Technical workshop : Dynamic nuclear fuel cycle

Reactor description in CLASS

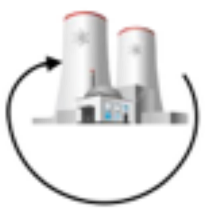
The CLASS Team
Baptiste LENIAU*

Institut d'Astrophysique de Paris
6-8 July, 2016

Subatech
cnrs
ECOLE DES MINES DE NANTES
UNIVERSITÉ DE NANTES

Summary

- The CLASS package : a brief overview



- Problematic of reactor description

$\langle \sigma \rangle$

- Mean cross section « prediction »

$\overrightarrow{N_0}$

- Fresh fuel construction

CLASS : general informations

First code line wrote at Subatech in **2011**. Soon after the LNC-**IRSN** joins the collaboration.

CLASS is a C^{++} library
~15000 code lines

Involved laboratories
Subatech, Nantes
IPNO, Orsay
LPSC,Grenoble
LNC, IRSN, Fontenay aux Roses

Publications
Articles : 3 (in 2015)
Conferences : 7
Past and present PHD : 3

3 teaching workshops

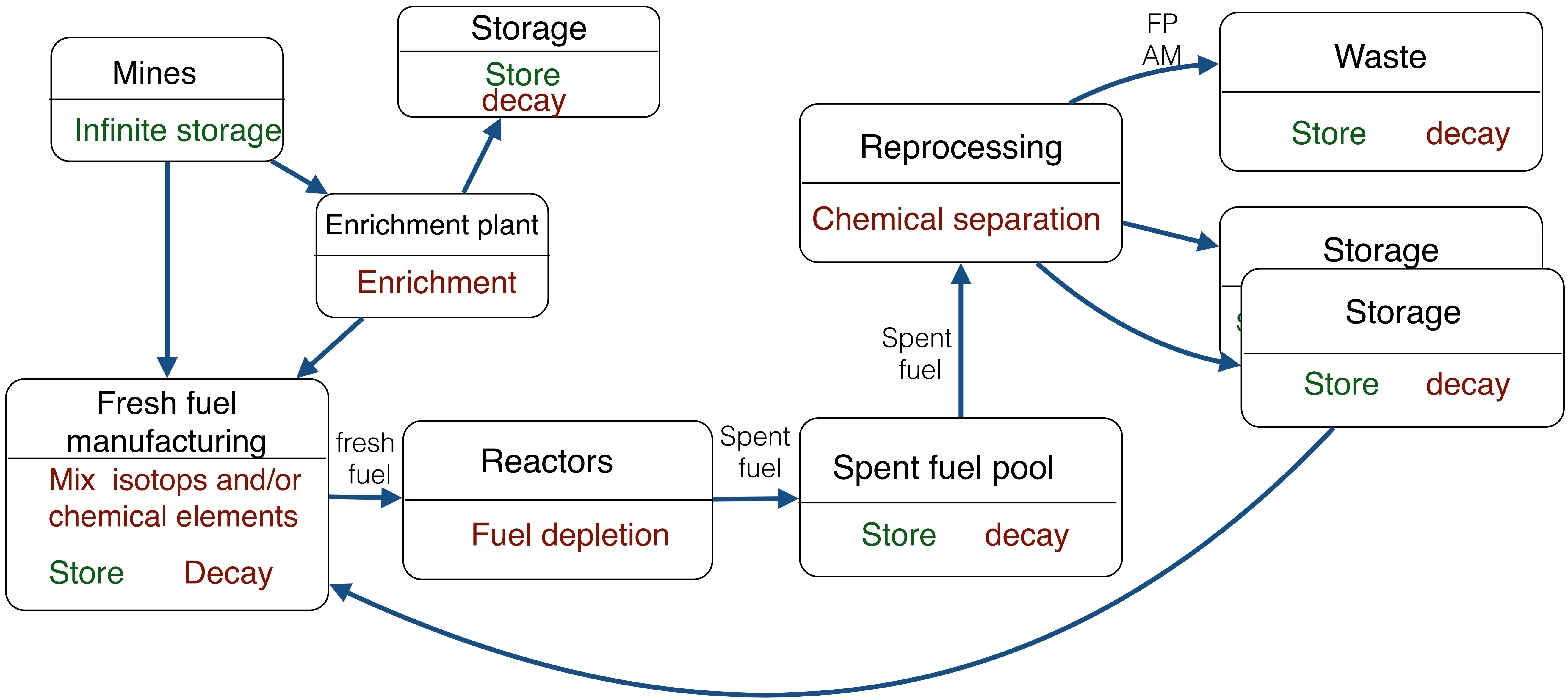
Main developers
B.Mouginot (post-doc) 2011->2015
F. Courtin (PHD) 2014->2017
N.Thiollière 2011 -> -
B.Leniau (post-doc) 2013->2016

