WPEC 2016 meetings
OECD/NEA Paris,
10–11 May 2016

XGPT: Sensitivity, Uncertainty Quantification, and Data Assimilation using Monte Carlo

Manuele Aufiero & Massimiliano Fratoni

"dreaming" about continuous-energy cross sections adjustment
thanks to:

G. Palmiotti and M. Salvatores

Generalized Perturbation Theory (GPT)

Available in deterministic codes from the 60s (Gandini, 1967?)

GPT in Serpent not discussed in details. Presented here at the 2014 JEFF WEEK

You can have a look at **Annals of Nuclear Energy, 85, 2015.** *A collision history...*

eXtended Generalized Perturbation Theory (XGPT)

Newly developed. Only possible in MC codes.

Continuous energy approach. No multi-group discretization.

First results of XS adjustment via XGPT

Is it possible to produce adjusted continuous energy XS?

Generalized Perturbation Theory capabilities

Effect of a perturbation of the parameter x on the response R :

$$S_x^R \equiv \frac{dR/R}{dx/x}$$

Considered response functions:

$R = k_{\text{eff}}$ Effective multiplication factor

$R = \frac{\langle \Sigma_1, \phi \rangle}{\langle \Sigma_2, \phi \rangle}$ Reaction rate ratios

$R = \frac{\langle \phi^\dagger, \Sigma_1 \phi \rangle}{\langle \phi^\dagger, \Sigma_2 \phi \rangle}$ Bilinear ratios (Adjoint-weighted quantities)

$R = \frac{E[e_1]}{E[e_2]}$ Something else

Berkeley
UNIVERSITY OF CALIFORNIA

Generalized response functions

If the quantity R can be estimated as the ratio of two generic Monte Carlo responses

$$R = \frac{E\left[e_1\right]}{E\left[e_2\right]}$$

the sensitivity coefficient of R with respect to x can be obtained as:

$$S_x^R = \frac{COV\left[e_1\ ,\ \sum^{history}\left(ACC_x - REJ_x\right)\right]}{E\left[e_1\right]} - \frac{COV\left[e_2\ ,\ \sum^{history}\left(ACC_x - REJ_x\right)\right]}{E\left[e_2\right]}$$

This is a continuous energy estimator but...

Unfortunately, energy discretization is required

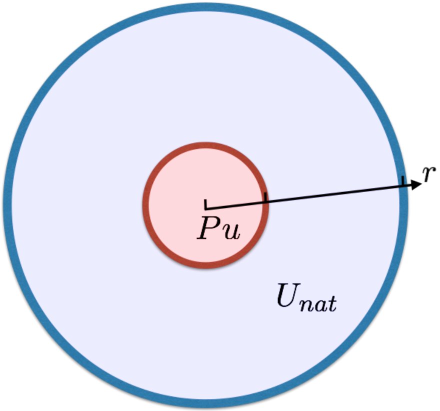
In continuous energy Monte Carlo transport, the probability of occurrence of a collision at the exact energy E is **ZERO**

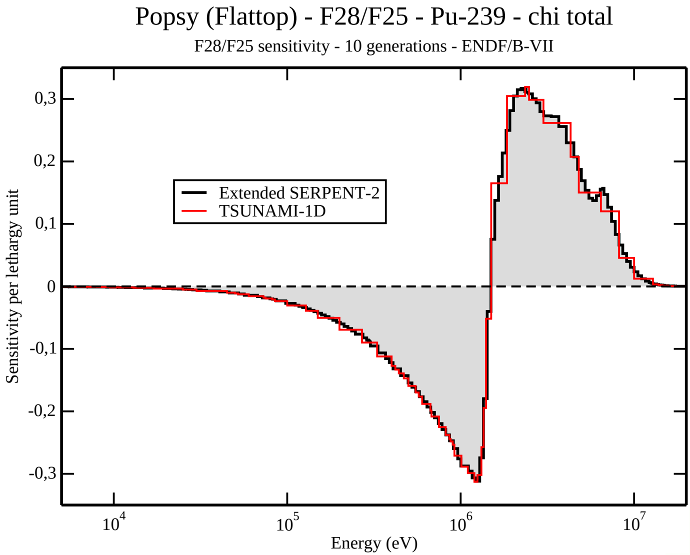
For this reason, energy-resolved sensitivity profiles are obtained (as in deterministic codes) by calculating group-wise integrals of S_{Σ}^{R}:

$$S_{\Sigma,g}^{R} = \int_{E_g}^{E_{g+1}} S_{\Sigma}^{R}(E)\, dE$$

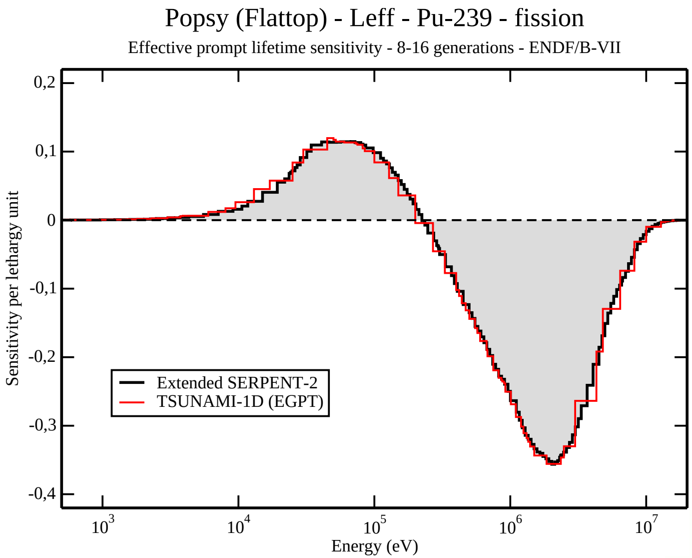
This is easily obtained by scoring the collisions occurred at energies between E_g and E_{g+1}.

Flattop-Pu




Popsy (Flattop) - F28/F25 - Pu-239 - chi total
F28/F25 sensitivity - 10 generations - ENDF/B-VII
Extended SERPENT-2
TSUNAMI-1D
Sensitivity per lethargy unit
0,3
0,2
0,1
0
-0,1
-0,2
-0,3
10^4
10^5
10^6
10^7
Energy (eV)

Generalized sensitivities: a couple of examples

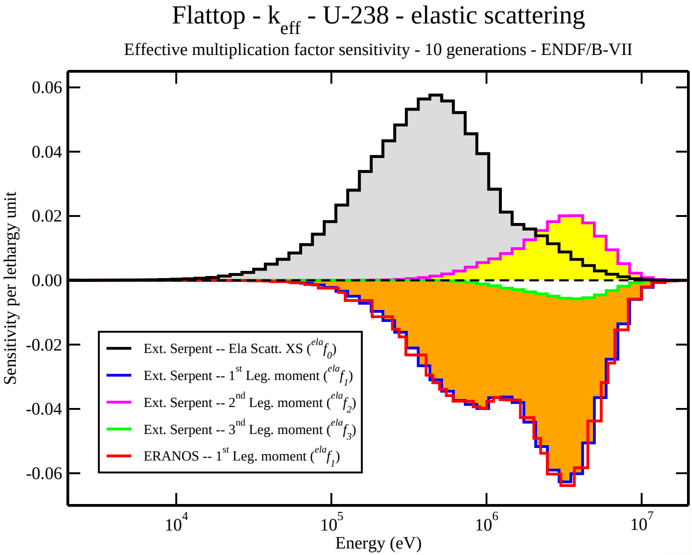


No bi-linear ratio sensitivity available in Scale (?)...
Adopting EGPT for comparison against Serpent

$$S_x^{\ell_{\text{eff}}} = \frac{\partial \left(\dfrac{\partial k_{\text{eff}} / k_{\text{eff}}}{\partial a_{1/v}} \right)}{\partial x / x} \Bigg/ \frac{\partial k_{\text{eff}} / k_{\text{eff}}}{\partial a_{1/v}}$$

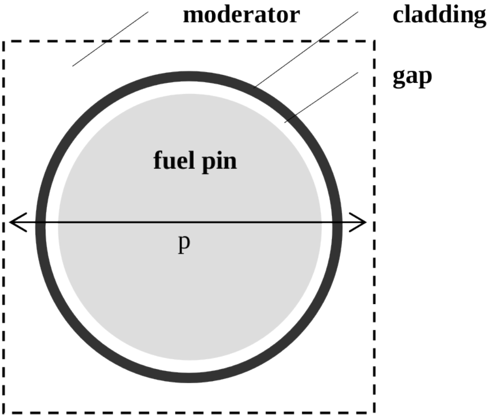
$$S_x^{\ell_{\text{eff}}} = \frac{\partial \left(\dfrac{\partial k_{\text{eff}} / k_{\text{eff}}}{\partial x / x} \right)}{\partial a_{1/v}} \Bigg/ \frac{\partial k_{\text{eff}} / k_{\text{eff}}}{\partial a_{1/v}} = \frac{\dfrac{\partial S_x^{k_{\text{eff}}}}{\partial a_{1/v}}}{\dfrac{\partial k_{\text{eff}} / k_{\text{eff}}}{\partial a_{1/v}}}$$

$$S_x^{\ell_{\text{eff}}} \simeq \frac{{}^*S_x^{k_{\text{eff}}} - S_x^{k_{\text{eff}}}}{\dfrac{{}^*k_{\text{eff}} - k_{\text{eff}}}{k_{\text{eff}}}}$$

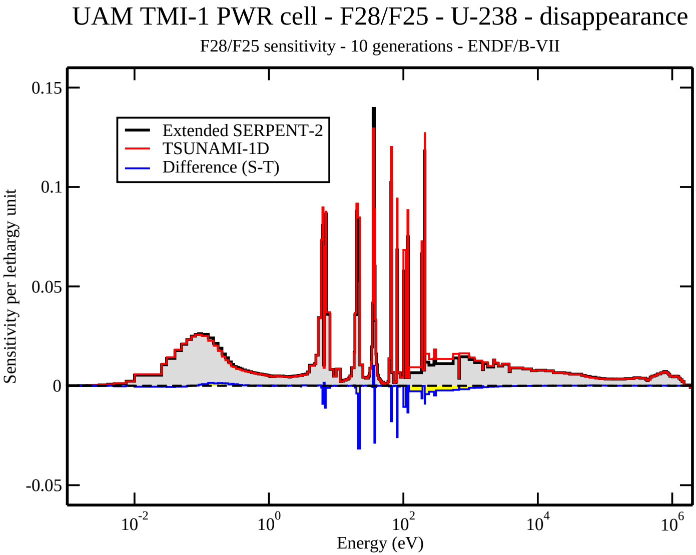


Flattop - k_eff - U-238 - elastic scattering
Effective multiplication factor sensitivity - 10 generations - ENDF/B-VII
Sensitivity per lethargy unit
0.06
0.04
0.02
0.00
-0.02
-0.04
-0.06
10^4
10^5
10^6
10^7
Energy (eV)
Ext. Serpent -- Ela Scatt. XS (ela f0)
Ext. Serpent -- 1st Leg. moment (ela f1)
Ext. Serpent -- 2nd Leg. moment (ela f2)
Ext. Serpent -- 3rd Leg. moment (ela f3)
ERANOS -- 1st Leg. moment (ela f1)

