

Statistical Uncertainty Analysis with SAMPLER

SCALE 6.3

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



U.S. DEPARTMENT OF
ENERGY

Learning goals

- Understand the stochastic sampling approach to uncertainty quantification in Sampler
- Learn how to propagate model input parameter uncertainties and nuclear data uncertainties through a SCALE sequence
- Explore available analysis capabilities

Stochastic Sampling



Sampler is a stochastic sampling sequence for statistical uncertainty analysis

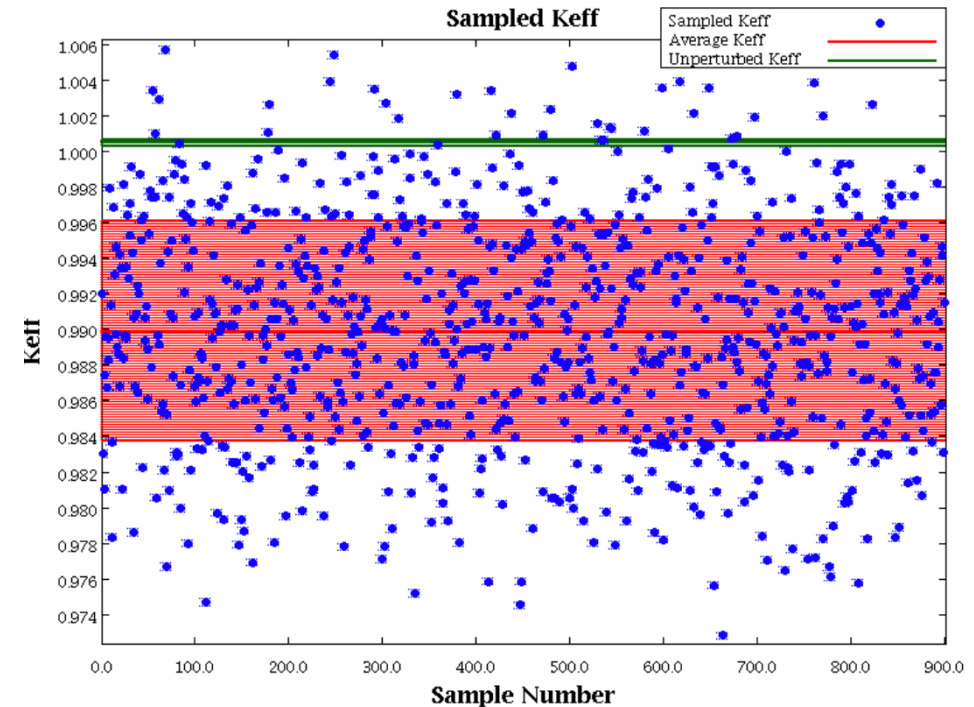
- Sampler is a versatile uncertainty quantification tool that can be applied in combination with any SCALE sequence
- **Sampler calculates the uncertainty in *any* result of a SCALE calculation by propagating two basic types of input uncertainty through the calculation:**

Model input parameters

Compositions (material density, enrichment, ...)
Geometry (fuel radius, cladding thickness, ...)
And any other value that is specified in as input parameter

Nuclear data

Multi-group cross sections
Fission product yields
Decay data
Kinetics data (betas, lambdas)

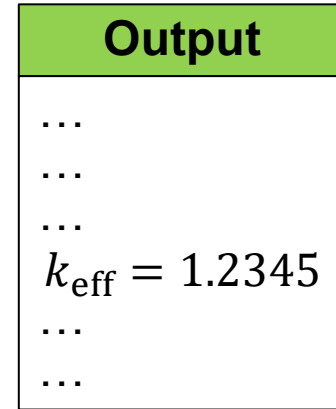
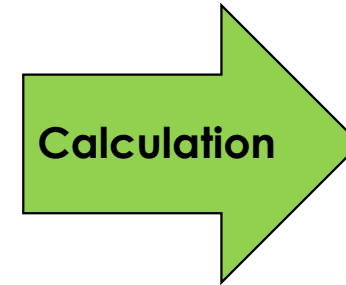
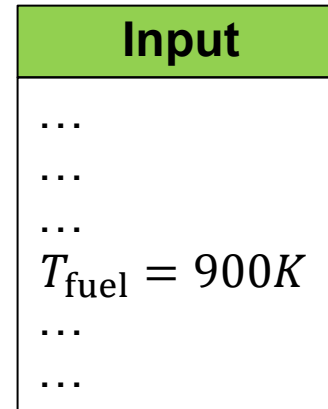


SCALE vision: Complete uncertainty quantification in all SCALE simulations

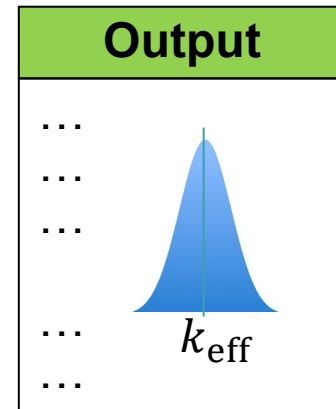
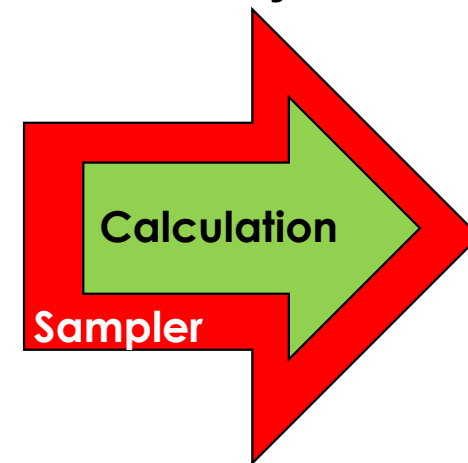
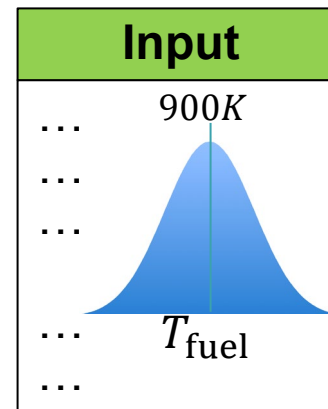
“**Complete**” means that every **value (number)** for an input parameter in the model is associated with an **uncertainty** that is propagated to the uncertainty in calculated results

- Known as the best-estimate plus uncertainty (BEPU) methodology
- Must include
 - **every input parameter**
cross sections, clad thickness, fuel temperature, decay constant, compositions, etc.
 - **every SCALE sequence**
TRITON, Polaris, ORIGEN, MAVRIC, etc.
- Must capture non-linearity, $\Delta f \approx \frac{\partial f}{\partial x} \Delta x$, that may be introduced in time-dependent problems and that may not be resolved with first-order perturbation theory methods (such as SCALE/TSUNAMI)

Best-estimate model



Uncertainty model



Users control the input parameter uncertainties, but not the nuclear data uncertainties

Input parameter uncertainties

- Users control input parameter uncertainties
- Sampler permits specifications:
 - Standard deviation
 - Distribution (normal, linear, beta)
 - Minimum/maximum values
- Sampler permits dependance between multiple perturbed input parameters
- Based on user-provided specification, Sampler creates perturbation of input parameters

Nuclear data uncertainties

- Users do **NOT** control nuclear data uncertainties
- Nuclear data uncertainties (covariance data) are taken from evaluated libraries and other resources (see next slide)
- Generation of nuclear data perturbations is non-trivial:
 - *What are the appropriate distributions? Normal? Log-normal?*
 - *How can we consider cross-correlations between two reactions, potentially between two different nuclides?*
 - *What happens if there is a gap in uncertainty information?*
 - *What can we do if the covariance data is not mathematically correct (e.g., covariance matrices not positive semidefinite)?*
- Perturbed nuclear data libraries were pre-generated and are provided in SCALE's data directory ("perturb" directory)
- Users generally use those provided perturbed libraries

SCALE 6.3 does not provide tools for users to generate nuclear data perturbations

There are several components of nuclear data uncertainties

Cross sections (XS) covariance data in SCALE:

ENDF/B-VII.1 or ENDF/B-VIII.0, supplemented by other sources (low-fidelity data, JENDL, ...)

Fission product (FP) yields covariance data in SCALE

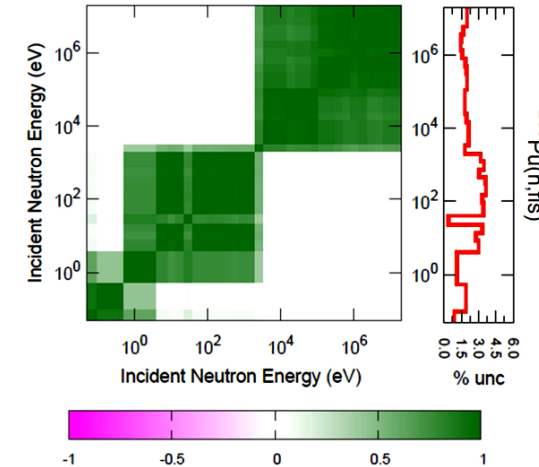
ENDF/B-VII.1 standard deviations, with correlations generated by combining independent and cumulative fission yields

Decay data covariance data in SCALE

ENDF-VII.1, modified to include branching correlations due to the constraint that branch sum=1.0

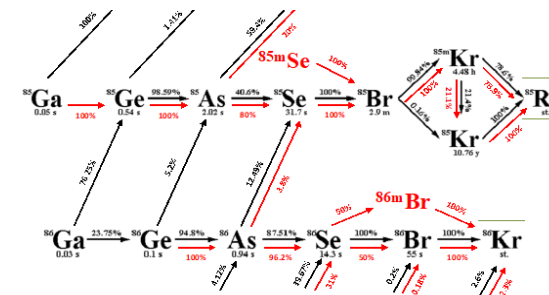
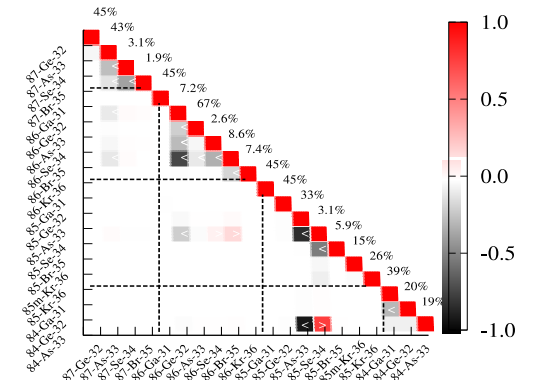
Kinetics data uncertainties in SCALE

Delayed neutron fractions (beta) and decay constants (lambda) based on R. J. Tuttle*