GIT/SVN repository, forum, documentation, manual :
https://gitlab.in2p3.fr/sens/CLASS
https://forge.in2p3.fr/projects/classforge

CeCILL License (in progress)
Open source code and models.
Each publication using CLASS should explicitly cite CLASS and models references

CLASS : a nuclear fuel cycle code

As any fuel cycle code, CLASS aims to describe « units » which **store** and/or **transform** and **exchange** matter. The goal being to calculate isotops inventories in units and material flux between units.



exemple of possible connexions

CLASS input : a C^{++} interface

Exemple of a CLASS input :

Storage :

```
/*=== Stock===*/
//Storage for UOX
Storage *StockUOX = new Storage(gCLASS->GetLog());  // Definition of the stock
StockUOX->SetName("StockUOX");                       // Its name
gCLASS->Add(StockUOX);                               //Adding the stock to the Scenario
```

Spent fuel pool :

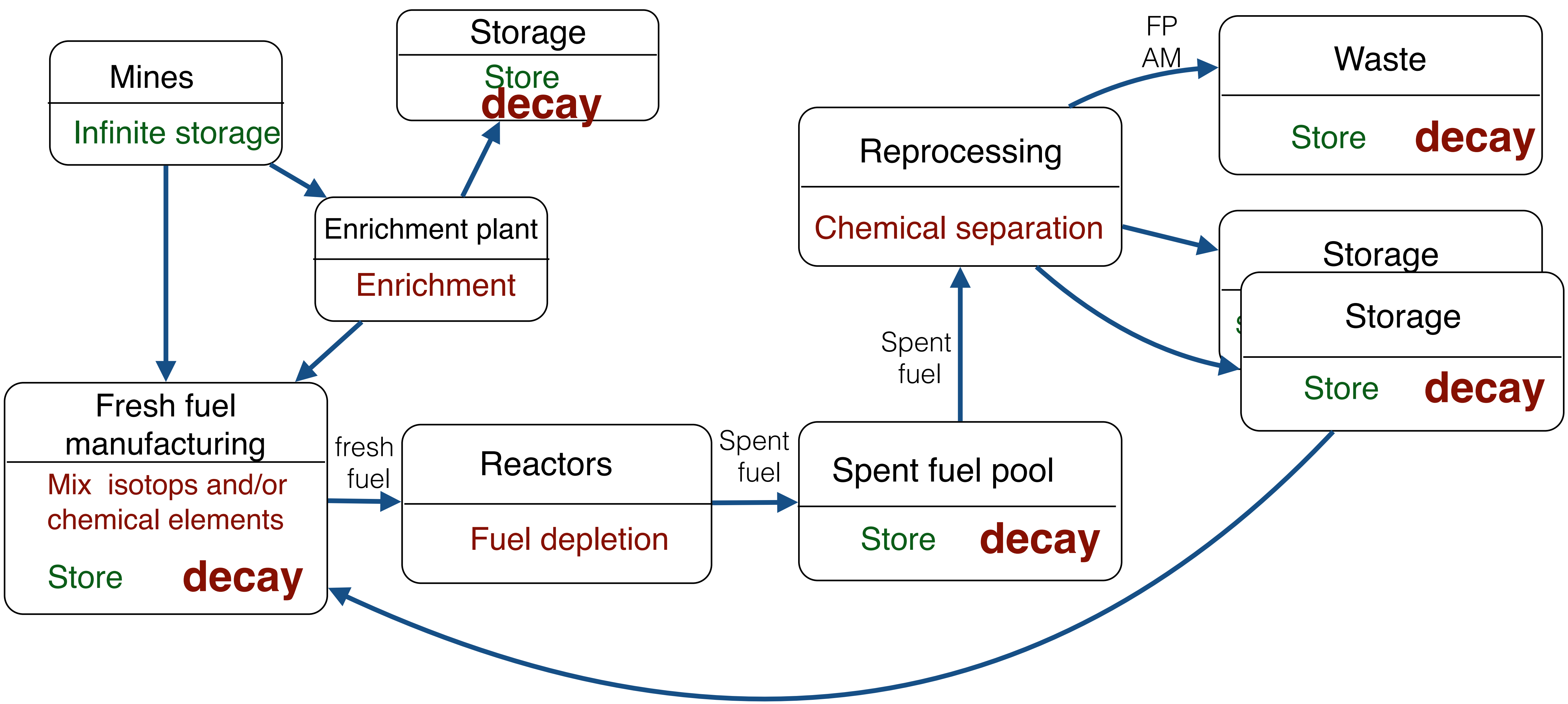
```
//Pool for UOX
Pool *Cooling_UOX = new Pool(gCLASS->GetLog(),StockUOX, Temps_Cooling_UOX);
Cooling_UOX->SetName("Pool_UOX");
gCLASS->Add(Cooling_UOX);
```