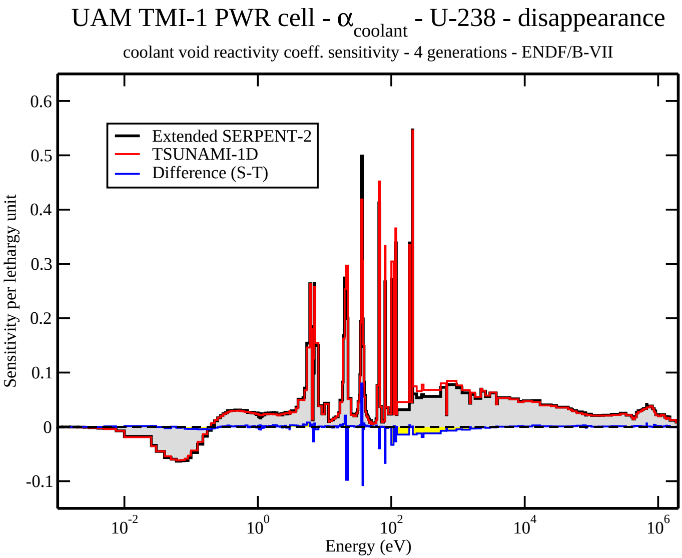
PWR pin cell



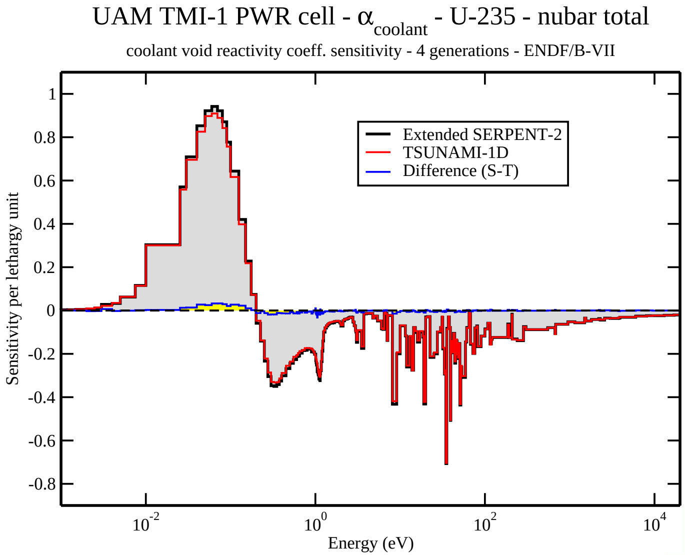
Generalized sensitivities: a couple of examples



Generalized sensitivities: a couple of examples



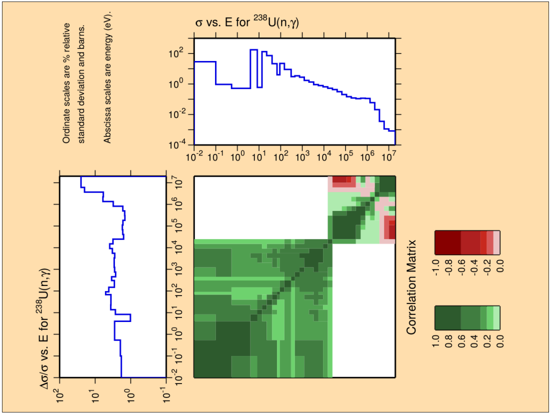
Generalized sensitivities: a couple of examples



Uncertainty propagation

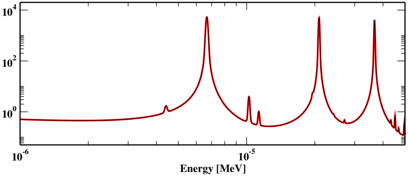
First-order uncertainty propagation formula (sandwich rule)

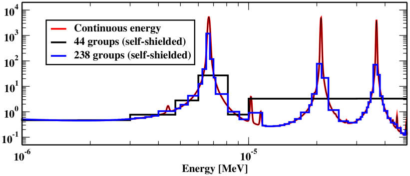
$$Var\left[R\right] = \underline{S_x^R}\; Cov\left[\underline{x}\right] \left(\underline{S_x^R}\right)^T$$



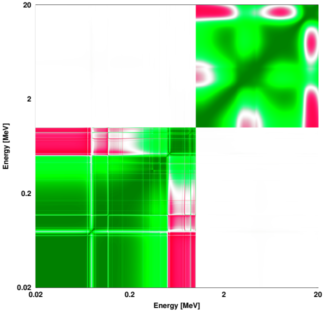
Continuous energy vs. multi-group

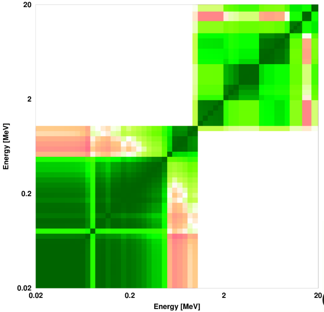
Cross sections





Covariance matrices





UNIVERSITY OF CALIFORNIA

Continuous energy transport, multi-group sensitivity/uncert. quantification/adjustment

- Choosing the multi-group energy grid is not an easy task (for both covariance matrices and sensitivity profiles)
- Special treatment required for self-shielding
- The efficiency of any Monte Carlo sensitivity estimator degrades quickly with finer discretizations

Continuous energy transport, multi-group sensitivity/uncert. quantification/adjustment

- Choosing the multi-group energy grid is not an easy task (for both covariance matrices and sensitivity profiles)
- Special treatment required for self-shielding
- The efficiency of any Monte Carlo sensitivity estimator degrades quickly with finer discretizations
- Multi-group sensitivity means multi-group adjusted XS
- What about higher moments of the uncertain responses?
- Multi-group & sandwich rule were the obvious choices for deterministic codes. Still the best choice for Monte Carlo?

Continuous-energy function sensitivity approach

The eXtended Generalized Perturbation Theory (XGPT) makes use of continuous-energy function sensitivity:

- No need for multi-group discretization of the sensitivity profiles or the covariance matrices
- Direct adoption of continuous energy covariances and resonance parameters covariances
- The new method has been implemented and tested in Serpent
- Uncertainty propagation tested against Total Monte Carlo
- Continuous energy XS adjustment under investigation

Continuous-energy function sensitivity approach

Continuous energy uncertainty propagation formula:

$$Var\left[R\right] = \int_{E_{min}}^{E_{max}} \int_{E_{min}}^{E_{max}} S_{\Sigma}^{R}\left(E\right) \cdot COV\left[\Sigma(E)\,,\Sigma(E')\right] \cdot S_{\Sigma}^{R}\left(E'\right)\, dE\, dE'$$

It is hard to solve double integrals with Monte Carlo transport.

- Legacy approach: multi-group discretization
  + sum over **bin-averaged** sensitivities

- New approach: eigenvalue expansion
  + sum over **continuous-energy** integrated sensitivities

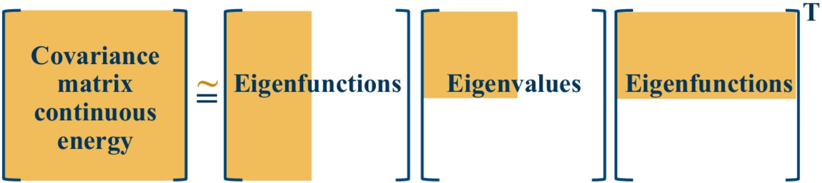
Eigenvalue decomposition of the covariance matrix

$$COV\left[\Sigma(E)\,,\Sigma(E')\right] = \sum_{j=1}^{\infty} U_j(E)\cdot V^j \cdot U_j(E')$$

$$\left[\begin{array}{c}\textbf{Covariance}\\ \textbf{matrix}\\ \textbf{continuous}\\ \textbf{energy}\end{array}\right] = \left[\textbf{Eigenfunctions}\right]\left[\textbf{Eigenvalues}\right]\left[\textbf{Eigenfunctions}\right]^{\mathbf{T}}$$

Eigenvalue decomposition of the covariance matrix

$$COV\left[\Sigma(E)\,,\Sigma(E')\right] \sim \sum_{j=1}^{n} U_j(E)\cdot V^j \cdot U_j(E')$$



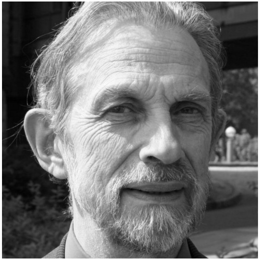
Continuous energy uncertainty propagation formula:

$$Var\left[R\right] = \int_{E_{min}}^{E_{max}} \int_{E_{min}}^{E_{max}} S_{\Sigma}^{R}\left(E\right) \cdot COV\left[\Sigma(E), \Sigma(E')\right] \cdot S_{\Sigma}^{R}\left(E'\right) \, dE \, dE'$$

$$COV\left[\Sigma(E), \Sigma(E')\right] \sim \sum_{j=1}^{n} U_j(E) \cdot V^j \cdot U_j(E')$$

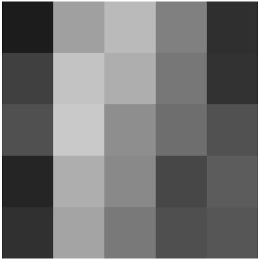
$$Var\left[R\right] \sim \sum_{j=1}^{n} V^j \cdot \left(\int_{E_{min}}^{E_{max}} U_j\left(E\right) \cdot S_{\Sigma}^{R}\left(E\right) \, dE \right)^2$$

$$Var\left[R\right] \sim \sum_{j=1}^{n} V^j \cdot \left(S_{U_j}^{R}\right)^2$$



Original
image

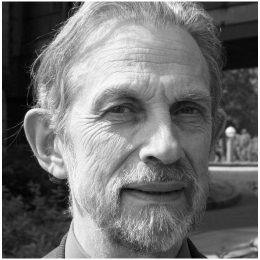
SVD/POD
5 basis functions



Multi-group
5 energy groups

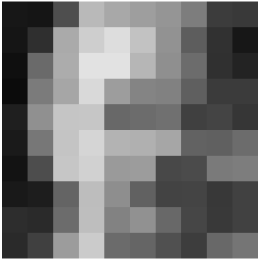
```
A=imread("massimo.png"); [A, map]=gray2ind(A,255);
[U, S, V]=svd(A);
A_SVD_5 = U(:,1:5) * S(1:5,1:i) * V(:,1:5)′;
imwrite(A_SVD_5, gray(255), "massimo_5.png");
```