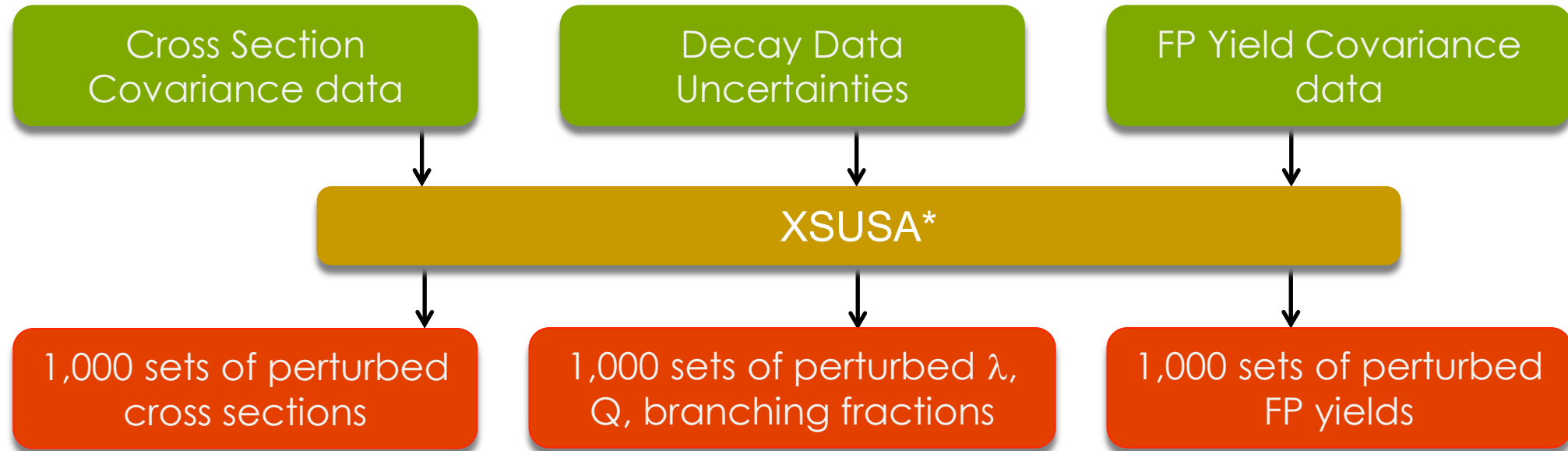
^{239}Pu fission covariance matrix

^{239}Pu fission yield correlation matrix



Mass A=85-87 yields

Generation of provided perturbed nuclear data libraries



- Nuclear data perturbations were pre-generated with XSUSA*
- Several assumptions were made during the generation of these perturbations:
 - Normal distribution
 - Truncation of normal distribution to avoid negative (unphysical) data
 - Mathematic problems with covariance matrices are resolved by adjusting covariance matrices
- 1,000 sets of perturbed data are available in the SCALE data directory, sub-directory “perturb”

Critical perspective on nuclear data uncertainties

- When examining uncertainties in results due to nuclear data uncertainties, engineering judgement and experience must be considered:
 - We have been predicting a 1σ uncertainty in k_{eff} of ~ 500 pcm in LWRs for years
 - However, validation data shows that we can predict k_{eff} for these reactors more accurately than within 500 pcm
 - Therefore, we conclude that the impact of nuclear data uncertainties is likely overestimated
- Propagated nuclear data uncertainty is useful to compare systems
 - For example, the 1σ uncertainty in k_{eff} in a thermal UO_2 system is ~ 500 pcm, whereas it is ~ 850 pcm in a fast Pu metal system
 - This indicates that we have better knowledge of cross sections affecting criticality in systems that use UO_2 fuel and that have a thermal neutron spectrum
- The covariance data provided in nuclear data libraries has gaps and sometimes differs significantly between different library versions
- The cross sections provided in the ENDF/B libraries are adjusted to many critical experiments; the corresponding covariance data was not adjusted to any application

Sampler performs three steps

1. Generation of perturbed SCALE inputs

- Input files contain perturbed model input parameters (perturbed geometry, material composition, etc.)
- Nuclear data perturbations are enabled either by linking directly to the pre-generated perturbed library, or by applying pre-generated perturbation factors to the nominal nuclear data library

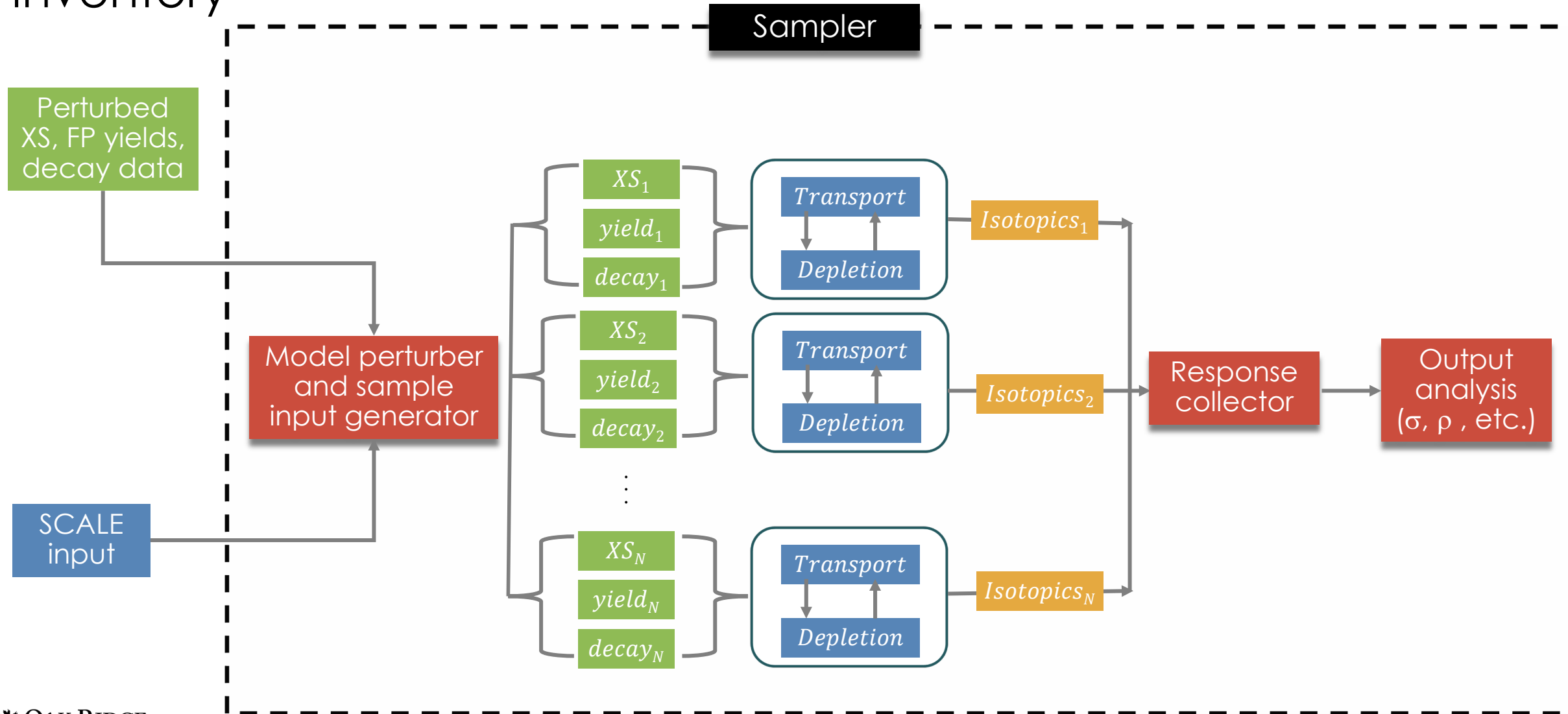
2. Management of SCALE input execution

- Execution of all individual SCALE runs by calling *scalerte* on each SCALE input
- Threading permits execution of multiple inputs in parallel
- (Users can also manually run all inputs, for example on a computing cluster)

3. Analysis of results

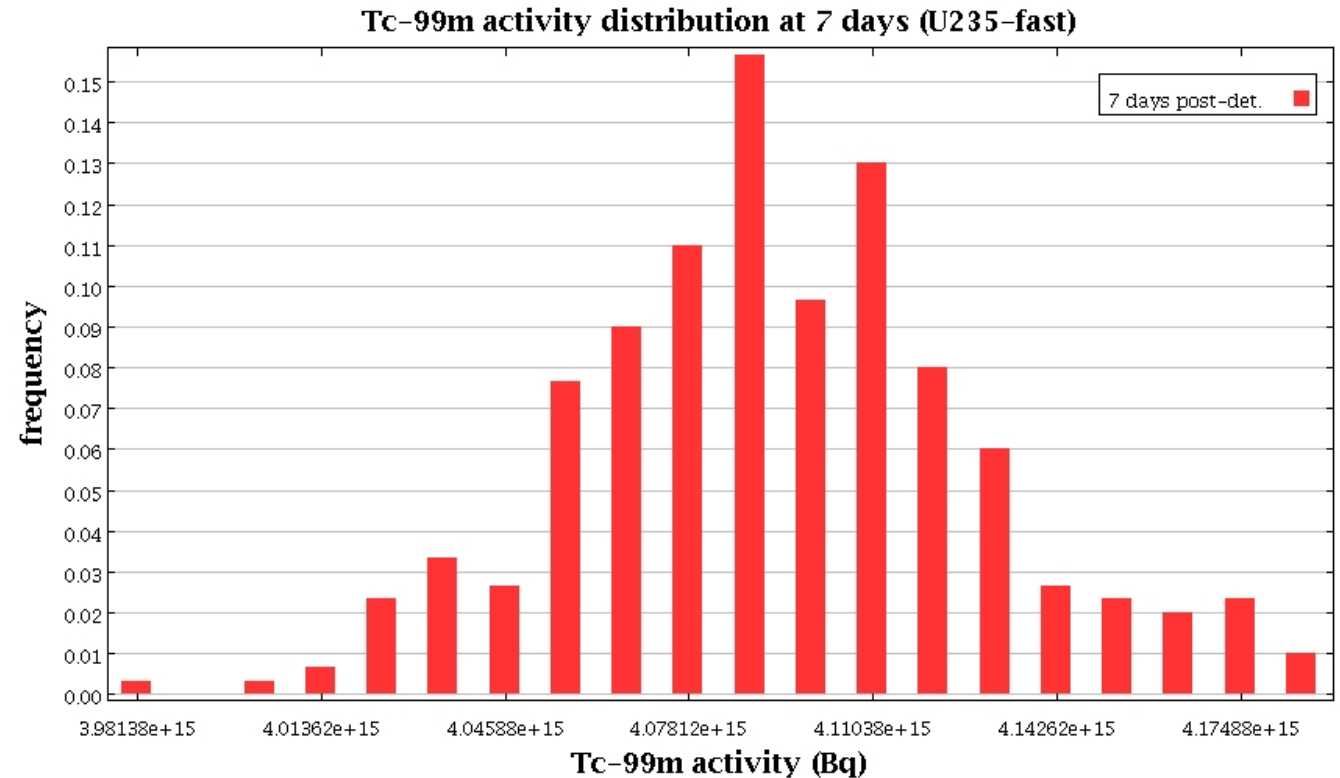
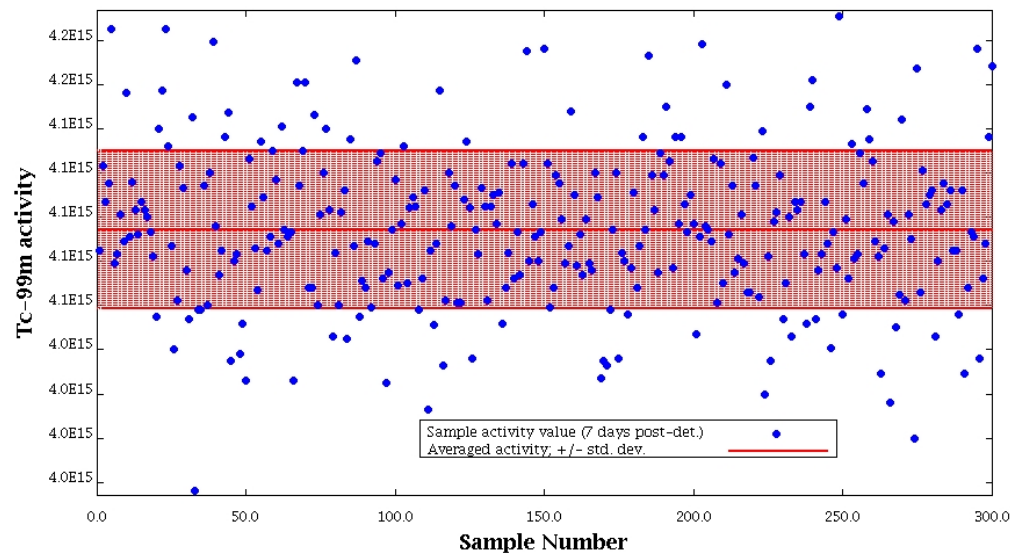
- Extraction of requested output responses (k_{eff} , nuclide concentrations, etc.)
- Calculation of standard deviation, correlation factors, etc.