Reactor :

```
Reactor* R_MOX = new Reactor(gCLASS->GetLog(),// Log
                             PHYMOD,             // The models used to build the fuel
                             FP_MOX,             // The FabricationPlant
                             Cooling_MOX,          // Connected Backend
                             Temps_Debut_MOX,      // Starting time
                             LifeTime,             // time of reactor l
                             Puissance_MOX,          // Power
                             MasseHM_MOX,        // HM mass
                             BurnUp_MOX,           // BurnUp
                             FacteurDeCharge);           // Load Factor

R_MOX->SetName("The_MOX");// name of the reactor (as it will l
gCLASS->AddReactor(R_MOX);//Add this reactor to the scenario
```

Decay handling (out core)



Decay handling (out core)

Sum of pre-calculated ascii tables

Nucleus inventory
Daughter1 inventory
Daughter2 inventory
• • •

Time

Z	A	I \ time	0	...	1,00E+08	2,00E+08	3,00E+08
94	241	0	1	...	0.858532	0.737077	0.632804
95	241	0	0	...	0.141097	0.261519	0.364194
93	237	0	0	...	0.00037098	0.00140392	0.00300153
91	233	0	0	...	1.20044e-11	4.69712e-11	1.01549e-10
92	233	0	0	...	1.17172e-10	9.40835e-10	3.10253e-09
90	229	0	0	...	3.96366e-16	6.49622e-15	3.251e-14
88	225	0	0	...	2.03528e-21	3.45768e-20	1.75189e-19
89	225	0	0	...	1.30994e-21	2.27899e-20	1.16443e-19
2	4	0	0	...	0.000371008	0.00140394	0.00300155
82	209	0	0	...	1.78109e-23	3.08871e-22	1.57801e-21
83	209	0	0	...	1.42923e-16	5.20101e-17	3.13304e-16
92	237	0	0	...	2.70291e-08	2.32053e-08	1.99225e-08

portion of a pre-calculated decay table of ONE NUCLEUS (^{241}Pu)

Example :

Considering a sample made of : N^1 nuclei of ^{241}Pu and N^2 nuclei of ^{238}Pu

What is the composition of this sample 3 month after ?

portion of ^{241}Pu table

$\overrightarrow{Daughter}_1$

Z	A	I \ time	0	...	1,00E+08	2,00E+08	3,00E+08
94	241	0	1	...	0.858532	0.737077	0.632804
95	241	0	0	...	0.141097	0.261519	0.364194
93	237	0	0	...	0.00037098	0.00140392	0.00300153
91	233	0	0	...	1.20044e-11	4.69712e-11	1.01549e-10
92	233	0	0	...	1.17172e-10	9.40835e-10	3.10253e-09
90	229	0	0	...	3.96366e-16	6.49622e-15	3.251e-14
88	225	0	0		2.03528e-21	3.45768e-20	1.75189e-19