Singular Value Decomposition



Original image

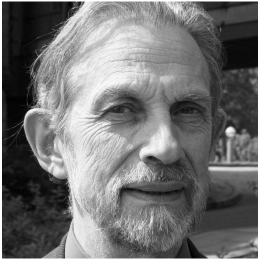
SVD/POD 10 basis functions



Multi-group 10 energy groups

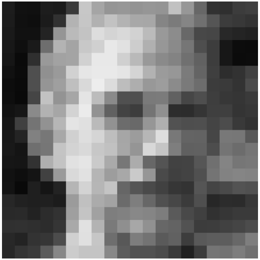
```
A=imread("massimo.png"); [A, map]=gray2ind(A,255);
[U, S, V]=svd(A);
A_SVD_10 = U(:,1:10) * S(1:10,1:i) * V(:,1:10)′;
imwrite(A_SVD_10, gray(255), "massimo_10.png");
```

Singular Value Decomposition



Original
image

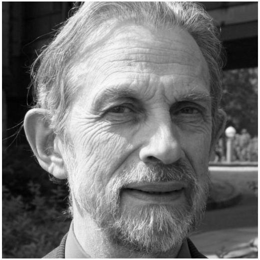
SVD/POD
20 basis functions



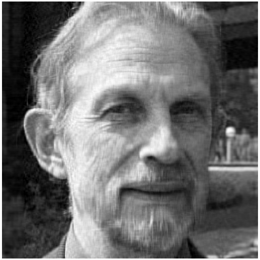
Multi-group
20 energy groups

```
A=imread("massimo.png"); [A, map]=gray2ind(A,255);
[U, S, V]=svd(A);
A_SVD_20 = U(:,1:20) * S(1:20,1:i) * V(:,1:20)';
imwrite(A_SVD_20, gray(255), "massimo_20.png");
```

Singular Value Decomposition



Original image

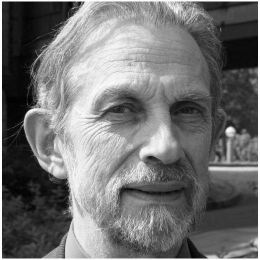


SVD/POD
40 basis functions

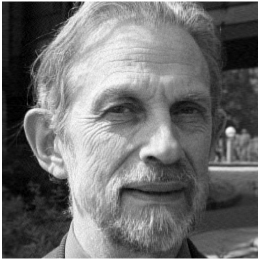
Multi-group
40 energy groups

```
A=imread("massimo.png"); [A, map]=gray2ind(A,255);
[U, S, V]=svd(A);
A_SVD_40 = U(:,1:40) * S(1:40,1:i) * V(:,1:40)';
imwrite(A_SVD_40, gray(255), "massimo_40.png");
```

Singular Value Decomposition



Original
image



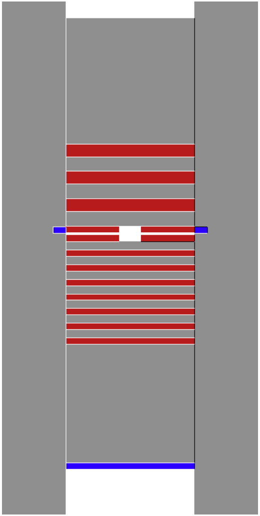
SVD/POD
80 basis functions

Multi-group
80 energy groups

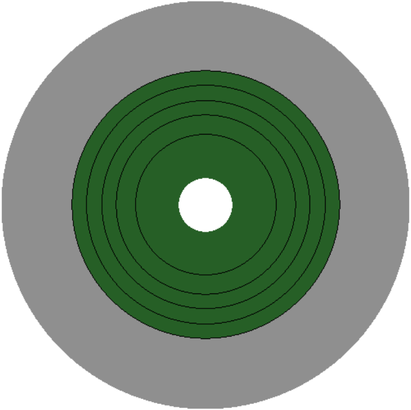
```
A=imread("massimo.png"); [A, map]=gray2ind(A,255);
[U, S, V]=svd(A);
A_SVD_80 = U(:,1:80) * S(1:80,1:i) * V(:,1:80)′;
imwrite(A_SVD_80, gray(255), "massimo_80.png");
```

Two simple case studies

HMF-64



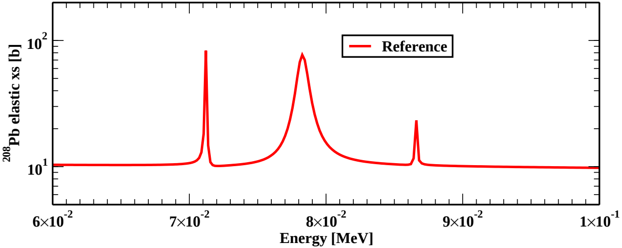
PMF-35

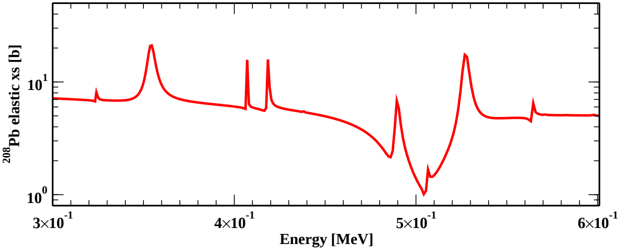


Random cross sections

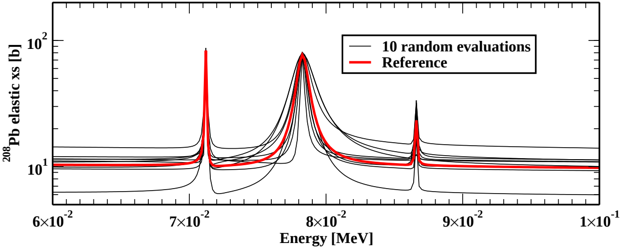
- 3000 different ^{208}Pb ENDF files from TENDL-2013
- Random MF2-MT151 (resonances), MF3-MT1, MF3-MT2 (elastic), MF3-MT51-58,91 (inelastic), and MF3-MT102 (n, γ) processed with NJOY
- 3000 ACE files with random cross sections
- The random continuous energy XS reflect the **UNCERTAINTIES** and their **CORRELATIONS** (according to TENDL-2013)
- Continuous-energy covariances reconstructed from random XS

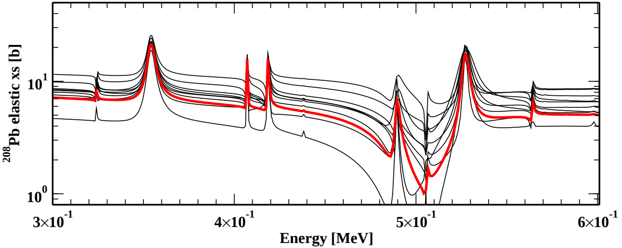
Random cross sections



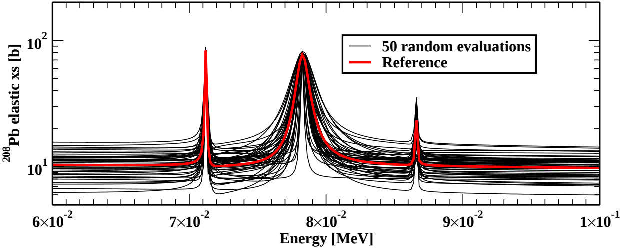


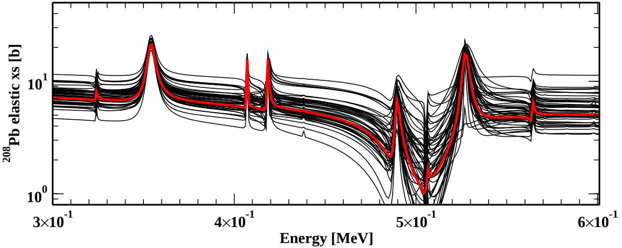
Random cross sections





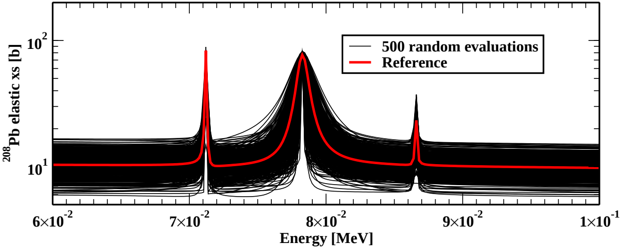
Random cross sections

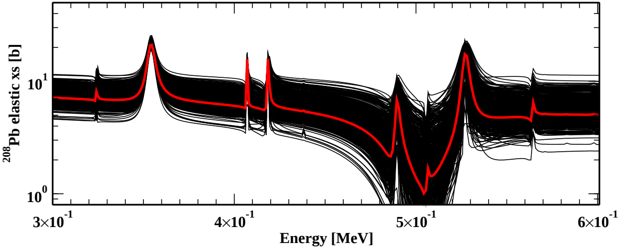




Berkeley
UNIVERSITY OF CALIFORNIA

Random cross sections



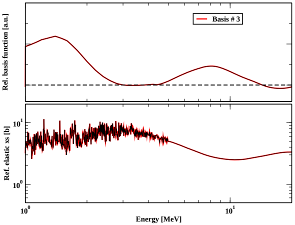


Berkeley UNIVERSITY OF CALIFORNIA

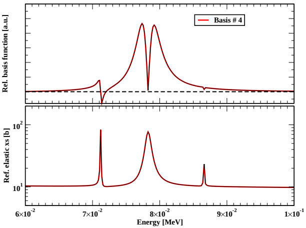
We want a set of orthogonal basis functions $\boldsymbol{b_{\Sigma,j}}$ so that:

$$\widetilde{\Sigma}_i(E) = \Sigma_0(E) \cdot \left(1 + \sum_{j=1}^{n} \alpha_i^j \cdot b_{\Sigma,j}(E) \right)$$

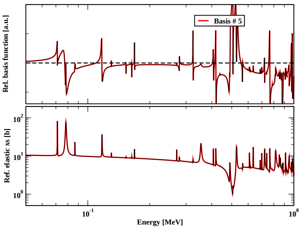
3 basis functions from the POD of ^{208}Pb (n, ela)



Basis # 3

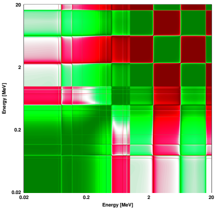


Basis # 4

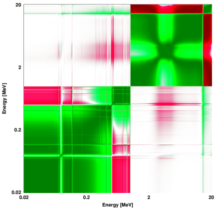


Basis # 5

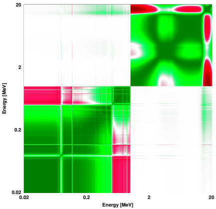
Correlation matrices after POD and reconstruction



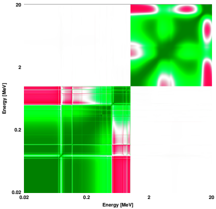
$n = 2$



$n = 5$

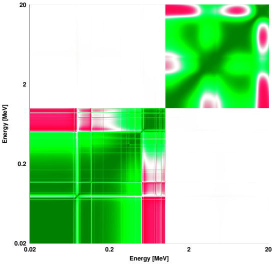


$n = 10$

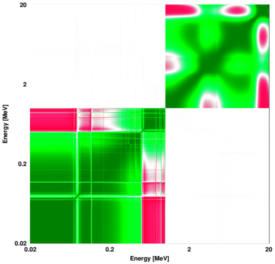


$n = 20$

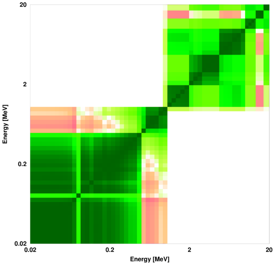
Proper Orthogonal Decomposition of Nuclear Data



Continuous
Energy



SVD/POD
20 basis functions



Multi-group
>100 ene g.