Sampler's uncertainty propagation through a coupled neutron-transport depletion calculation to obtain uncertainties in nuclide inventory



Sampler provides several plots and statistics for uncertainty quantification of requested responses

Example: Impact of FP yield and decay data uncertainties for ^{235}U fission burst



Sampler's analysis capabilities

- Sampler automatically calculates mean values and standard deviations for requested output responses
- Users can request the following additional values to be calculated:
 - Pearson correlation coefficient
 - Covariances between output responses
 - Correlation matrix between output responses
 - Covariance matrix between output responses
 - R^2 sensitivity index (see next slide)

Sampler's analysis capabilities – R^2 sensitivity index

- R^2 is a sensitivity index (more accurately: squared multiple correlation coefficient) that is used in Sampler calculations that propagate cross section uncertainties to output responses
- R^2 captures the contribution to the uncertainty in the output response of interest that is caused by the uncertainties of a given nuclide reaction, including the fraction of uncertainties that the given nuclide reaction has in common with other nuclide reactions due to interdependencies (correlations)
- In easier words: R^2 answers the question “What are the most important contributors to the observed uncertainty in the response and what is their ranking?”
- Characteristics of R^2 :
 - $0 \leq R^2 \leq 1$
 - $R^2 = 0$, if the response is not correlated to any of the cross sections of a nuclide reaction
 - $R^2 = 1$, if the response is a linear function of only the cross sections of the given nuclide reaction
 - R^2 only captures linear effects
- R^2 is implemented for the analysis of *cross section* uncertainties (not yet fission yields, decay data, or kinetics data)

Sampler Input



Sampler input blocks

```
=sampler
  read parameters
  ...
end parameters
  read variable [myVariable]
  ...
end variable
  read response [myResponse]
  ...
end response
  read analysis [myAnalysis]
  ...
end analysis
  read case [myCase]*
    sequence=t-depl/origen/opus
    ...
  end sequence
end case
end
```

Sampler parameters: what to perturb, how many samples, libraries etc.

Variable: a variable in the SCALE input that is allowed to be modified by Sampler (e.g., fuel pin radius)

Response: quantity of interest for statistical analysis (e.g., k_{eff})

Analysis: type of analysis performed on defined responses** (e.g., *correlation coefficient between two responses*)

Case: one or more SCALE sequences that calculate output quantities of interest (e.g., CSAS model)

* alternative:

```
read case [myCase]
import=/path/to/file
end case
```

** Note that the mean value and the standard deviation are by default calculated for each defined response.

Sampler Input

Specification of perturbations



PARAMETERS block – Samples, execution, and print control

- Sample control:
 - `n_samples=<number of samples>`
 - `first_sample=<number of first sample>` (default: **1**)
- Run control:
 - `run_cases=[yes|no]` – If "no", only the sample input files are generated
 - `force_run=[yes|no]` – If "yes", enforces re-run of all sample inputs even if output files are present
- Output control:
 - `plt=[yes|no]` – Produce plot file histograms with response distributions that can be viewed with Fulcrum)
 - `csv=[yes|no]` – If "yes", produce CSV files with response data in individual tables
 - `print_data=[yes|no]` – Print per-sample values in the main output
 - `print_chi2=[yes|no]` – Print chi-square normality test in the main output

Defaults indicated in **bold**

PARAMETERS block – Perturbation types

- **Model input parameter perturbation:**

- `perturb_geometry=[yes|no]` – Perform input file perturbation

} Requires VARIABLE definitions (see next slides)

- **Nuclear data perturbation:**

- `perturb_xs=[yes|no]`
 - `library=<path>`
 - `mg_factors_library=<path>`
 - `coverx_library=<path>`
 - `perturb_kinetics=[yes|no]`
- `perturb_yields=[yes|no]`
- `perturb_decay=[yes|no]`

- Cross section perturbation

- Name of the multigroup library that is to be perturbed (e.g., xn252v7.1)
- Path to the multigroup perturbation factors (default: `${DATA}/perturb/56n.v7.1`)
- Path of the covariance library (default: `${DATA}/scale.rev08.56groupcov7.1`)
- When perturbing XS, perturb decay constants of delayed neutron groups
- FP yield perturbation
- Decay data perturbation

Nuclear data perturbations are limited to a sample size of 1,000 because 1,000 pre-generated nuclear data perturbations are available with SCALE.

Cross section perturbations require specification of “**library**”. This library name must match the multigroup library that is used in the used SCALE sequence.

```
library="xn252v7.1"  
...  
sequence=csas6  
my title  
xn252v7.1  
...
```

VARIABLE block – input parameter perturbation

- Available distributions for sampling composition and geometry input parameters:
 - Uniform: uniform distribution over an interval
 - Normal: normal distribution, optionally truncated
 - Beta: beta distribution within interval
- A variable can also be defined as dependent from another variable:
 - Distribution type “expression”
 - Expression definition supports basic arithmetic operations and the use of other defined variables
- The variable can be associated with specific cases by adding a list of cases
- Variable block requires specification of `perturb_geometry=yes` in the Sampler parameters block

Example uniform distribution definition

```
read variable[rfuel]  
  distribution=uniform  
  minimum = 0.40  
  maximum = 0.42  
end variable
```

Example expression definition

```
read variable[c1_u238]  
  distribution = expression  
  expression="100.0-c1_u235"  
end variable
```

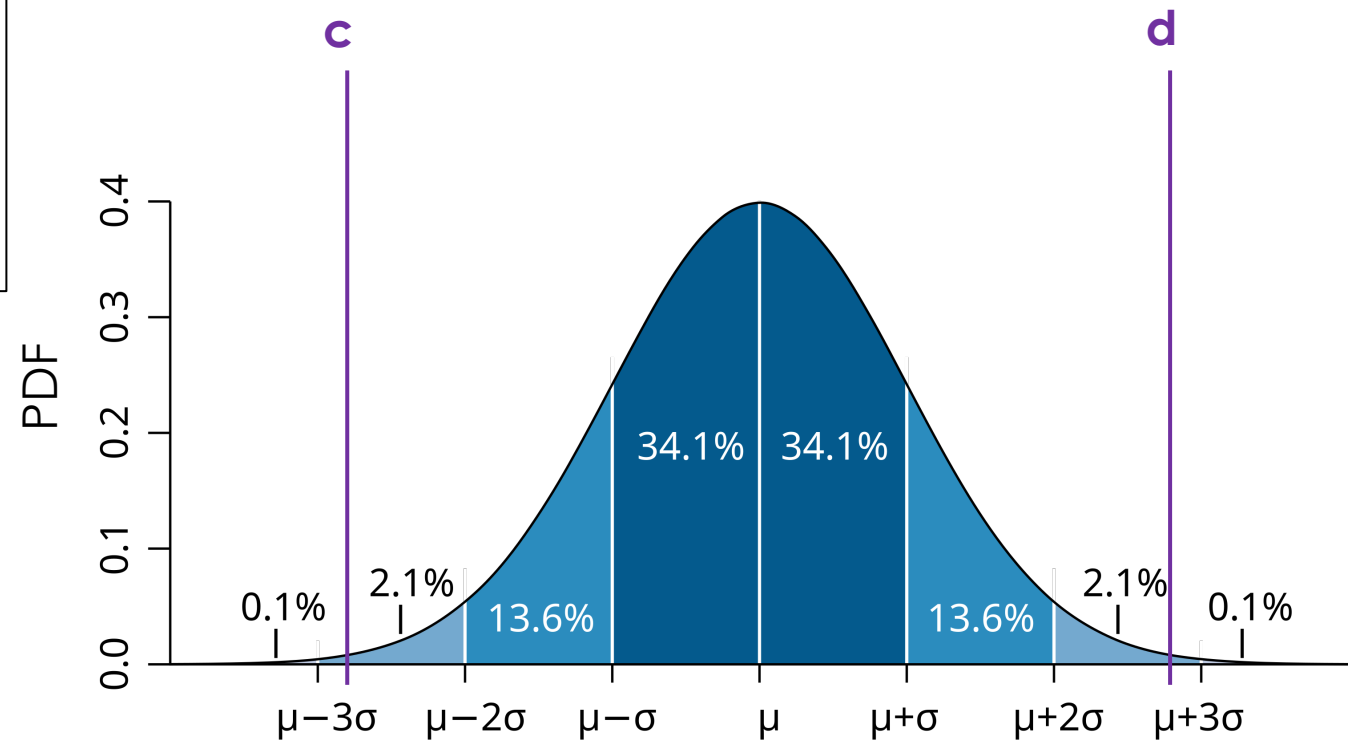
VARIABLE block – Normal distribution

Definition

```
read variable[var_id]
  distribution=normal
  value= $\mu$ 
  stddev= $\sigma$ 
  minimum=c
  maximum=d
  cases = <case1> <case2> ... end
end variable
```

Example

```
read variable[c1_u235]
  distribution=normal
  value=95.0
  stddev=0.05
end variable
```