$N^1 \cdot \overrightarrow{Daughter}_1$

\+

Sample with 3m of decay

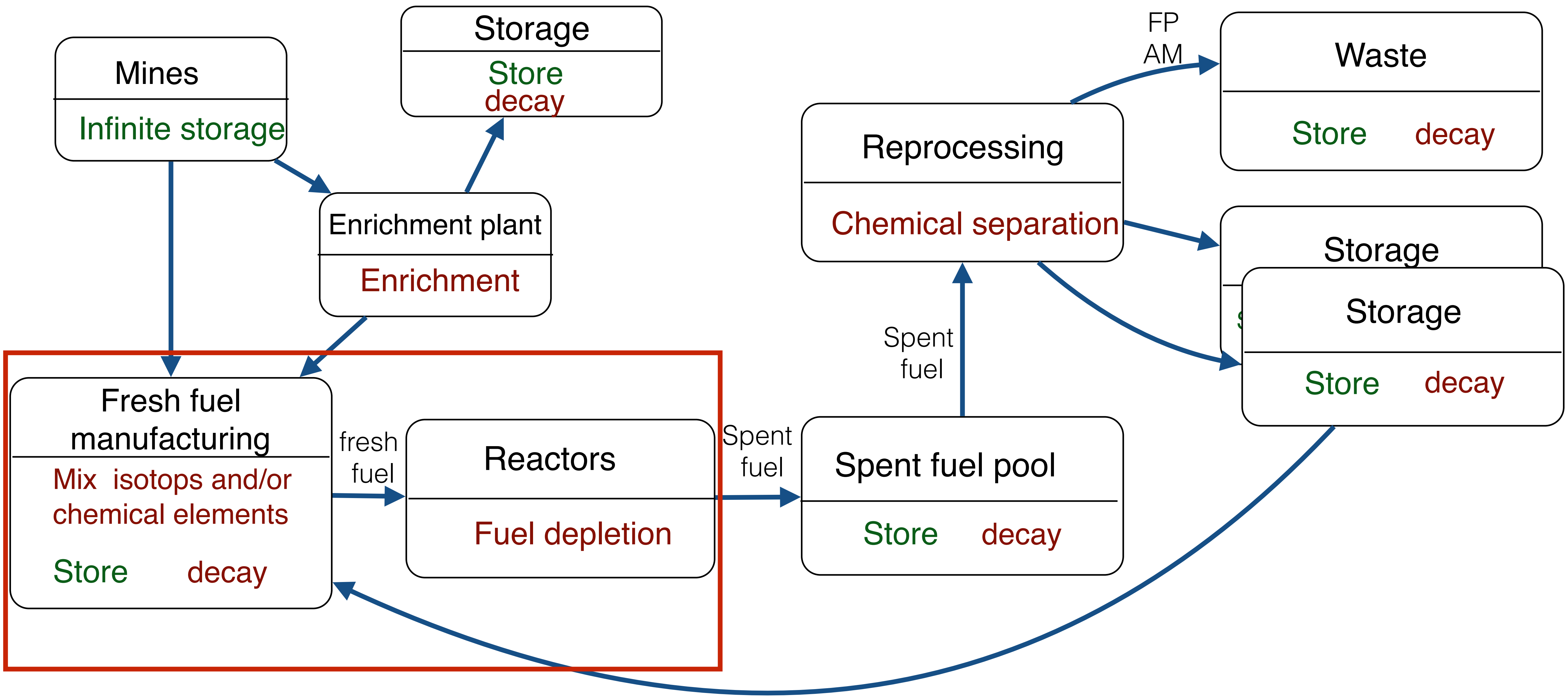
portion of ^{238}Pu table

$\overrightarrow{Daughter}_2$

Z	A	I / time	0	...	1,00E+08	2,00E+08	3,00E+08
94	238	0	1	...	0.982621	0.980163	0.977712
92	234	0	0	...	0.0173787	0.0198366	0.0222883
90	230	0	0	...	5.45339e-08	[illegible].11686e-08	8.99978e-08
88	226	0	0	...	3.71127e-13	5.53621e-13	7.8774e-13
86	222	0	0	...	2.3802e-18	3.55962e-18	5.07497e-18
2	4	0	0	...	0.0173788	0.0198367	0.0222885
82	210	0	0		8.56866e-17	1.46297e-16	2.34353e-16