XGPT+POD: calculating sensitivities to CE basis functions

We need to calculate the effect on the response $\boldsymbol{R}$ due to a perturbation on $\boldsymbol{\Sigma}$ equal to $\boldsymbol{b_{\Sigma,j}}$

$$S^R_{b_{\Sigma,j}} = \frac{dR/R}{d\,b_{\Sigma,j}} = \int_{E_{min}}^{E_{max}} b_{\Sigma,j}\,(E) \cdot S^R_{\Sigma}\,(E)\,dE$$

XGPT+POD: calculating sensitivities to CE basis functions

Using the collision history approach, $S^R_{b_{\Sigma,j}}$ can be estimated from the covariance between the terms of R and the collisions weighted by $b_{\Sigma,j}(E)$

$$S^R_{b_{\Sigma,j}} = \frac{COV\left[e_1 \,,\, \sum^{history} G_{b_{\Sigma,j}}\right]}{E\left[e_1\right]} - \frac{COV\left[e_2 \,,\, \sum^{history} G_{b_{\Sigma,j}}\right]}{E\left[e_2\right]}$$

Berkeley
UNIVERSITY OF CALIFORNIA

XGPT+POD: uncertainty propagation

From the Proper Orthogonal Decomposition of Nuclear data...

$$\Sigma_i(E) \simeq \widetilde{\Sigma}_i(E) = \Sigma_0(E) \cdot \left(1 + \sum_{j=1}^{n} \alpha_i^j \cdot b_{\Sigma,j}(E)\right)$$

...and the basis functions sensitivity coefficients $\boldsymbol{S^R_{b_{\Sigma,j}}}$, we can approximate the response function $\boldsymbol{R_{\Sigma_i}}$ for each random XS $\boldsymbol{\Sigma_i}$

$$R_{\Sigma_i} \simeq \widetilde{R_{\Sigma_i}} = R_{\Sigma_0} \cdot \left(1 + \sum_{j=1}^{n} \alpha_i^j \cdot S^R_{b_{\Sigma,j}}\right)$$

Berkeley UNIVERSITY OF CALIFORNIA

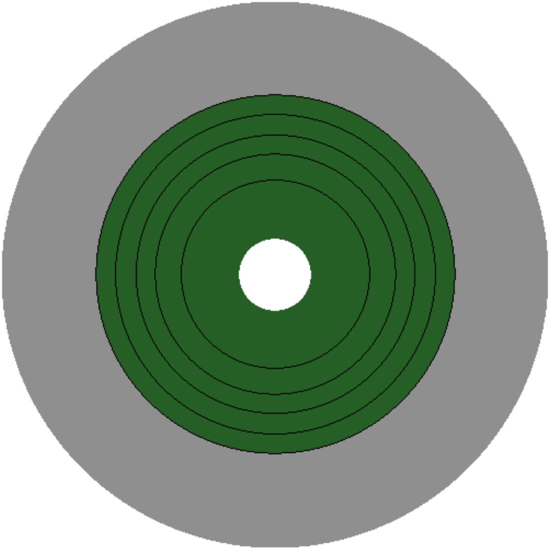
Estimating the k_{eff} distribution in the simple case study

$$k_{\text{eff}_{\Sigma_i}} \simeq \widetilde{k_{\text{eff}}}_{\Sigma_i} = k_{\text{eff}_{\Sigma_0}} \cdot \left(1 + \sum_{j=1}^{n} \alpha_i^j \cdot S_{b_{\Sigma,j}}^{k_{\text{eff}}}\right)$$

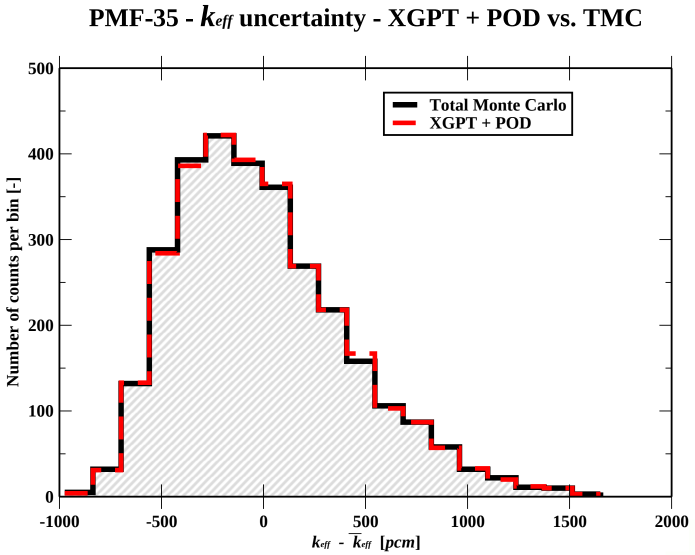
The k_{eff} for all the N (3000) random XS Σ_i were calculated in a single Serpent run (ACE file for Σ_0) with $n = 50$ bases

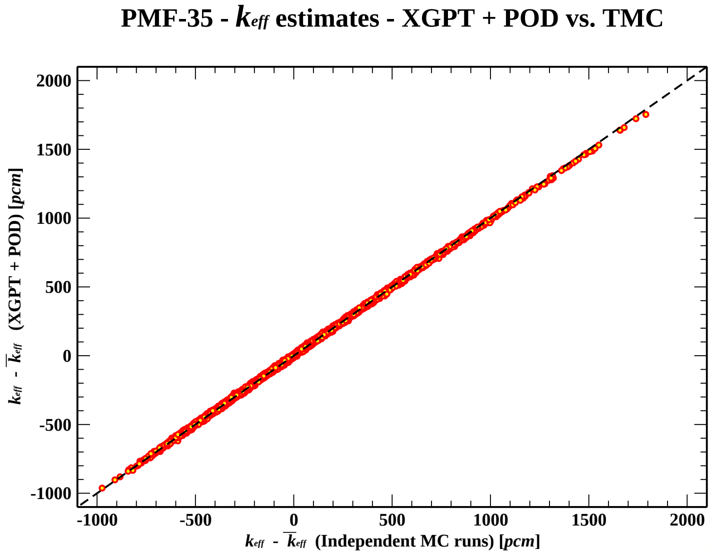
The XGPT+POD results are compared to TMC results (3000 separate Serpent runs)

PMF-35

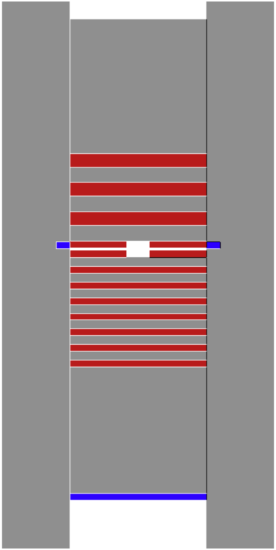


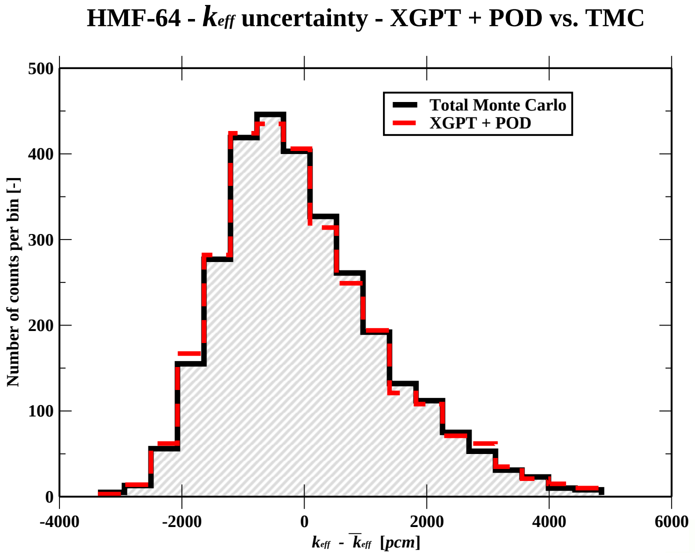
XGPT+POD: uncertainty propagation in PMF-35 – k_{eff}




PMF-35 - keff estimates - XGPT + POD vs. TMC
2000
1500
1000
500
0
-500
-1000
-1000
-500
0
500
1000
1500
2000
keff - keff (XGPT + POD) [pcm]
keff - keff (Independent MC runs) [pcm]

PMF-35




HMF-64 - keff uncertainty - XGPT + POD vs. TMC
Total Monte Carlo
XGPT + POD
Number of counts per bin [-]
0
100
200
300
400
500
-4000
-2000
0
2000
4000
6000
keff - keff [pcm]

XGPT+POD: uncertainty in **PMF-35 & HMF-64**

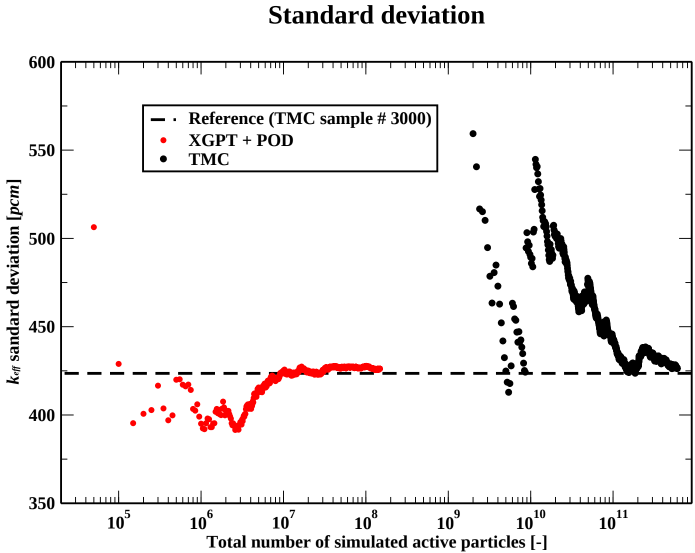
Table: Standard deviation, skewness and kurtosis of the *PMF-35* k_{eff} distribution from TENDL-2013 ^{208}Pb cross section data.