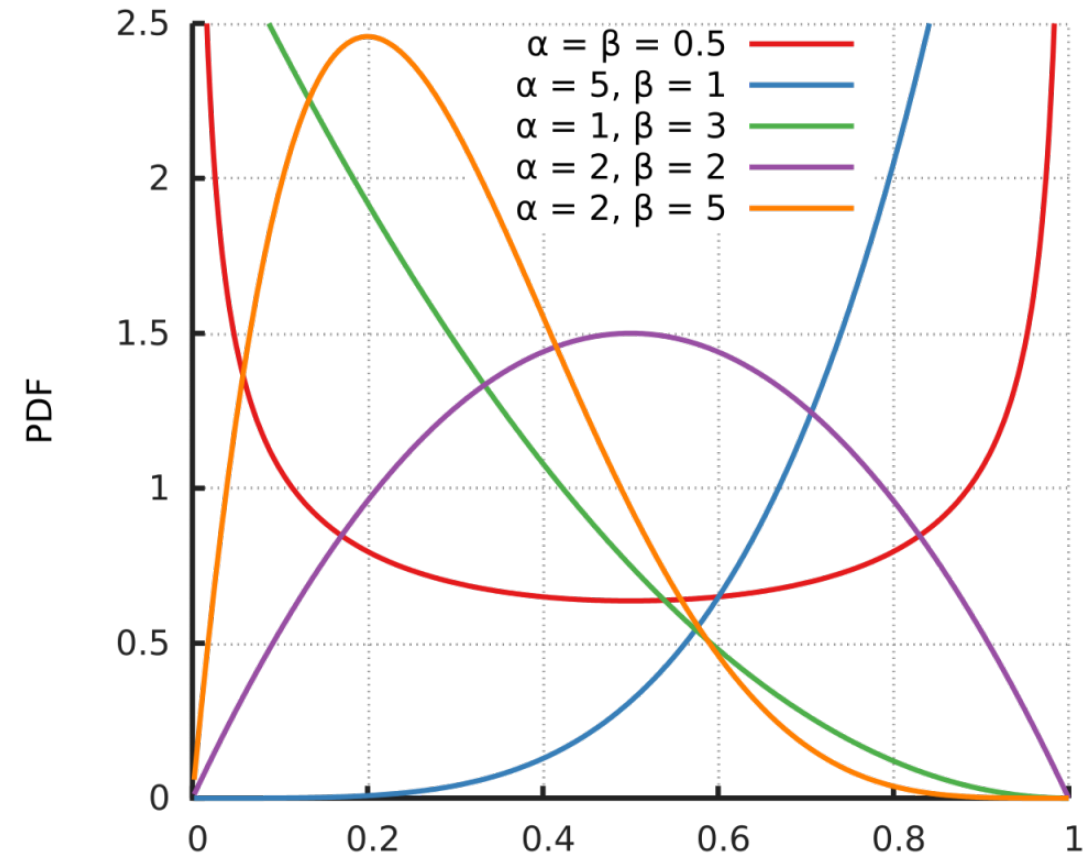
VARIABLE block – Beta distribution

Definition

```
read variable[var_id]
  distribution=beta
  minimum=c
  maximum=d
  beta_a= $\alpha$ 
  beta_b= $\beta$ 
  cases = <case1> <case2> ... end
end variable
```

Example

```
read variable[rclad]
  distribution=beta
  beta_a=2
  beta_b=6
  minimum=0.45
  maximum=0.50
end variable
```



VARIABLE block – Allowed operations in expression

Operator	Example
+	3+3, x+y
+	+3, +(-3), +x
-	3-3, x-y
-	-3, -(3+3), -x
*	3*3, x*y
/	3/3, x/y
%	3%3, x%y
^	3^3, x^y
==	3==3, x==y
!=	3!=3, x!=y
<	3<3, x<y
>	3>3, x>y
<=	3<=3, x<=y
>=	3>=3, x>=y
and	1 and 1, x==y and y==z
or	1 or 1, x==y or y==z
nand	1 nand 1, 1 nand 0, x nand y
xor	1 xor 1, 1 xor 0, x xor y
nor	1 nor 1, 1 nor 0, x nor y
abs(x)	abs(1),abs(-1),abs(x),abs(y)
min(x,...n)	min(1),min(1,2,3,4,5,x,y),etc
max(x,...n)	max(1),max(1,2,3,4,5,x,y),etc
avg(x,...n)	avg(1),avg(1,2,3,4,5,x,y),etc
Sum(x,...n)	sum(1),sum(1,2,3,4,5,x,y),etc
mul(x,...n)	mul(1),mul(1,2,3,4,5,x,y),etc
floor(x)	floor(1)=1, floor(-1.0)=-1.0, floor(-1.1)=-2.0

Operator	Example
floor(x)	floor(1)=1, floor(-1.0)=-1.0, floor(-1.1)=-2.0
ceil(x)	ceil(1)=1,ceil(-1.1)=-1
round(x)	round(1.1)=1,round(1.5)=2,round(1.9)=2
roundn(x,n)	roundn(1.15,1)= 1.2, roundn(1.519,2)= 1.52
cos(x)	cos(0.0)=1
sin(x)	sin(0.0)=0
tan(x)	tan(0.0)=0
sec(x)	sec(0.0)=1
csc(x)	csc(1.0)= 1.1884
cot(x)	cot(1.0)= 0.642093
sinh(x)	sinh(1.0)= 1.1752
cosh(x)	cosh(1.0)= 1.54308
tanh(x)	tanh(1.0)= 0.761594
log(x)	log(2.7182818)=1
log10(x)	log10(10.0)=1
exp(x)	exp(1)=2.7182818
clamp(l,x,u)	clamp(-1,1,+1)=1,clamp(-1,-5,+1)=-1
inrange(l,x,u)	inrange(-1,1,+1)=1,inrange(-1,-5,+1)=0
sgn(x)	sgn(+1)=1,sgn(-1)=-1,sgn(0)=0,sgn(5)=1
sqrt(x)	sqrt(1)=1,sqrt(4)=2
root(x,y)	root(4,2)=2,root(8,3)=2
true	true
false	False
if(cond,true,false)	if(x > y, x, y) = x

Using placeholders to apply defined variables

A placeholder inside the SEQUENCE block is defined via **#{variable_id}**. Sampler will replace this placeholder by a variable value when building the sample input file.

- No expressions are allowed
- Variable must exist

Sampler variable definition

```
read variable[u235_wo]
  distribution=normal
  value=5.00
  stddev=0.01
  min=4.98
  max=5.02
end variable
read variable[u238_wo]
  distribution=expression
  expression="100.0 - u235_wo"
end variable
```

Nominal composition block

```
read composition
  uo2 1 den=10.4 900
                                92235 5.0
                                92238 95.0 end
end composition
```



Composition block with placeholders

```
read composition
  uo2 1 den=10.4 900
                                92235 #{u235_wo}
                                92238 #{u238_wo} end
end composition
```

Verifying the variable definition

- **shell** is a valid SCALE sequence and supports basic shell commands
- Using shell as sequence in the Sampler input offers an easy way to test the sampled variables
- Use “echo” in shell to print sampled variables

```
=sampler
  read parameters
    n_samples = 5
    perturb_geometry = yes
    force_run = yes
    run_cases = yes
  end parameters

  read case[my_case]
    sequence=shell
      echo NORMAL= #{MY_NORMAL}
      echo UNIFORM= #{MY_UNIFORM}
    end sequence
  end case

  read variable[MY_NORMAL]
    distribution=normal
    minimum=-2 value=-1 maximum=0 stddev=2
  end variable

  read variable[MY_UNIFORM]
    distribution=uniform
    minimum=-2 value=-1 maximum=0
  end variable
end
```

Sampler Input

Post-processing and analysis



RESPONSE block

- The RESPONSE block is used to collect quantities of interest from the sample calculations
- Responses can be entered once and shared across all selected cases, or they can be entered inside a specific CASE block and only be evaluated for that case
- Sampler calculates mean value and standard deviation for all responses
 - Results printed in the .out file
 - Default parameter `CSV=yes` causes mean values, standard deviations,, etc. to be printed into CSV files
 - Default parameter `PTP=yes` causes histograms, scatter plots, etc. to be printed into PTP files
- Further analysis of responses can be requested through ANALYSIS block (see later)

```
read response[r_id]  
  type=TYPE  
  [type-specific-params]  
end variable
```

Available types:

- **grep** – Regular expression from the text output file
- **origen_nuclides** – Concentrations from ORIGEN concentration (F71) file
- **opus_plt** – Data from OPUS-generated PLT file
- **triton** – TRITON's homogenized cross-sections (xfile016)
- **stdcmp** – Concentrations from TRITON's standard composition files
- **variables** – Perturbed input parameter

RESPONSE block – GREP response

- A generic regular expression (regexp) mechanism is provided to allow extraction of results from text output file
- “grep” is executed with the “-o” option, which returns the matched portion of the line (and not the whole line)
- Usually, two expressions are necessary: one to find the line of interest, and another to extract only the desired value
- Grep supports the POSIX character classes
 - `[:digit:]` matches a single digit 0-9
 - `[:space:]` matches a single space or tab
 - “+” is used to match one or more repeats
 - “*” is used to match zero or more repeats
 - “.” matches any character
 - “\.” matches a period

Example to grep k_{eff} response

```
read response[mykeff]
  type = grep
  regexp="keff = [[:digit:]]\.[[:digit:]]*"
  regexp="[[:digit:]]\.[[:digit:]]*"
end response
```

RESPONSE block – GREP response shortcuts

- A common GREP response is k_{eff}
- Grepping this output value can be complicated with POSIX
- SCALE shortcuts simplify the GREP definition

Regular expression shortcuts for SCALE sequences

<code>regexp = ":kenova.keff:"</code>	k-eff from KENO5 / CSAS5 sequence
<code>regexp = ":kenova.ealf:"</code>	EALF from KENO5 / CSAS5 sequence
<code>regexp = ":kenova.nubar:"</code>	nu-bar from KENO5 / CSAS5 sequence
<code>regexp = ":kenovi.keff:"</code>	keff from KENO6 / CSAS6 sequence
<code>regexp = ":xsdrn.lambda:"</code>	lambda from XSDRN / CSAS1 sequence
<code>regexp = ":scale.number:"</code>	matches any number (e.g. "1", "1.0", "1.23e-7", "-0.3")

- "regexp" gets the k_{eff} value
- "eregexp" gets the uncertainty (error) in the k_{eff} value
- The uncertainty of interest is the standard deviation of the k_{eff} values; the statistical uncertainty in each k_{eff} value is neglected in Sampler, as usually done in such uncertainty analyses.

Target response in output file

```
best estimate system k-eff      0.99352 + or - 0.00019
```

GREP response with POSIX

```
read response[mykeff]
  type = grep
  regexp = "best estimate system k-eff[[:space:]]+[[:digit:]]+\.[[:digit:]]+"
  regexp = "[[:digit:]]+\.[[:digit:]]+"
end response
```

GREP response with shortcut

```
read response[mykeff]
  type = grep
  regexp = ":kenova.keff:"
  eregexp = ":kenova.keff:"
end response
```

RESPONSE block – ORIGIN_NUCLIDES response

	origen_nuclides
units	moles
nuclides	User-selected at the time of post-processing (no need to rerun sampler to look at different nuclides!)
filename	*.samplerfiles/ \${SUBDIR}/ \${OUTBASENAME}.f71
format	binary
precision	16 digits

- **Analysis of nuclide concentrations in the ORIGIN concentration file (.f71 file)**
- Traditionally, this file has file name "ft71f001" in the SCALE temporary working directory `${TMPDIR}`
- Sampler copies the file automatically into a specific subfolder for a sample:
`${TMPDIR}/ft71f001 → *.samplerfiles/my_pert_c00001/my_pert_c00001.f71`
 - **Sampler must find this file; otherwise, it repeats the calculation!**
 - The copy command for the file must succeed or the post-processing does not work.

