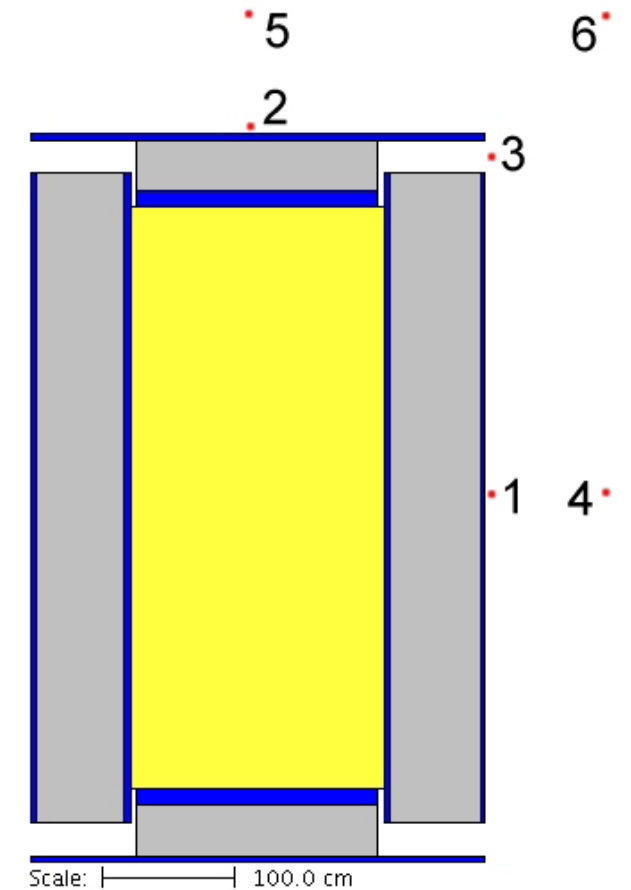
- Try a cask problem similar to the example shown in the [MAVRIC manual](#)
- Start with *starters/caskNanalog.inp*
- Use a coarse mesh (to speed things up)

<i>x (47)</i>				<i>y (36)</i>			<i>z (50)</i>			
-170	15	108	156	-170	-55	65	-285.6	-50	210	264
-168	25	112	158	-168	-45	75	-280.6	-30	216.2	267.9
-146	35	116	160	-155	-35	85	-255.2	-10	222.4	271.8
-122	45	120	162	-141	-25	90	-240.6	10	228.6	275.7
-100	55	124	164	-127	-15	95	-228.6	30	232.6	279.6
-95	65	128	165	-113	-5	100	-210	50	236.6	280.6
-90	75	132	166	-100	5	113	-190	70	240.6	281.6
-60	85	136	167	-95	15	127	-170	90	245.1	282.6
-40	90	140	168	-90	25	141	-150	110	249.7	283.6
-20	95	144	169	-85	35	155	-130	130	254.2	284.6
-5	100	148	170	-75	45	168	-110	150	255.2	285.6
5	104	152		-65	55	170	-90	170	256.2	
							-70	190	260.1	

Coarse mesh for Denovo



Mesh specifically tailored for detector positions 3 and 6



Summary of the CADIS method in MAVRIC

- Minimal changes are needed to convert an analog MAVRIC to use the CADIS method
 - Add “importanceMap” block with parameters for Denovo
 - Use a new/existing “gridGeometry” for Denovo
- Adjoint flux solutions are obtained from Denovo and an importance map is created automatically
- If needed, available MAVRIC utilities can be utilized for further modifications to Denovo solutions