Methos	Standard deviation	skewness	kurtosis
TMC	426 *pcm*	0.81	3.62
XGPT	423 *pcm*	0.80	3.58

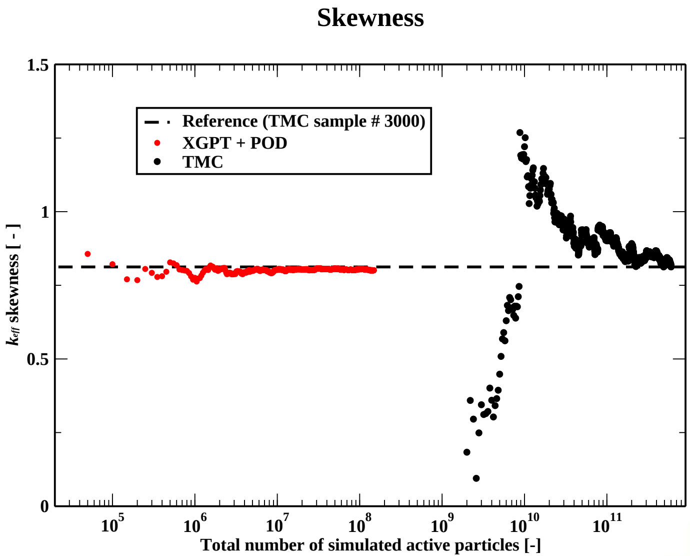
Table: Standard deviation, skewness and kurtosis of the *HMF-64* k_{eff} distribution from TENDL-2013 ^{208}Pb cross section data.

	Standard deviation	skewness	kurtosis
TMC	1326 *pcm*	0.74	3.49
XGPT	1371 *pcm*	0.81	3.65

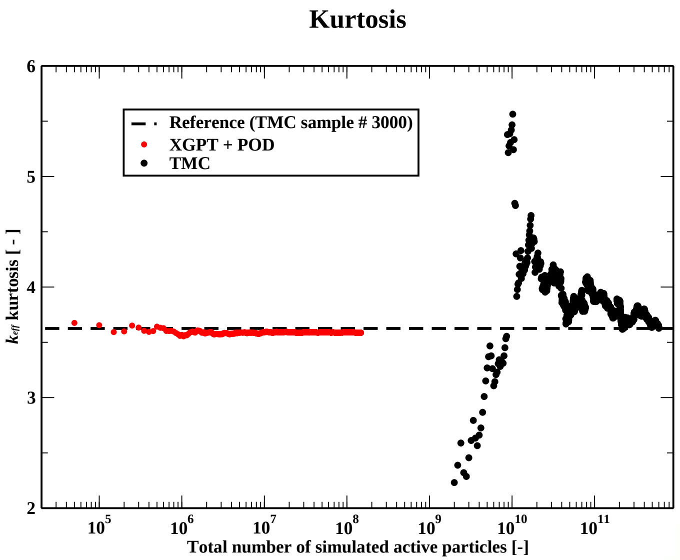
XGPT+POD: uncertainty propagation in PMF-35 – k_{eff}



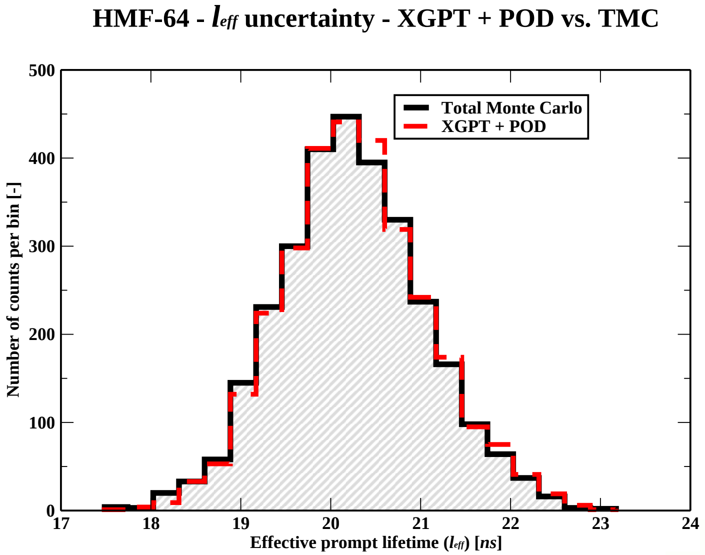
XGPT+POD: uncertainty propagation in PMF-35 – k_{eff}



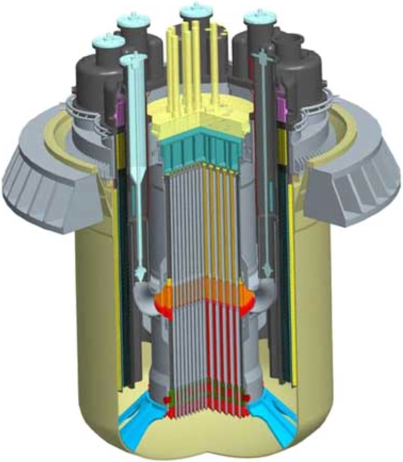
XGPT+POD: uncertainty propagation in PMF-35 – k_{eff}



XGPT+POD: uncertainty propagation in HMF-64 – ℓ_{eff}

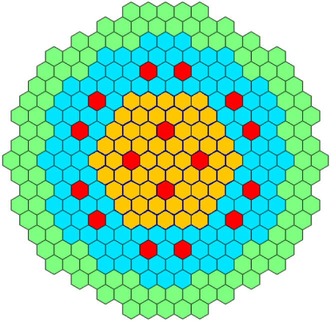


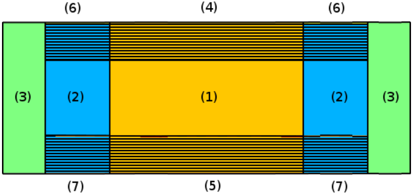
ALFRED, is a small-size (300MWth) pool-type LFR.



Advanced Lead Fast Reactor European Demonstrator

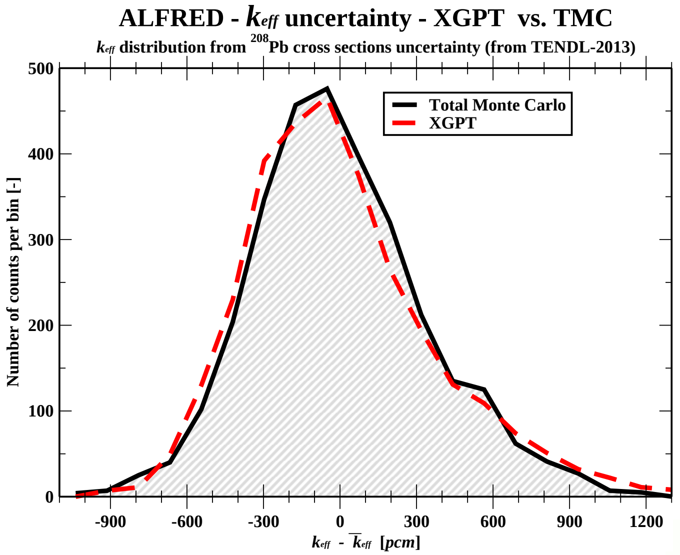
Developed within the European FP7 LEADER project





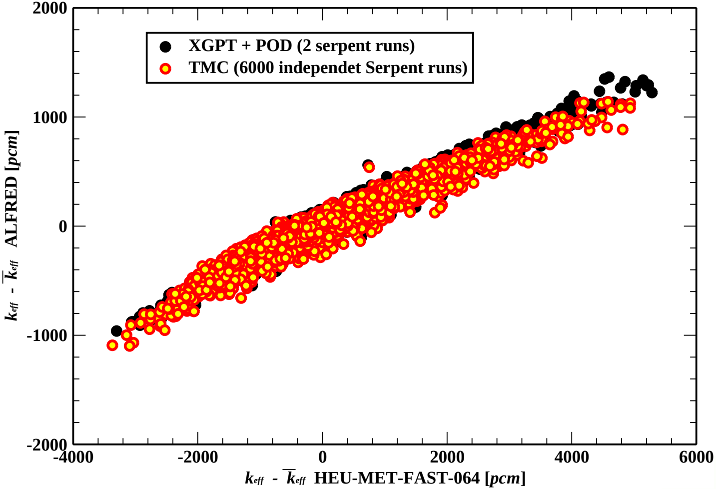
171 FAs are subdivided into two radial zones with different plutonium fractions

k_{eff} uncertainty in ALFRED

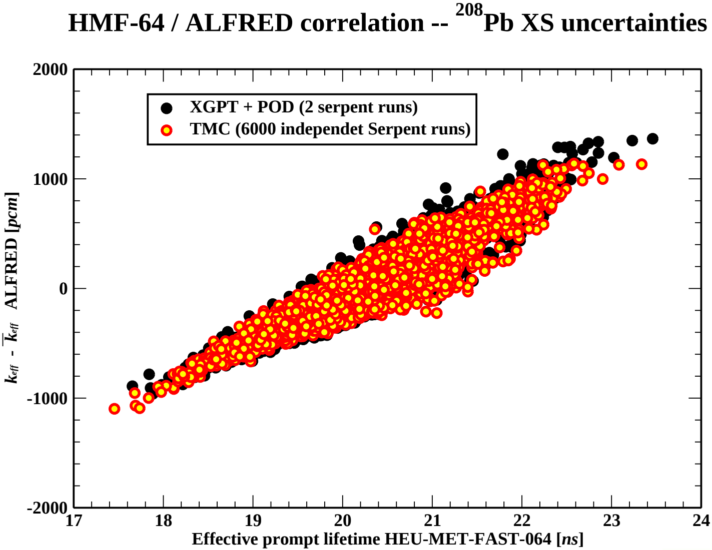


Representativity study ALFRED vs. HMF-64

HMF-64 / ALFRED correlation -- ^{208}Pb XS uncertainties



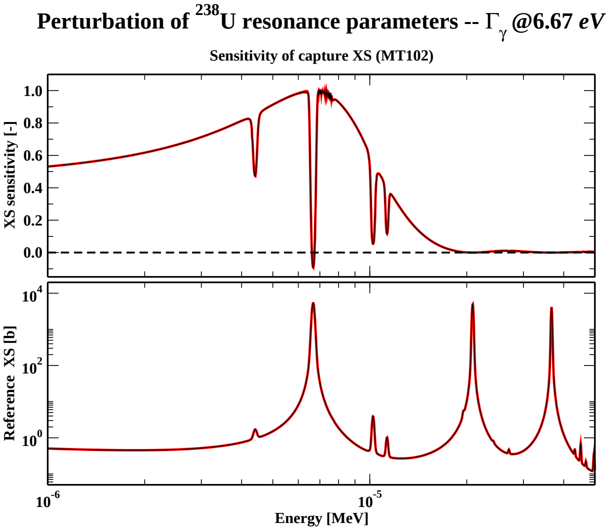
Representativity study ALFRED vs. HMF-64



Video time

Cross section sensitivity to resonance parameters

Cross sections derivatives are calculated numerically via NJOY



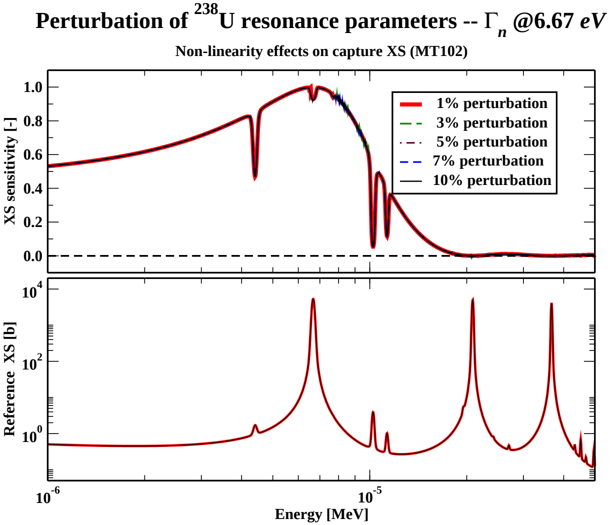
Definition of sensitivity coefficient: $S_x^R \equiv \frac{dR/R}{dx/x}$

$$S_{\Gamma_\gamma}^{k_{\text{eff}}} = \int S_{\sigma_{capture}}^{k_{\text{eff}}}(E) \cdot S_{\Gamma_\gamma}^{\sigma_{capture}}(E) \cdot dE + \int S_{\sigma_{elastic}}^{k_{\text{eff}}}(E) \cdot S_{\Gamma_\gamma}^{\sigma_{elastic}}(E) \cdot dE + \ldots$$

Multi-group discretization is usually introduced here...