RESPONSE block – ORIGEN_NUCLIDES response (cont.)

```
read response[moles]
type = origen_nuclides
mixture = 1
nuclides = rn-200 rn-221 rn-222
           rn-223 rn-224 rn-225
           rn-226 end
end response
```

- Selected nuclides can include any of the 2,237 nuclides on the f71 file
- Selected mixture is a *case number* on the f71 file
- Case number depends on the sequence that generated the f71 file:
 - TRITON: case number is mixture number
 - Polaris: case number is material number
 - ORIGEN: case number is ORIGEN case id

RESPONSE block – OPUS_PLT response

	opus_plt
units	User-selected
nuclides	User-selected at time of running all the sample inputs via powerful ranking or explicit selection
filename	*.samplerfiles/\${OUTBASENAME}.<n>.plt (where <n> is a very long integer starting at 0)
format	text
precision	5 digits

- **Analysis of OPUS results from OPUS result file (plt file)**
- OPUS extracts data from an ORIGEN concentration file (.f71 file)
- OPUS calculates additional quantities based on this data: decay heat, activity, spectral data, etc.
 - See SCALE manual chapter [5.1.7](#) for more details on OPUS
- Each “=opus” call (OPUS sequence in Sampler input) creates a numbered *.plt file from an ORIGEN concentration file
- The Sampler response accesses the file by the **ndataset** number (**depletion step**)

RESPONSE block – OPUS_PLT response (cont.)

```
' first call to opus results in *0.plt file
=opus
  data="ft71f001"
  library="end7dec"
  units=curies
  nrank=40
  symnuc=rn-222 end
end
```

```
' access *0.plt contents
read response[mycuries]
  type=opus_plt
  ndataset=0
  nuclides=rn-222 total end
end response
```

- By default, OPUS lists the top 40 nuclides contributing to the units of interest
 - For example, if units=curies, then the top 40 nuclides contributing to activity are included
- Only available nuclides in the plt file can be requested in the response block
 - This includes useful *total* and *subtotal*
- To get the **ndataset** number right, it's useful to run the nominal calculation first to confirm the availability and naming of the plt files

OPUS .plt file

```
time (d)
Ci
nuclide
6 24
      0.000e+00 3.653e+03 3.653e+05 3.653e+07 3.653e+09 3.653e+11
u238 8.001e-05 8.001e-05 8.001e-05 8.001e-05 7.989e-05 6.852e-05
th234 0.000e+00 8.001e-05 8.001e-05 8.001e-05 7.989e-05 6.852e-05
pa234m 0.000e+00 8.001e-05 8.001e-05 8.001e-05 7.989e-05 6.852e-05
u234 0.000e+00 2.238e-09 2.256e-07 1.968e-05 7.989e-05 6.852e-05
pb210 0.000e+00 1.089e-17 1.347e-10 7.086e-06 7.989e-05 6.852e-05
bi210 0.000e+00 1.089e-17 1.347e-10 7.086e-06 7.989e-05 6.852e-05
po210 0.000e+00 1.089e-17 1.347e-10 7.086e-06 7.989e-05 6.852e-05
po218 0.000e+00 1.484e-16 1.347e-10 7.086e-06 7.989e-05 6.852e-05
rn222 0.000e+00 1.484e-16 1.347e-10 7.086e-06 7.989e-05 6.852e-05
ra226 0.000e+00 1.484e-16 1.347e-10 7.086e-06 7.989e-05 6.852e-05
th230 0.000e+00 1.029e-13 1.035e-09 7.086e-06 7.989e-05 6.852e-05
bi214 0.000e+00 1.484e-16 1.347e-10 7.086e-06 7.989e-05 6.852e-05
pb214 0.000e+00 1.484e-16 1.346e-10 7.085e-06 7.987e-05 6.850e-05
po214 0.000e+00 1.484e-16 1.346e-10 7.084e-06 7.987e-05 6.850e-05
pa234 0.000e+00 1.280e-07 1.280e-07 1.280e-07 1.278e-07 1.096e-07
tl210 0.000e+00 3.117e-20 2.828e-14 1.488e-09 1.678e-08 1.439e-08
at218 0.000e+00 2.969e-20 2.693e-14 1.417e-09 1.598e-08 1.370e-08
tl206 0.000e+00 1.458e-23 1.803e-16 9.488e-12 1.070e-10 9.174e-11
rn218 0.000e+00 2.969e-23 2.693e-17 1.417e-12 1.598e-11 1.370e-11
hg206 0.000e+00 2.068e-25 2.559e-18 1.346e-13 1.518e-12 1.302e-12
pb209 0.000e+00 2.650e-24 2.404e-18 1.265e-13 1.426e-12 1.223e-12
bi209 0.000e+00 0.000e+00 5.210e-47 4.726e-41 2.487e-36 3.053e-35
subtotal 8.001e-05 2.402e-04 2.404e-04 3.307e-04 1.119e-03 9.593e-04
total 8.001e-05 2.402e-04 2.404e-04 3.307e-04 1.119e-03 9.593e-04
```

Response post-processing generates CSV results files for further analysis with external tools

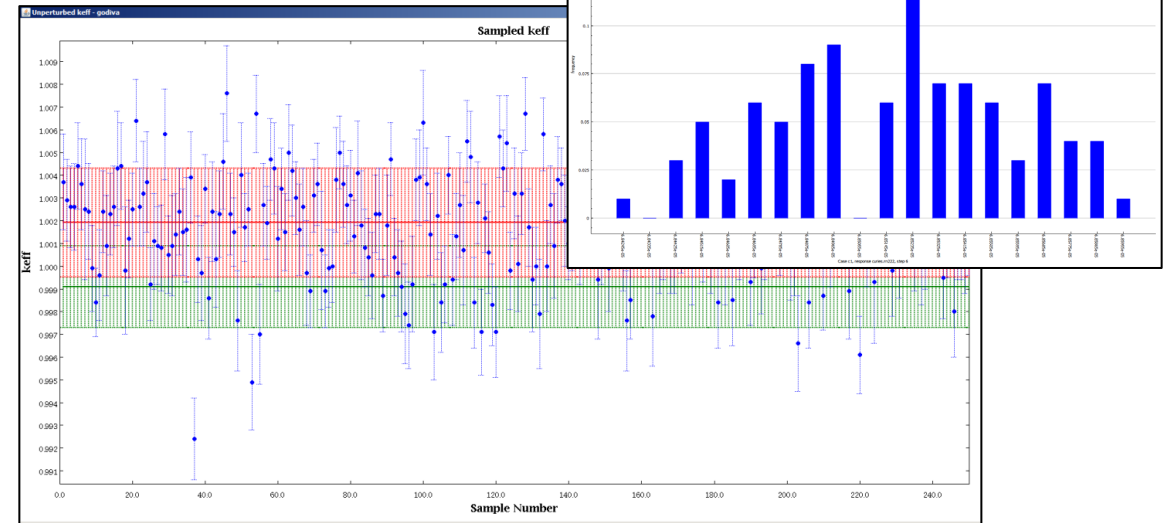
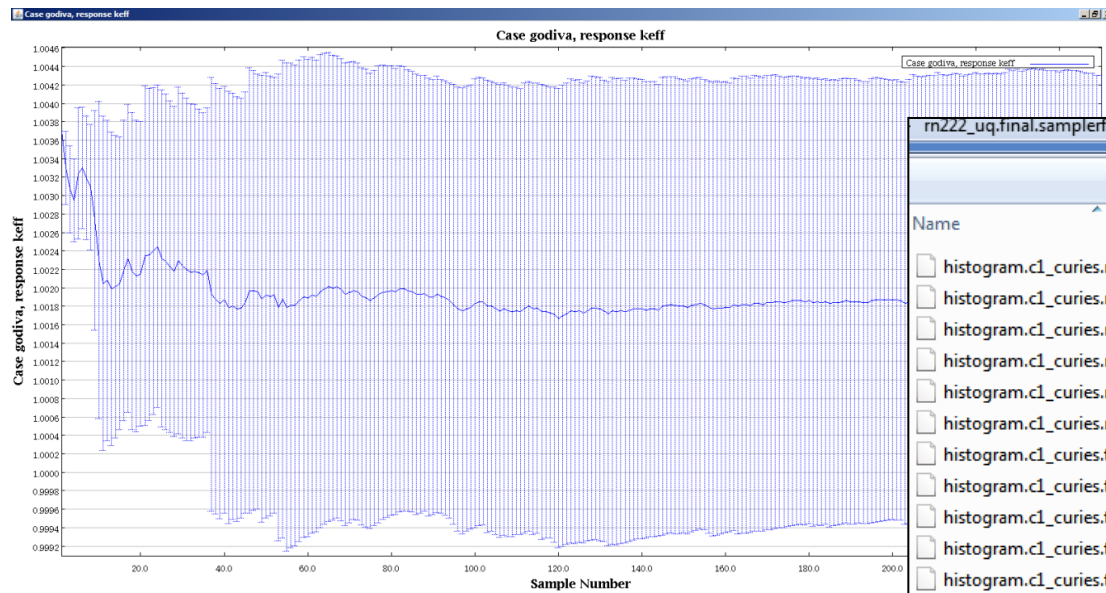
- Standard deviation
- Averages
- Individual sample values
- Correlation between responses
- Covariances between responses

Note: Our CSV files are **semi-colon (;) separated**—not comma (,) separated—to avoid international incidents where "," may be used as the decimal (e.g. "3,14" instead of "3.14").

```
.
├── my_sampler.inp
├── my_sampler.out
├── my_sampler.samplerfiles
│   ├── c1_pert_00000.inp
│   ├── c1_pert_00000.out
│   ├── c1_pert_00001.inp
│   ├── ...
│   ├── correlations
│   │   ├── response_table.dynamic.1.corr.step0001.csv
│   │   ├── ...
│   │   └── response_table.dynamic.2.corr.step0013.csv
│   ├── covariances
│   │   ├── response_table.dynamic.1.cov.step0001.csv
│   │   ├── ...
│   │   └── response_table.dynamic.2.cov.step0013.csv
│   ├── histograms
│   │   └── *.ptp
│   ├── histories
│   │   └── *.ptp
│   ├── response_tables
│   │   ├── response_table.dynamic.1.avg.csv
│   │   ├── response_table.dynamic.1.stddev.csv
│   │   ├── response_table.dynamic.1.val.csv
│   │   ├── response_table.dynamic.2.avg.csv
│   │   ├── response_table.dynamic.2.stddev.csv
│   │   └── response_table.dynamic.2.val.csv
│   ├── running_averages
│   │   └── *.ptp
│   └── scatter_plots
│       └── *.ptp
```

Response post-processing generates PTP results files for visualization with Fulcrum

- Scatter plots showing
 - All data points
 - Average result and nominal result
- Histograms
- Running averages



rn222_uq.final.samplerfiles > histograms			rn222_uq.final.samplerfiles > scatter_plots		
Name	Date modified	Type	Name	Date modified	Type
histogram.c1_curies.rn222.step00001.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00001.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.rn222.step00002.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00002.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.rn222.step00003.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00003.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.rn222.step00004.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00004.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.rn222.step00005.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00005.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.rn222.step00006.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.rn222.step00006.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00001.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00001.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00002.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00002.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00003.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00003.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00004.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00004.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00005.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00005.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_curies.total.step00006.ptp	3/2/2016 4:38 PM	PTP	scatter_plot.c1_curies.total.step00006.ptp	3/2/2016 4:38 PM	PTP File
histogram.c1_moles.rn222.step00001.ptp	3/2/2016 4:38 PM	PTP			