$N^2 \cdot \overrightarrow{Daughter}_2$

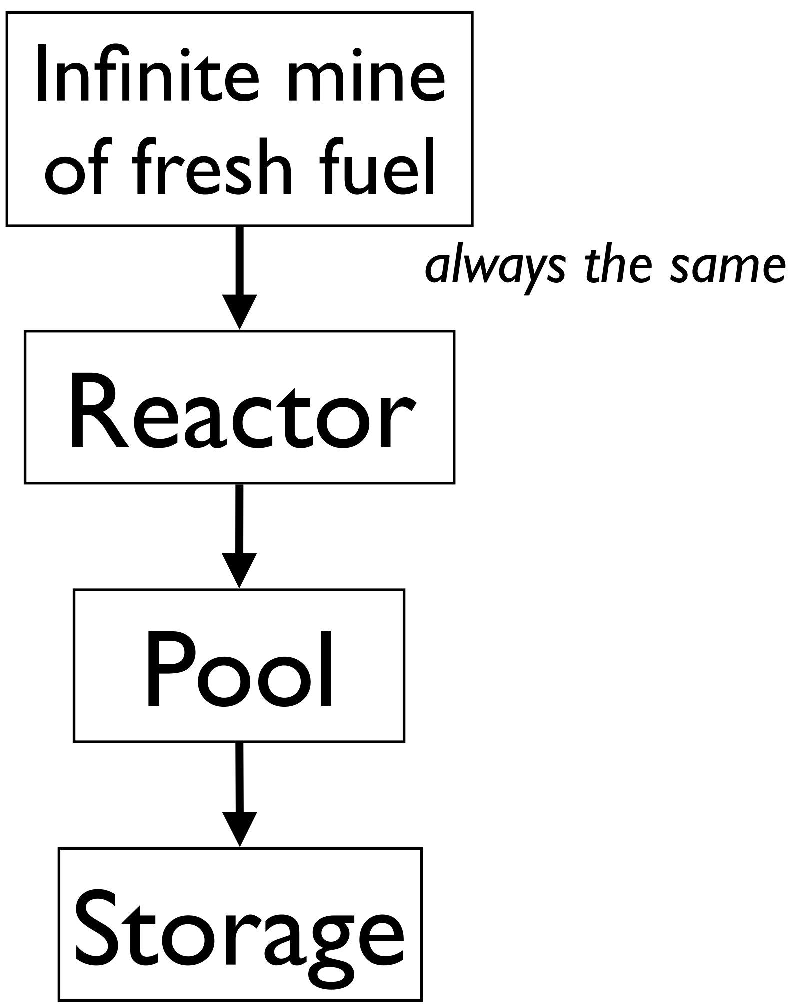
Reactor handling



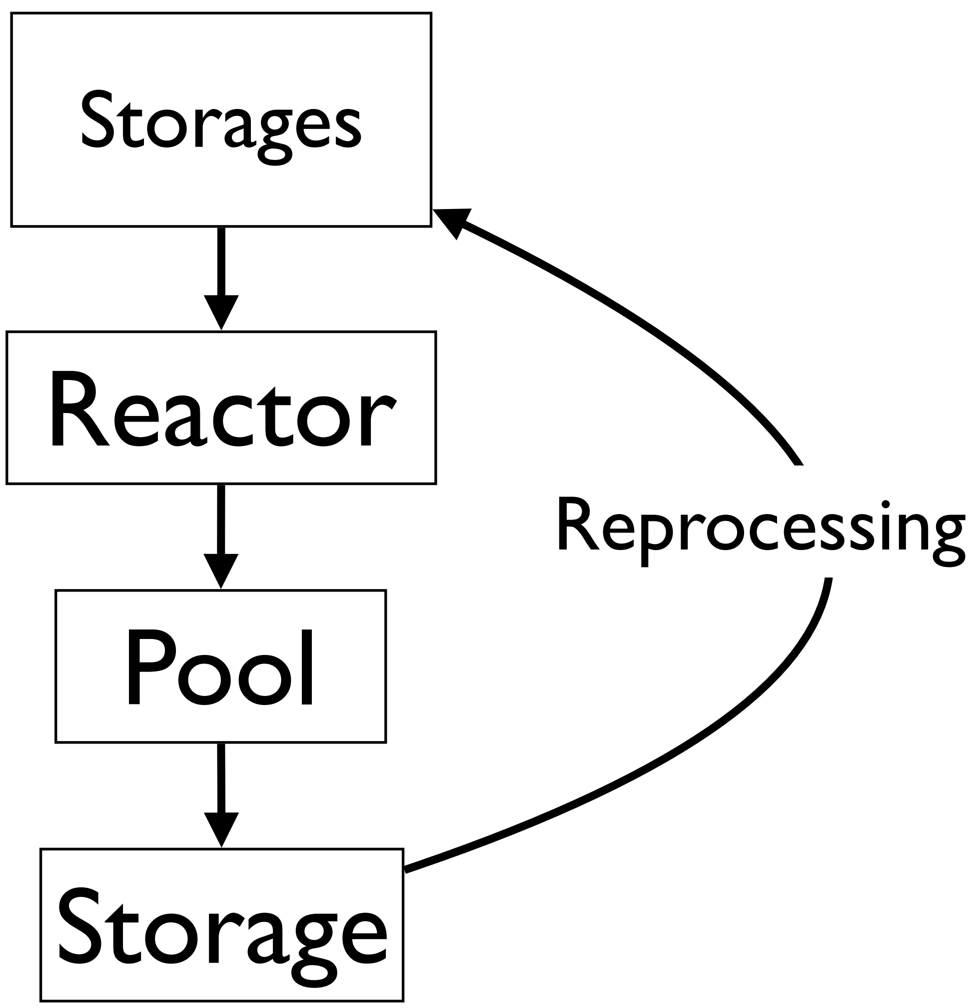