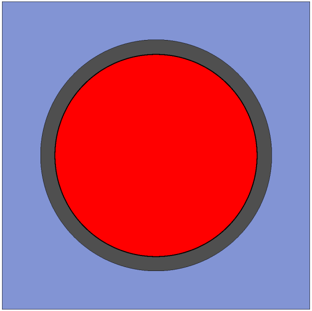
The new method avoids any discretization

Accuracy of the numerical derivative calculation via NJOY

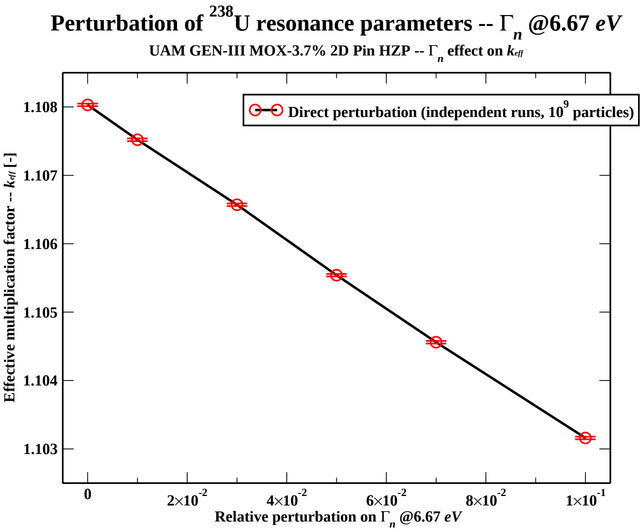


Material compositions and geometry specifications from:
Benchmarks for uncertainty analysis in modelling (UAM) for the design, operation and safety analysis of LWRs

Case: GEN-III PWR MOX 2D pin cell
Pu content in fuel 3.7%

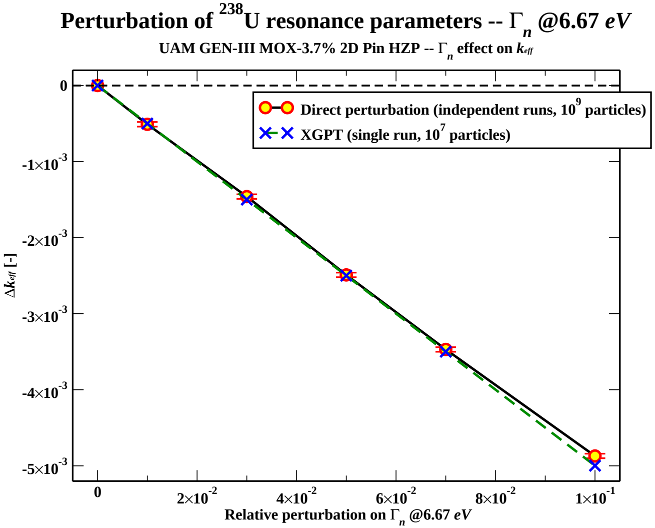


Verification against direct perturbation: ^{238}U



Berkeley UNIVERSITY OF CALIFORNIA

Verification against direct perturbation: ^{238}U



Berkeley
UNIVERSITY OF CALIFORNIA

Verification against direct perturbation: ^{238}U

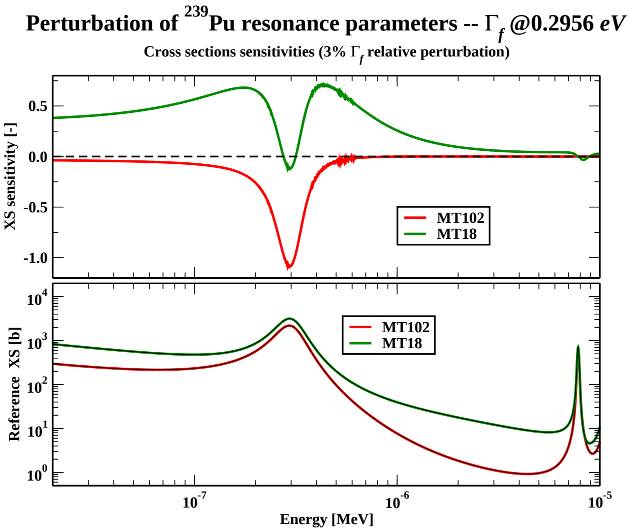
^{238}U @6.67eV

	Direct perturbation	GPT
$S^{k_{\text{eff}}}_{\Gamma_\gamma}$	-4.603×10^{-2} $\pm 9.9 \times 10^{-4}$	-4.469×10^{-2} $\pm 1.4 \times 10^{-4}$
$S^{k_{\text{eff}}}_{\Gamma_n}$	-4.392×10^{-2} $\pm 9.9 \times 10^{-4}$	-4.512×10^{-2} $\pm 1.6 \times 10^{-4}$

- Sensitivities are very large
- GPT is more efficient: 10^7 vs 10^9 particles, smaller err.
- All GPT sensitivities calculated in a single run

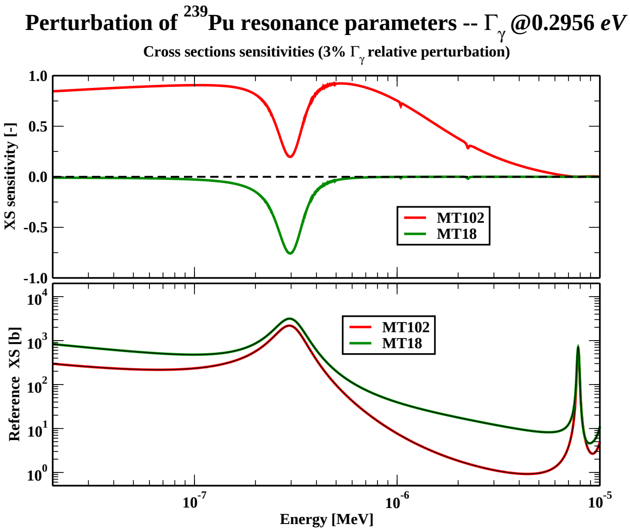
Berkeley
UNIVERSITY OF CALIFORNIA

Verification against direct perturbation: ^{239}Pu

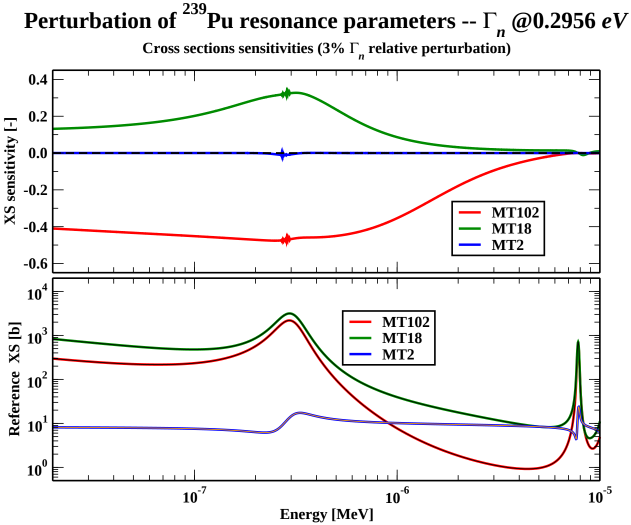


Berkeley
UNIVERSITY OF CALIFORNIA

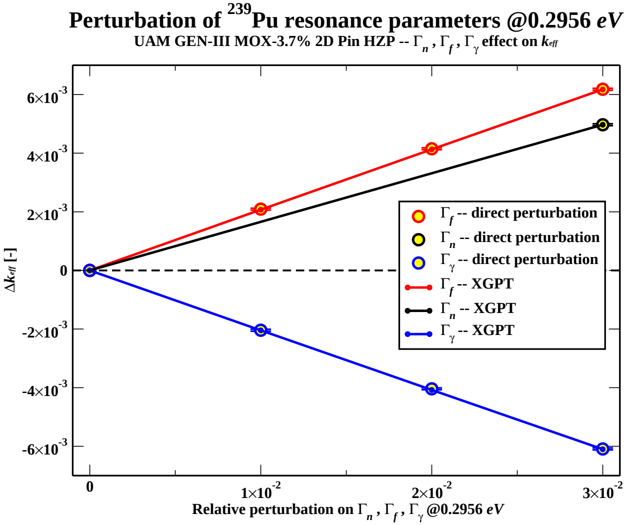
Verification against direct perturbation: ^{239}Pu



Verification against direct perturbation: ^{239}Pu



Verification against direct perturbation: ^{239}Pu



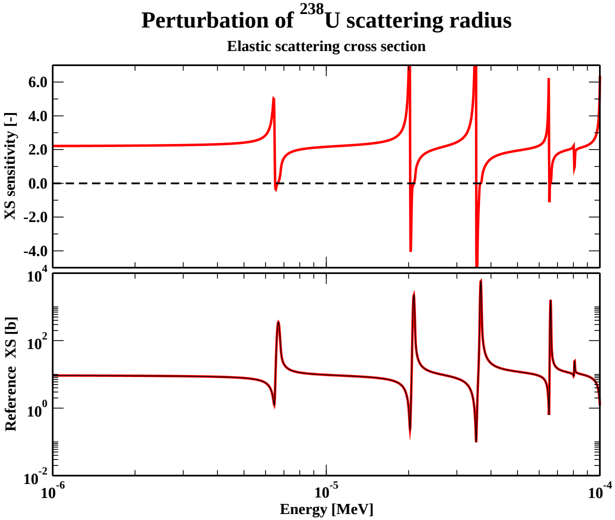
Verification against direct perturbation: ^{239}Pu