ANALYSIS block

Allows users to analyze relationship between responses

- Any response type at any depletion step/index
- Relationship evaluated between “target” and “source” responses
- Results written in CSV files in “response_tables” subdirectory

```
read analysis [analysis_name]
  type = TYPE
  target = tresponse1 tresponse2 ... end
  sources = sresponse1 sresponse2 ... end
end analysis
```

Analysis types

pearson_corr
covariances

correlation_matrix

target responses are ignored for this type

covariance_matrix

target responses are ignored for this type

r2

sensitivity index for nuclear data uncertainty analysis

ANALYSIS block – Target and source response definition

- **case:response.sub(instance)** used to identify user defined case-dependent responses
 - case: user-defined Sampler case name
 - response: user-defined response name under the Sampler case
 - sub: sub-response for the response (e.g., nuclide names for *origen_response* type), not specified if there is none
 - instance: index
 - Occurrence of grep response match in output (e.g., for multiple depletion calculations with multiple k-eff results in the output)
 - Time/depletion step for nuclide concentration
 - Starts with 0 (default)
 - “last”: the last occurrence
 - “all”: creates a list of responses for all indices from 0 to last
- Examples : **myc1:myorig.u-235(15)**, **myc1:myorig.u-235**, **myc1:mykeff(all)**

ANALYSIS block: covariances

- Covariance calculation between each response in the *sources* response list and the *target* response list

```
read analysis [cov_with_U235_initial]
  type = covariances
  targets = mycase1:app.u-235(0)
  sources = mycase2:exp.u-235(last), mycase2:ex.pu-239(14)
end
end analysis
```

- Calculated covariances:
 - cov(mycase1:app.u-235(0), mycase2:exp.u-235(last))
 - cov(mycase1:app.u-235(0), mycase2:ex.pu-239(14))

ANALYSIS block: pearson_corr

- Pearson correlation coefficient between each responses

```
read analysis [do_they_have_same_sensitivity]
  type = pearson_corr
  targets = mycase1:app.u-238(0) end
  sources = mycase2:exp.u-235(last) mycase2:ex.pu-239(last) end
end analysis
```

- Calculated covariances:
 - `cor(mycase1:app.u-238(0), mycase2:exp.u-235(last))`
 - `cor(mycase1:app.u-238(0), mycase2:ex.pu-239(last))`

Is the Pearson correlation coefficient the same as similarity coefficient c_k ?

ANALYSIS block: correlation_matrix and covariance_matrix

- Correlation (covariance) calculation between responses in the sources response list (*target* responses are ignored for this type)

```
read analysis [iso_ck]
  type = covariances
  sources = c1:rca1.u-235(0), c1:rca1.pu-239(last)
end
end analysis
```

- Calculated matrixes:
 - cov(c1:rca1.u-235(0), c1:rca1.pu-239(last))
 - var(mycase2:exp.u-235(last))
 - var(mycase1:app.u-235(15))

	c1:rca1.u-235(0)	c1:rca1.pu-239(last)
c1:rca1.u-235(0)	var1	cov
c1:rca1.pu-239(last)	cov	var2

ANALYSIS block: R^2

- R^2 calculation for responses in the target response list, for all nuclide reactions of all nuclides listed in the sources

```
read analysis [whos_fault_large_dk]  
  type = r2  
  targets = c1:response_keff(last) end  
  sources = h-1 u-235 u-238 end  
end analysis
```

- For each response, results include a ranked list of the following three values for each nuclide reaction: R^2 / 95% confidence interval / 95% significance bound
- Results must be carefully inspected to determine if R^2 results are above the significance bound
- Results must be examined with caution: correlations between reactions impact the ranking; understanding of these correlations is required to correctly interpret results
 - Most common example is the correlation between elastic and inelastic scattering causing large R^2 values for elastic scattering although this reaction may not be a significant uncertainty contributor
 - Ranking should not be blindly taken, but expert judgement and additional analyses must be applied

Sampler Input

Parametric study



Parametric study

PARAMETRIC block allows investigating the impact of one or more modeling parameters on selected model responses

- The parameter values are selected from a user defined range of variation in equal intervals (**uniform** distribution).
- All combinations of different modeling parameters are analyzed if there are more than one modeling parameter.
- Sampler generates a summary table of the parametric study, including values for which the minimum and maximum of each response occurs.
- Sampler generates PTP plot files showing the dependency of each requested response on each variable.

Example: Investigation of the effect of the moderator density and the fuel temperature on k_{eff}

```
read variable[mod_dens]
  distribution=uniform
  minimum = 0.5
  maximum = 0.9
end variable
read variable[fuel_temp]
  distribution=uniform
  minimum = 700.0
  maximum = 900.0
end variable
read parametric
  variables = mod_dens fuel_temp end
  n_samples =      10      6 end
end parametric
```

Parametric study involves $10 \times 6 = 60$ samples to adequately cover combinations of density and temperature

Sampler Input

How to convert a standard SCALE
input into a Sampler input



How to convert a standard SCALE input into a Sampler input

1. Ensure that the SCALE sequence input in question is accurate and runs successfully
2. Wrap the sequence input in sequence and case blocks, and assign the case an arbitrary identifier
3. Add a parameters block and enable perturbation of choice

```
=sampler
read parameters
  n_samples=100
  perturb_xs=yes
  perturb_geometry=yes
end parameters
read case[my_case_id]
  sequence=csas5
  sample problem 1 case 2c8 bare
  v7.1-252
  read comp
    uranium 1 den=18.76 1 300 92235 93.2 92238 5.6 92234 1.0
              92236 0.2 end
  end comp
  read geometry
    unit 1
      cylinder 1 1 5.748 5.3825 -5.3825
      cuboid 0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
    end geometry
    read array
      gbl=1 ara=1 nux=2 nuy=2 nuz=2 fill f1 end fill
    end array
  end data
end sequence
end case
end
```

How to convert a standard SCALE input into a Sampler input

4. a. For nuclear data perturbations:

- Change the nuclear data library alias in the sequence to the actual filename
- Add this library name into the Sampler parameters block as "library"

```
=sampler
read parameters
  n_samples=100
  perturb_xs=yes
  perturb_geometry=yes
  library="xn252v7.1"
end parameters
read case[my_case_id]
  sequence=csas5
  sample problem 1 case 2c8 bare
  xn252v7.1
  read comp
    uranium 1 den=18.76 1 300 92235 93.2 92238 5.6 92234 1.0
    92236 0.2 end
  end comp
  read geometry
    unit 1
    cylinder 1 1 5.748 5.3825 -5.3825
    cuboid 0 1 6.87 -6.87 6.87 -6.87 6.505 -6.505
  end geometry
  read array
    gbl=1 ara=1 nux=2 nuy=2 nuz=2 fill f1 end fill
  end array
end data
end sequence
end case
end
```

How to convert a standard SCALE input into a Sampler input

4. b. For input parameter perturbations:

- Add variable blocks
- Add corresponding placeholders into SCALE sequence

```
=sampler
read parameters
  n_samples=100
  perturb_xs=yes
  perturb_geometry=yes
end parameters
read variable[my_var]
...
end variable
read case[my_case_id]
  sequence=csas5
  sample problem 1 case 2c8 bare
  v7.1-252
  read comp
    uranium 1 den=18.76 1 300 92235 #{my_var}
              92238 5.6 92234 1.0 92236 0.2 end
  end comp
...
  read array
    gbl=1 ara=1 nux=2 nuy=2 nuz=2 fill f1 end fill
  end array
end data
end sequence
end case
end
```

How to convert a standard SCALE input into a Sampler input

5. Add response and analysis blocks

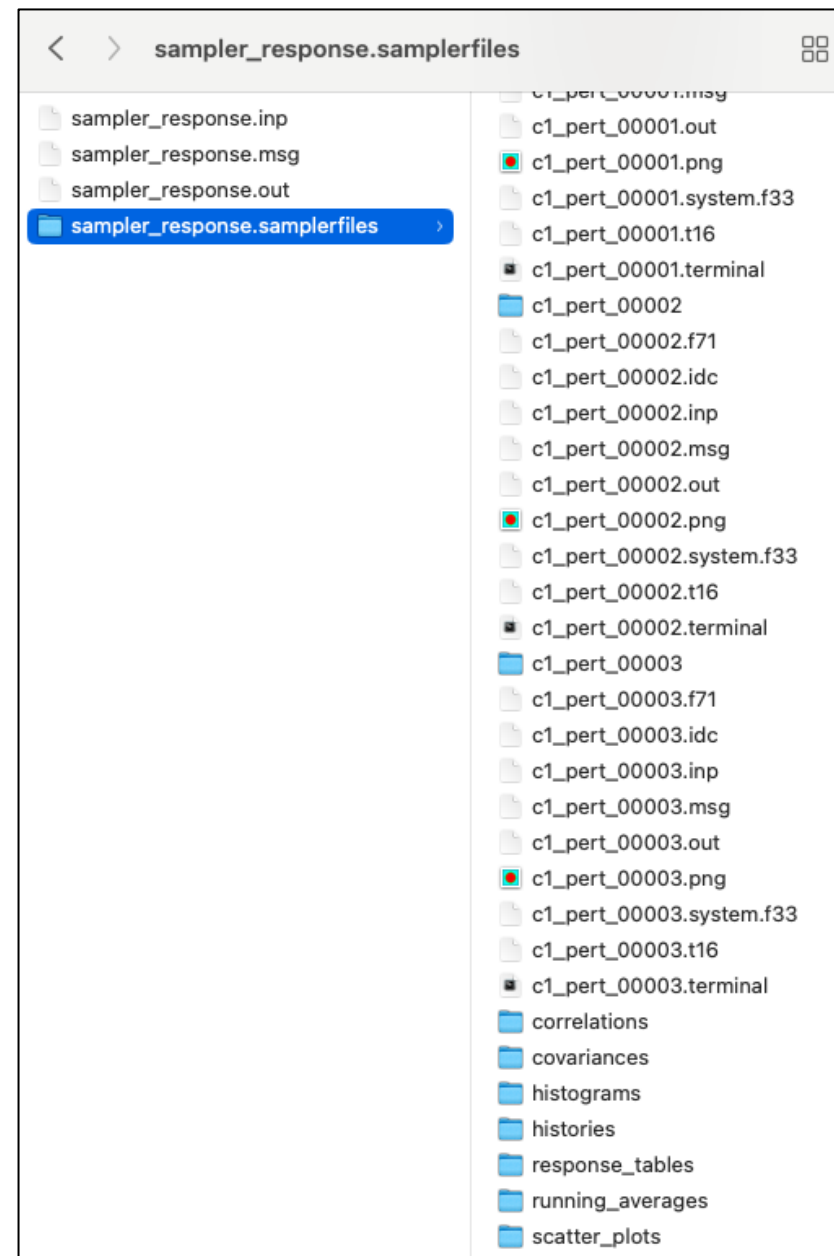
```
=sampler
read parameters
  n_samples=100
  perturb_xs=yes
  perturb_geometry=yes
end parameters
read variable[my_var]
...
end variable
read response[my_resp]
...
end response
read analysis[my_analysis]
...
end analysis
read case[my_case_id]
  sequence=csas5
  sample problem 1 case 2c8 bare
  v7.1-252
  read comp
    uranium 1 den=18.76 1 300 92235 #{my_var}
  ...
end data
end sequence
end case
end
```