^{239}Pu @0.295eV

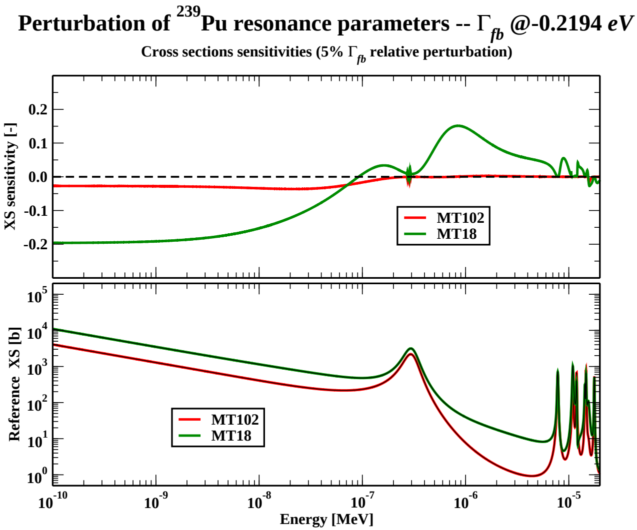
	Direct perturbation	GPT
$S^{k_{\text{eff}}}_{\Gamma_\gamma}$	-1.832×10^{-2}	-1.835×10^{-2}
	$\pm 9.9 \times 10^{-4}$	$\pm 3.8 \times 10^{-4}$
$S^{k_{\text{eff}}}_{\Gamma_n}$	1.495×10^{-2}	1.495×10^{-2}
	$\pm 9.9 \times 10^{-4}$	$\pm 1.6 \times 10^{-4}$
$S^{k_{\text{eff}}}_{\Gamma_f}$	1.859×10^{-2}	1.857×10^{-2}
	$\pm 9.9 \times 10^{-4}$	$\pm 4.1 \times 10^{-4}$

- Sensitivities are very large
- GPT is more efficient: 10^7 vs 10^9 particles, smaller err.
- All GPT sensitivities calculated in a single run

Fancy resonance parameters (scattering radius)



Fancy resonance parameters (negative energy resonances)



Continuous energy adjustment

1 Generate continuous-energy basis functions

- **1.1.a** SVD of the continuous-energy covariance matrices
- **1.1.b** POD of the random ENDF/random nuclear model parameters
- **1.2.a** XS derivatives for resonance parameters, scatt. radius & co.
- **1.2.b** Include resonances in the continuous-energy covariances

2 Project the uncertainties on these bases (MF32-33)

3 Run Serpent-XGPT once for each system

- **3.1.a** Get the uncertainty for each response function

$$Var\left[R\right] \sim \sum_{j=1}^{n} V^{j} \cdot \left(S^{R}_{U_j}\right)^{2}$$

- **3.1.b** Get the "uncertain distribution" of each response function

$$R_{\Sigma_i} \simeq \widetilde{R_{\Sigma_i}} = R_{\Sigma_0} \cdot \left(1 + \sum_{j=1}^{n} \alpha_i^j \cdot S^{R}_{b_{\Sigma,j}}\right)$$

Berkeley
UNIVERSITY OF CALIFORNIA

4 Send C/E, $Var[R]$, V^j, and $S^R_{U_j}$ to Pino

5 Ask him to solve the GLLS problem and send you: ${}^{adjusted}COV[\underline{\mathbf{V}}, \underline{\mathbf{V}}]$, $\Delta\alpha^j_U$

6 Reconstruct the adjusted Σ

$$ {}^{adj}\Sigma(E) \simeq {}^{prior}\Sigma(E) \cdot \left(1 + \sum_{j=1}^{n} \Delta\alpha^j_U \cdot U_j(E)\right) $$

7 Reconstruct the adjusted $COV[\Sigma, \Sigma]$

- ${}^{prior}COV[\Sigma(E), \Sigma(E')] \sim \sum_{j=1}^{n} U_j(E) \cdot V^j \cdot U_j(E')$
- ${}^{adj}COV[\Sigma(E), \Sigma(E')] \sim \underline{\mathbf{U}}\ {}^{adj}COV[\underline{\mathbf{V}}, \underline{\mathbf{V}}]\ \underline{\mathbf{U}}^T$

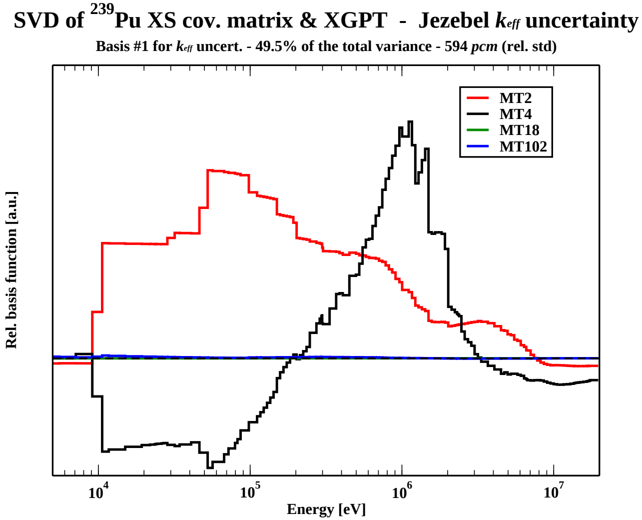
$$= \begin{bmatrix} U_1(E) & U_2(E) & \dots \end{bmatrix} COV[\underline{\mathbf{V}}, \underline{\mathbf{V}}] \begin{bmatrix} U_1(E) \\ U_2(E) \\ U_3(E) \\ \vdots \end{bmatrix}$$

Adjustment via XGPT: first attempt

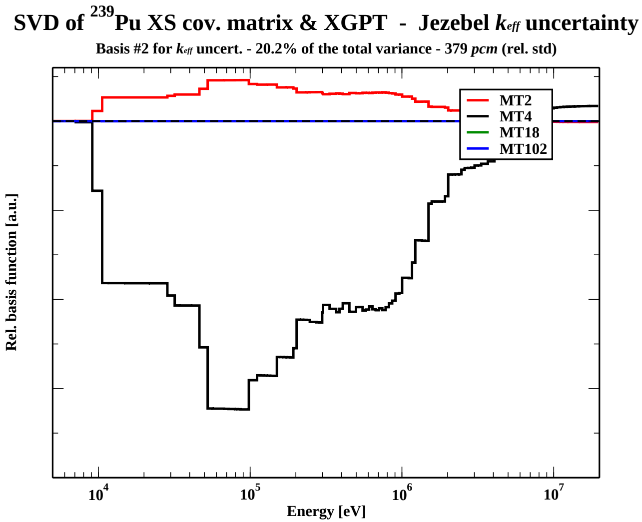
Very simple case study

- Only one system: Jezebel
- Only one response function: k_{eff}
- Only one isotope: ^{239}Pu
- Covariances from ENDF/B-VII.0 (multigroup!)

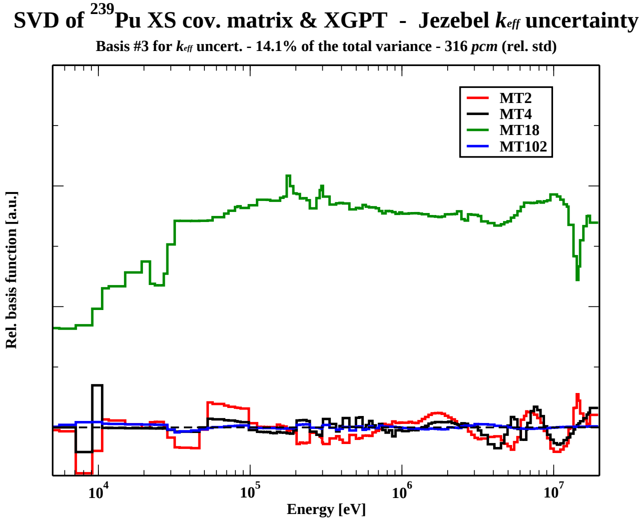
Adjustment via XGPT: first attempt



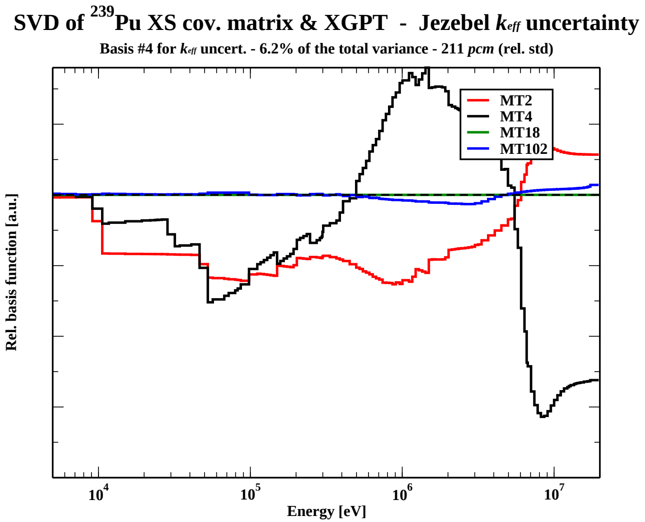
Adjustment via XGPT: first attempt



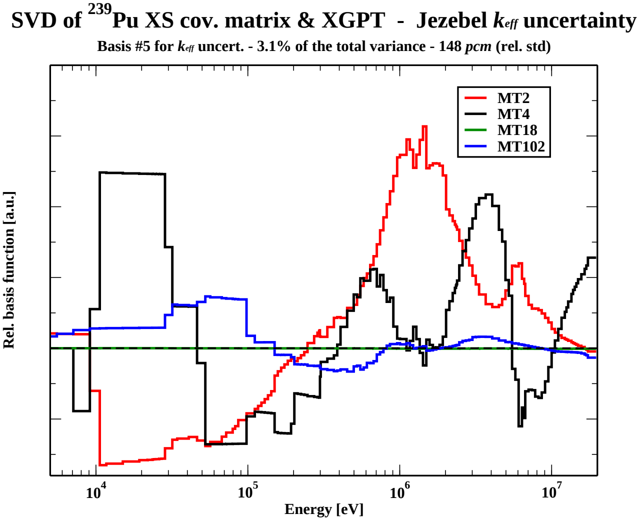
Adjustment via XGPT: first attempt



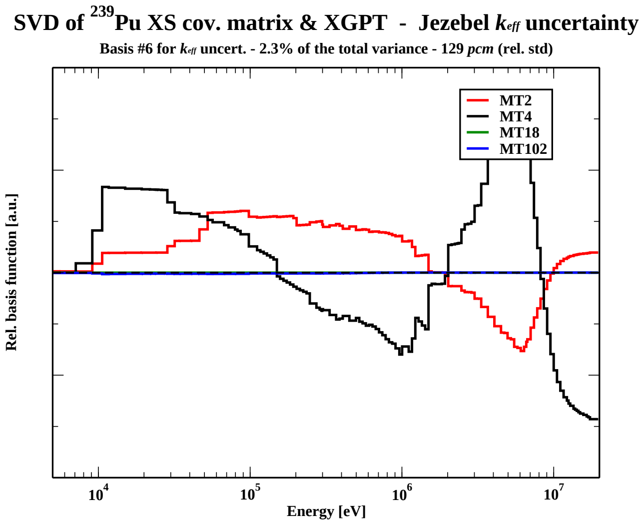
Adjustment via XGPT: first attempt



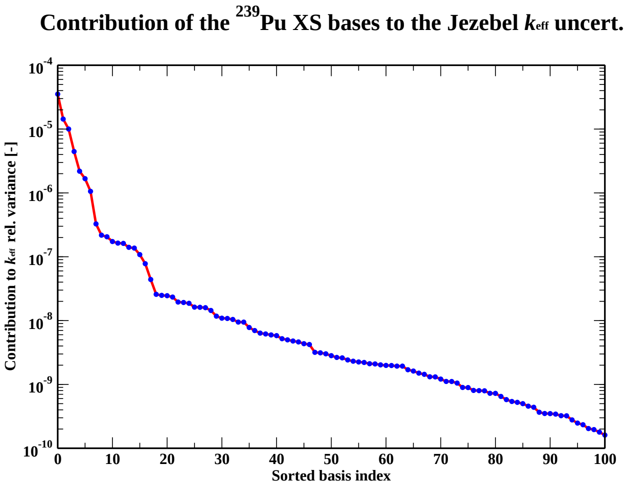
Adjustment via XGPT: first attempt



Adjustment via XGPT: first attempt



Adjustment via XGPT: first attempt



Adjustment via XGPT: first attempt (SQUADRA output)

(E-C)/C (%) BEFORE AND AFTER ADJUSTM.

#	EXPERIMENT	BEFORE	AFTER	CHANGE
1	JEZEBEL KEFF	0.014	0.001	-0.013

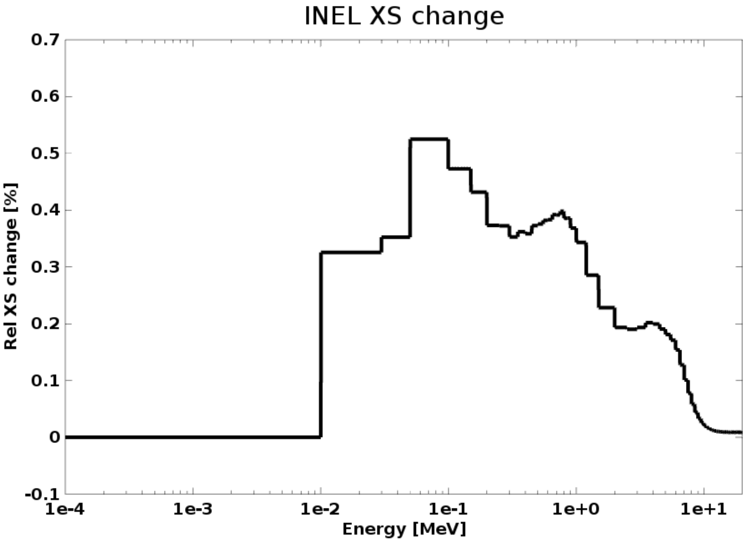
EXPERIMENT UNCERT. (%) BEFORE AND AFTER ADJUSTM.

#	EXPERIMENT	BEFORE	AFTER	CHANGE
1	JEZEBEL KEFF	0.950	0.196	-0.754

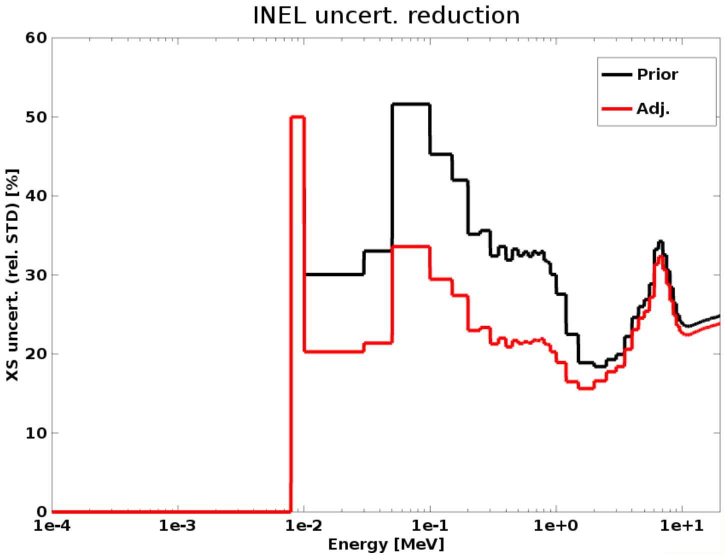
NUCL. DATA UNCERT. BEFORE AND AFTER ADJUSTM.

NUCLEAR DATA		NUC. DATA CHANGE	BEFORE	AFTER	CHANGE
EIGENV_INEL	1	-14.1	1274.8	820.4	-454.4
EIGENV_INEL	2	-2.5	463.0	429.4	-33.6
EIGENV_KHI	1	-2.1	600.3	583.0	-17.3
EIGENV_INEL	4	-0.9	342.4	336.3	-6.1
EIGENV_INEL	5	-0.5	218.7	215.7	-3.0
EIGENV_KHI	2	-0.4	247.4	246.2	-1.2
EIGENV_CAPT	1	0.5	884.6	883.9	-0.7
EIGENV_KHI	3	0.2	115.4	114.9	-0.5

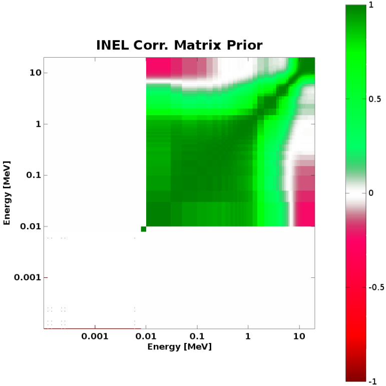
Adjustment of the inelastic XS

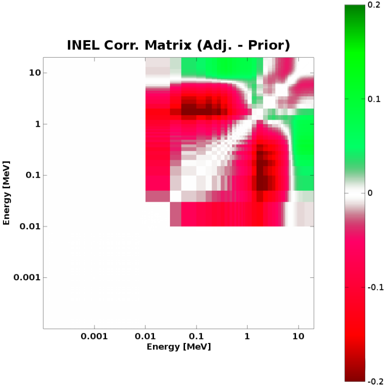


Small reduction of the inelastic uncertainty



Strong negative correlation in the inelastic covariance





Berkeley
UNIVERSITY OF CALIFORNIA

Adjustment via XGPT: first attempt

PROBLEMS/OPEN ISSUES:

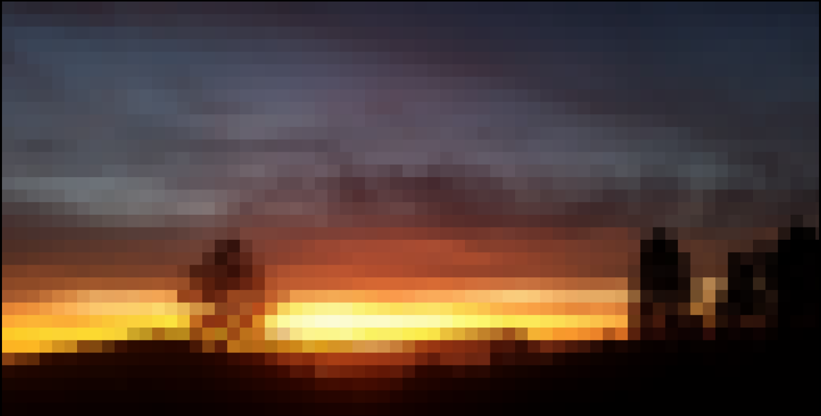
- Continuous-energy cov. matrices are not available
- Unphysical negative eigenvalues
- Few cross-terms in the covariances
- Inelastic: MT4 or MT51-91
- Resonances: MF32 or MF33
- Angular distributions
- TO DO: $S(\alpha, \beta)$ and URR

Acknowledgments

Thanks to:
A. Bidaud (LPSC Grenoble)
D. Rochman (PSI)
A. Sartori (SISSA Trieste)

Jaakko & Serpent developers team

THANK YOU FOR THE ATTENTION



QUESTIONS? SUGGESTIONS? IDEAS?

The bay area from the Berkeley hills (multi-group version).

Off-line steps

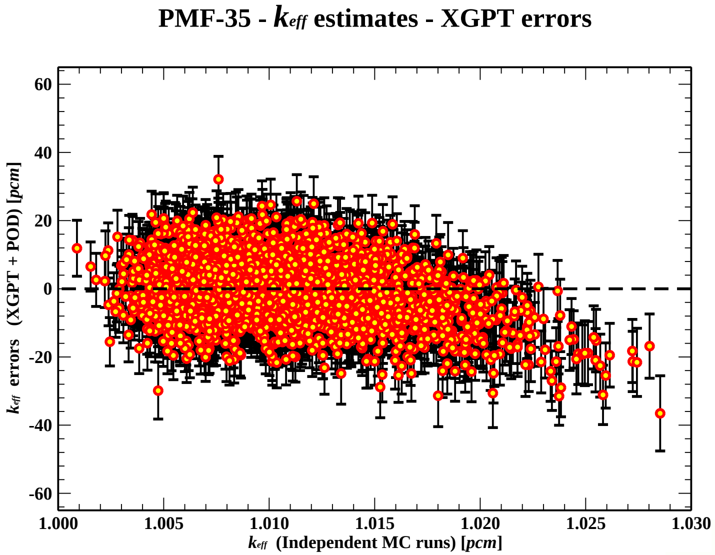
- Generate the optimal bases via POD (from random XS):
  - Load N random ACE files for the selected isotope
  - Score the rel. diff. of the XS on the unionized e-grid
  - Build the (weighted) correlation matrix $K \in \mathbb{R}^{N \times N}$
  - Solve `[S, V] = EIG(K)` for the first n eigenvalues
  - Reconstruct the bases and store them in a cache-friendly way
- Generate the optimal bases via SVD (from cov. matrices):

  Should you already have the relative covariance matrices, the bases can be obtained directly via SVD:
  Solve `[U, S, V] = SVD(COV)` for the first n eigenvalues
- The off-line steps need to be done just once

CPU-time & memory

	TMC	GPT + COV	XGPT + POD
CPU-time	$\propto N$	small $\propto$ # of coll.	small $\propto n$
Memory	–	$\propto$ # of coll. $\times\ \lambda \times$ pop	$\propto n \times \lambda \times$ pop

XGPT+POD: uncertainty propagation in **PMF-35** – k_{eff}



XGPT+POD: uncertainty propagation in PMF-35