Sampler Execution



Sampler execution

- The following steps are executed when running a Sampler input:
 1. Subdirectory “\${BASENAME}.samplerfiles” is created
 2. Sample input files are generated and stored in this new subdirectory
 3. Sample inputs are run
 4. Post-processing is performed
 5. Results of the post-processing and analyses (CSV and PTP files) are stored in sub-folders within “\${BASENAME}.samplerfiles”
- Important note on the post-processing: Sampler must find all expected output files to avoid triggering a re-run of inputs and unintentional overwriting of outputs:
 - *.out file with statement “SCALE is finished” and no execution error
 - *.f71 and/or *.plt, depending on the requested responses



Advanced Sampler execution

- Sampler supports **threading**:
 - Threading enables parallel execution of sample inputs
 - Threading is enabled via the “-I” parameter
 - Example command line call: `scalerte -I 4 my_sampler.inp`
 - Threading can be enabled in Fulcrum by customizing the SCALE call
- Sample inputs can be run manually if desired:
 1. Use Sampler to generate sample inputs (using `run_cases=no`)
 2. Submit sample inputs to computing cluster
 3. Ensure successful completion of calculations and collect results
 4. Call Sampler for post-processing

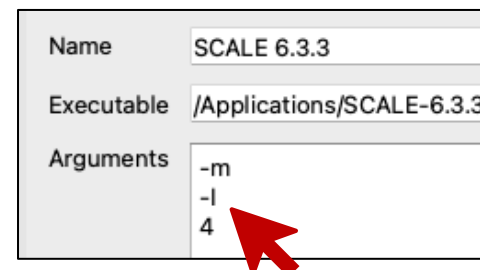
Reminder: Sampler must find all expected output files to avoid triggering a re-run of inputs and unintentional overwriting of outputs! Missing files or incorrect paths are the most common cause of issues when performing post-processing after manually running sample inputs.

Enable threading in Fulcrum

1. Click on "SCALE 6.3.3" and find the "Customize"



2. Add arguments: one flag or value per line



Tips & Tricks



Sampler input generation tips & tricks

- Inspect generated sample inputs without running them by using `run_cases=no`
- Ensure nominal and first sample calculation run successfully (sample 0 and sample 1) before running all samples
 - Also check that they are providing different results
- Test variable definitions with echo commands in a shell sequence
- Double-check paths if you are using any
 - If using the `${DATA}` variable, ensure this is correctly set in the environment
 - If using injection files, ensure these are correctly copied to the `${TMPDIR}` so that they can be found during SCALE input execution
 - Check the input file for sample 1 to confirm all listed paths are valid

Advanced Sampler application and insights



Advanced variable definition with SIREN

- SCALE Input Retrieval ENgine (SIREN) is a package which provides an XPath-like interface to the SCALE input deck represented by a Document Object Model (DOM).
- In Sampler, the user can specify `siren="path"` to have the respective token(s) replaced by a variable value.
- Example: The variable `c1_u235` value is inserted as the weight percent of ^{235}U in the basic standard composition declared as "uranium" in the CSAS5 sequence.
- Detailed information on SIREN can be found in SCALE manual [chapter 6.4.6](#)

```
read variable[c1_u235]  
  distribution=normal  
  value=95.0  
  stddev=0.05  
  siren="/csas5/comps/stdcomp[decl='uranium']/wtpt_pair[id='92235']/wtpt"  
end variable
```

Grepping output responses that are not in the *.out file

Example: Grepping the fast total cross section from the T16 file (TRITON or Polaris output file)

Step 1: Add a “sequence=shell” case to the Sampler input to print the target result to the output

Option A:
Shell*

```
sequence=shell
grep "' *Energy group *1" -A 2 ${OUTDIR}/${BASENAME}.t16 | grep
"^[[[:space:]]][[:space:]]" | awk -v label="total_fast= " "{print label,\\$1}"
end sequence
```

Option B:
Python**

```
sequence=shell
python3 myT16Parser.py -grep "' *Energy group *1" -label "total_fast= "
end sequence
```

Step 2: Add a grep response for the target value to the Sampler input:

```
read response[totalxs]
type=grep
regexp="total_fast= *:scale.number:"
regexp="*:scale.number:"
end response
```

* Remember that grep statements must always be on a single line because each line in a shell sequence is executed separately with SCALE.

** Python scripts can be used for easier grepping or additional pre-processing of results. Ensure that the python script is available in the shell by copying it to the \${TMPDIR} or by using absolute paths.

Insights into application of nuclear data perturbations

- For each nuclear data type, SCALE provides 1,000 pre-generated nuclear data perturbations
- Cross sections:
 - The “**Sample***” files in the “`${DATA}/perturb/56n.v7.1`” folder are AMPX multigroup libraries that store perturbation factors
 - SCALE module **Clarolplus** multiplies these perturbation factors with the multigroup cross sections in the selected master library (e.g., v7.1-56)
 - Sampler adds a Clarolplus input to the beginning of each input
- Decay data:
 - The “**end7dec***” files in the “`${DATA}/perturb/`” folder are binary decay libraries that store perturbed decay data (not perturbation factors)
 - Sampler adds a link to a perturbed decay data library to the beginning of each input
- Fission Yields:
 - The “**Yield***” files in the “`${DATA}/perturb/`” folder are binary libraries that store perturbation factors
 - File “`${DATA}/perturb/yieldxsdir`” links each binary library to a specific actinide and energy
 - SCALE module **YieldSampler** multiplies these perturbation factors with the applied fission yield data
 - Sampler adds a YieldSampler input to the beginning of each input

```
=shell
  ln -sf ~/data/scale.rev04.xn56v7.1 ft89f001
  ln -sf ~/data/perturb/56n.v7.1/Sample1 ft87f001
end
=clarolplus
  in=89
  out=88
  var=87
  isvar=10
  bond=yes
  sam=1
end
=shell
  mv ft88f001 xn56v7.1
  unlink ft87f001
  unlink ft89f001
  mv ft10f001 crawdadPerturbMGLib
end
=shell
  ln -sf ~/data/perturb/end7dec_0000 end7dec
end
=YieldSampler
  pertnum=1
end
=shell
  ls -l ${TMPDIR}
  ln -sf yields yield.data
end
=t-newt parm=bonami
sampler sensitivity
xn56v7.1
read composition
u-235 1 0 1e-5 300 end
u-238 1 0 1e-4 300 end
h-1 1 0 1e-2 300 end
end composition
...
end model
end
```

Sampler Workshop

Material and geometry perturbation



Problem 1: Propagation of fuel temperature uncertainty to PWR pin cell decay heat and nuclide concentrations

1. Copy “pwr_pin_triton.inp” to “sampler_pwr_fuel_temp.inp”
 - Note that this is a TRITON-NEWT depletion of a PWR pin cell to a burnup of ~30 GWd/MTU
2. Modify the TRITON input to obtain a Sampler input
 - Sample size: 10
 - Fuel temperature perturbation: uniform distribution between 873 K and 973 K
 - Add OPUS sequences based on the inputs listed on the right to obtain decay heat and nuclide concentrations for U-235, Pu-239, Cm-244, Nd-148, Cs-137, Sm-149, Gd-155
3. Run Sampler using 4 threads
4. Analyze the results:
 - What is the decay heat uncertainty? Why are there 4 sets of results for decay heat?
 - What are the nuclide concentration uncertainties?
 - Where can you find the mean values, standard deviations, and all individual results for each sample?

```
=opus  
data="ft71f001"  
library="end7dec"  
nrank=40  
units=watts  
end
```

```
=opus  
data="ft71f001"  
library="end7dec"  
case=1  
nrank=40  
symnuc=u235 pu239 cm244  
nd148 cs137 sm149 gd155 end  
units=gram  
end
```

Solution 1:

- Decay heat uncertainty is ~0%, i.e. decay heat is not impacted by fuel temperature
- There are 4 sets of results for decay heat because the f71 file has 4 cases: 1, 0, -1, -2 (see TRITON f71 file description)
- EOL nuclide concentration uncertainties:
 - U-235: 0.11%
 - Pu-239: 0.26%
 - Cm-244: 0.12%
 - Nd-148: 0.01%
 - Cs-137: 0.00%
 - Sm-149: 0.26%
 - Gd-155: 0.16%
- Mean values and standard deviations of all responses are printed at the end of the .out file.
- Mean values, standard deviations, and all individual results for each sample are available in CSV files in the *response_tables* subdirectory.

```
=sampler
read parameters
n_samples=10
perturb_geometry=yes
run_cases=yes
end parameters
read variable [fueltemp]
distribution=uniform
minimum=873
maximum=973
end variable
read response [dh]
type=opus_plt
ndataset=1
nuclides=total end
end response
read response[conc]
type=opus_plt
ndataset=2
nuclides=u235 pu239 cm244 nd148 cs137 sm149 gd155 end
end response
read case[pwr_pin]
sequence=t-depl parm=(bonami,addnux=2)
PWR pin example
v7.1-56
read comp
uo2 1 0.95 #{fueltemp} 92235 4.0 92238 96.0 end
zirc2 2 1 595 end
h2o 3 den=0.7135 1 579 end
boron 3 den=0.7135 653e-06 579 end
n 4 den=0.00125 1 595 end
end comp
...
end sequence
sequence=opus
...
end sequence
end case
end
```

Sampler Workshop

Nuclear data perturbation



Problem 2: Propagation of cross section uncertainties to PWR pin cell decay heat and nuclide concentrations

1. Copy "pwr_pin_triton.inp" or input from problem 1 to "sampler_pwr_pin_xs.inp"
 - Note that this is a TRITON-NEWT depletion of a PWR pin cell to a burnup of ~30 GWd/MTU
2. Modify the TRITON input to obtain a Sampler input
 - Sample size: 10
 - Enable cross section perturbations
 - Ensure the "library" parameter has the same name as the TRITON input
 - Add OPUS sequences to obtain decay heat and nuclide concentrations for U-235, Pu-239, Cm-244, Nd-148, Cs-137, Sm-149, Gd-155
3. Run Sampler using 4 threads
4. Analyze the results:
 - What is the decay heat uncertainty?
 - What are the nuclide concentration uncertainties?

```
=opus  
data="ft71f001"  
library="end7dec"  
nrank=40  
units=watts  
end
```

```
=opus  
data="ft71f001"  
library="end7dec"  
case=1  
nrank=40  
symnuc=u235 pu239 cm244  
nd148 cs137 sm149 gd155 end  
units=gram  
end
```

Solution 2:

- EOL decay heat uncertainty is ~0.14%, i.e. cross section uncertainties have only a small impact on decay heat
- EOL nuclide concentration uncertainties:
 - U-235: 0.52%
 - Pu-239: 1.46%
 - Cm-244: 11.61%
 - Nd-148: 0.31%
 - Cs-137: 0.03%
 - Sm-149: 2.18%
 - Gd-155: 17.60%
- Mean values and standard deviations of all responses are printed at the end of the .out file.
- Mean values, standard deviations, and all individual results for each sample are available in CSV files in the *response_tables* subdirectory.

```
=sampler
read parameters
n_samples=10
perturb_xs=yes
perturb_geometry=no
run_cases=yes
end parameters
read response [dh]
type=opus_plt
ndataset=1
nuclides=total end
end response
read response[conc]
type=opus_plt
ndataset=2
nuclides=u235 pu239 cm244 nd148 cs137 sm149 gd155 end
end response
read case[pwr_pin]
sequence=t-depl parm=(bonami,addnux=2)
PWR pin example
v7.1-56
read comp
uo2 1 0.95 923 92235 4.0 92238 96.0 end
zirc2 2 1 595 end
h2o 3 den=0.7135 1 579 end
boron 3 den=0.7135 653e-06 579 end
n 4 den=0.00125 1 595 end
end comp
...
end sequence
sequence=opus
...
end sequence
end case
end
```

Problem 3: Cross section uncertainty and sensitivity analysis for a PWR pin cell k_{eff}

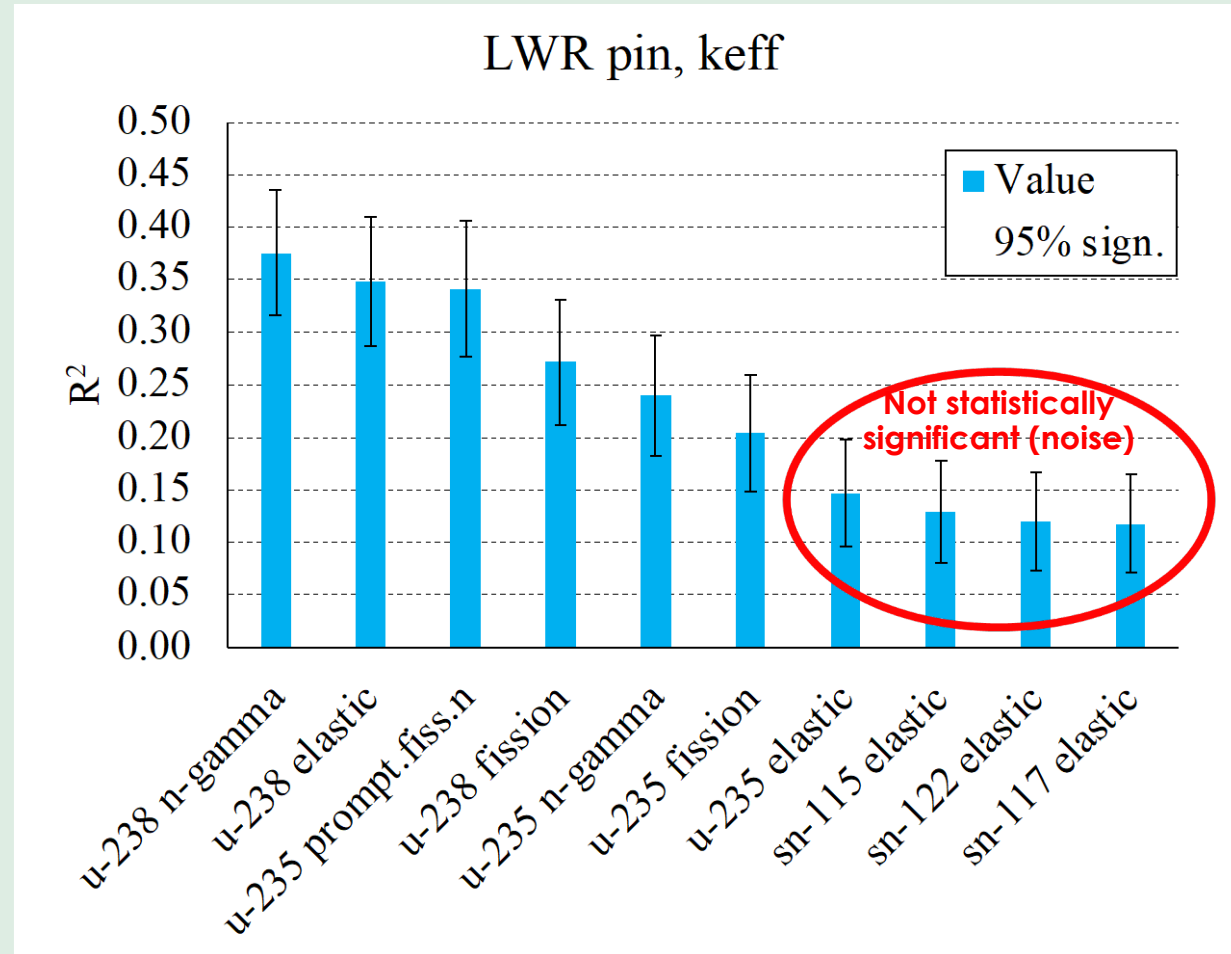
- **Goal:** Determine k_{eff} uncertainty due to nuclear data for a simple LWR pin and identify top contributors to this uncertainty using R^2
- **Prepare:**
 - Use “sampler_lwr_xs_r2_starter.inp” and rename to “sampler_lwr_xs_r2.inp”
 - Confirm that the nuclear data library in the T-XSDRN model is consistent with the one in the Sampler parameter block
 - What is the minimum sample size for the R^2 calculation of this problem, given the default applied covariance library?
 - What are the relevant nuclides of this problem?
- **Update and run input:**
 - Add response block to grep k_{eff} from T-XSDRN using regular expression shortcut (note that XSDRN provides “lambda”)
 - Add analysis block:
 - Type: R2
 - Targets: k_{eff} response
 - Sources: relevant nuclides in the model
 - Add appropriate sample size
 - Run input using threading
- **Analyze:**
 - Check if Sampler calculation was run successfully: warning and error messages, sensitivity analysis output in out/msg files, csv file generated
 - What is the k_{eff} uncertainty?
 - Open sensitivity analysis result file: **.samplerfiles/response_tables/response_table.analysis.analysis_id.csv*
 - How many statistically significant values for R^2 did you obtain?
 - Which reaction did you not expect?

Solution 3:

- Uncertainty in k_{eff} is $\sim 0.56\%$, as typical for LWRs
- With 500 samples, 22 statistically significant values are obtained for R^2 when requesting analysis of R_2 for all nuclides in the system
- Consider both statistical error bars and significant bound
- Elastic scattering reactions are sometimes found among significant contributors because of the correlation with other significant reactions
 - Load covariance library with Fulcrum and check for correlations
 - Also use TSUNAMI and check list of top contributors to develop understanding of your model

ENDF 7.1 covariance library records

```
▼ u-238
  > mt=1 total
  > mt=1018 chi
  > mt=102 n,gamma
  > mt=16 n,2n
  > mt=18 fission
  ▼ mt=2 elastic
    ▼ u-238
      > u-238-mt=2 elastic to u-238-mt=102 n,gamma
      > u-238-mt=2 elastic to u-238-mt=18 fission
      > u-238-mt=2 elastic to u-238-mt=2 elastic
        u-238 mt=2 elastic - Std dev by energy
        u-238 mt=2 elastic - Std dev by group
```



Sampler Workshop

Analysis Block Examples



Problem 4: Similarity analysis of PWR LEU and HALEU fuel with respect to nuclear data uncertainties

1. Copy "sampler_pwr_xs.inp" from problem 1 to sampler_pwr_ck.inp
2. Duplicate pwr_pin case (read case[pwr_pin]... end case) and rename the second case to pwr_pin_haleu (read case[pwr_pin_haleu]... end case)
3. Modify the Triton sequence under pwr_pin_haleu case
 - Increase pin enrichment to 11% and burnup to 70 GWd/MTU
4. Add new response blocks to capture k_{inf} and Pu-239 inventory
5. Add a new analysis block to investigate similarity of k_{inf} at the end of LEU depletion to k_{inf} at 6, 8, 10, 12, 14th depletion steps of the HALEU fuel
6. Add a new analysis block to investigate similarity in Pu-239 productions at MOC, EOC
7. Analyze the results:
 - Does HALEU fuel behave similar to the LEU fuel ?
 - Does the similarity decrease with increasing burnup?, n_samples ?

```
read burndata
  power=40.0 burn=1750 nlib=14 end
end burndata
```

```
regexp = "k-eff =[[[:space:]]]+[[[:digit:]]]+\.[[:digit:]]+"
regexp = "[[:digit:]]+\.[[:digit:]]+"
```

```
read response[nucpu]
type = origen_nuclides
nuclides = pu-239 end
mixture = 1
end response
```

Solution 4:

- LEU and HALEU fuel show similar trends with respect to changes in Nuclear Data. Eigenvalues show similar sensitivities. We can convince ourselves if we can simulate LEU fuel we should be able to simulate HALEU fuel with the same accuracy.

Pearson Corr	pwr_pin:kinf.6
pwr_pin_haleu:kinf.0	0.748
pwr_pin_haleu:kinf.6	0.889
pwr_pin_haleu:kinf.8	0.921
pwr_pin_haleu:kinf.10	0.946
pwr_pin_haleu:kinf.12	0.962
pwr_pin_haleu:kinf.14	0.971

- Pu-239 inventories also show high similarity with burnup. However, the similarity is stronger with burnup regime rather than burnup value.

Pearson Corr	pwr_pin_haleu:nucpu.pu239.2	pwr_pin_haleu:nucpu.pu239.6	pwr_pin_haleu:nucpu.pu239.14
pwr_pin:nucpu.pu239.2	0.970	0.970	0.951
pwr_pin:nucpu.pu239.3	0.964	0.972	0.965
pwr_pin:nucpu.pu239.6	0.939	0.962	0.977

- For accurate analysis, number of samples should be increased until analysis results converge.

```
=sampler
read parameters
...
' ----- TRITON input #1 -----
read case[pwr_pin]
...
end case
' ----- TRITON input #2 -----
read case[pwr_pin_haleu]
...
end case
' ----- SAMPLER responses -----
read response[nucpu]
type = origen_nuclides
...
read response[kinf]
type = grep
...
end response
' ----- SAMPLER analysis -----
read analysis [similar_k]
type = pearson_corr
targets = pwr_pin_haleu:kinf(0) pwr_pin_haleu:kinf(6)
.....
sources = pwr_pin:kinf(6) end
end analysis
read analysis [similar_iso]
type = pearson_corr
targets = pwr_pin:nucpu.pu239(2) ... end
sources = pwr_pin_haleu:nucpu.pu239(2) ... end
end analysis
end
```

Summary



Summary

- Sampler's key features are:
 - Propagation of model input parameter and nuclear data uncertainties to results of any SCALE sequence
 - Streamlined sample input file generation
 - Support of multiple distributions and consideration of dependencies for input parameter uncertainties
 - Propagation of cross section, fission yield, decay data, and kinetics data uncertainties
 - Analysis of results covering standard and advanced statistical metrics
 - Analysis results provided in formats for visualization in Fulcrum and for further post-processing with external tools
- Incoming new features in SCALE 7:
 - New distribution type “custom_values” in VARIABLE block
 - Target value search
 - Advanced similarity assessment