Reactor handling : two cases

Fixed fresh fuel

(recipe based)



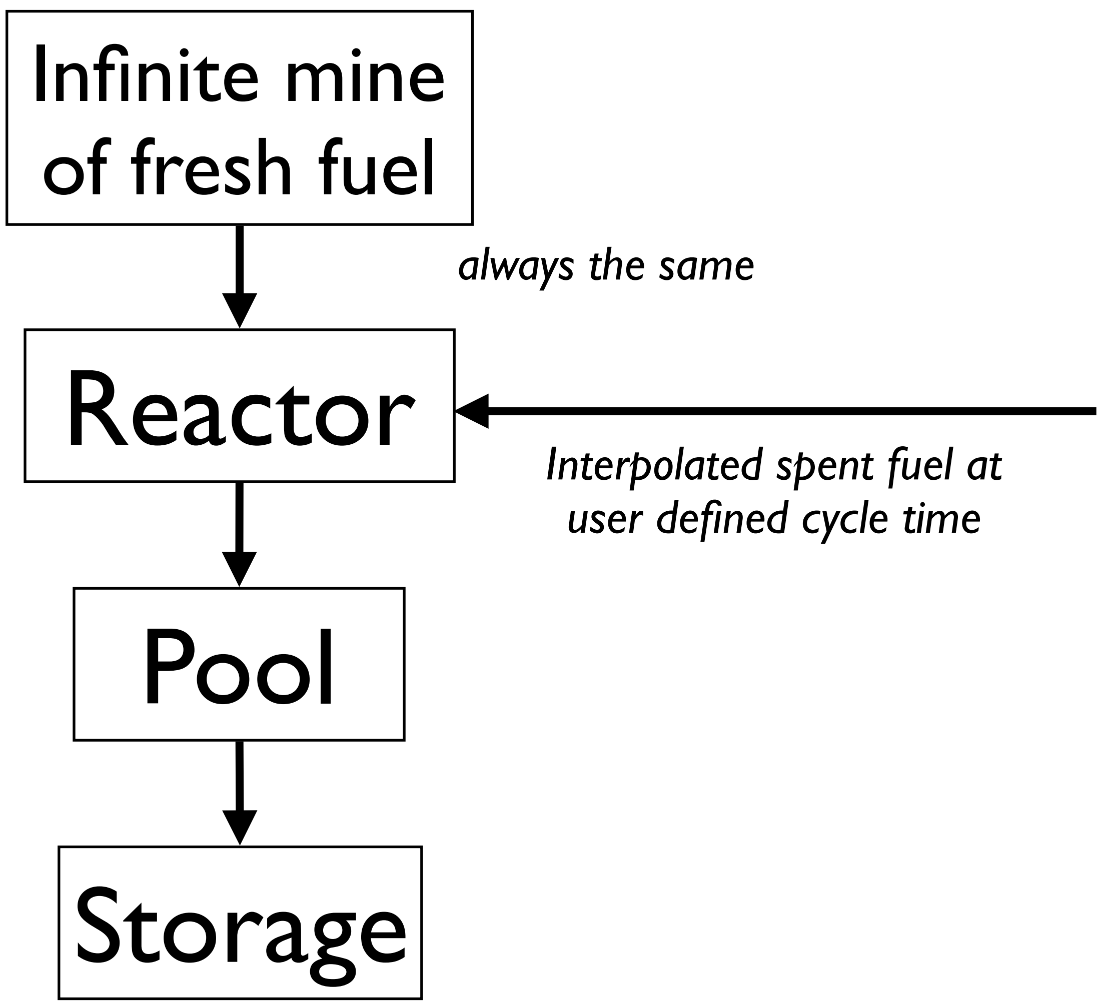
Dynamic fresh fuel



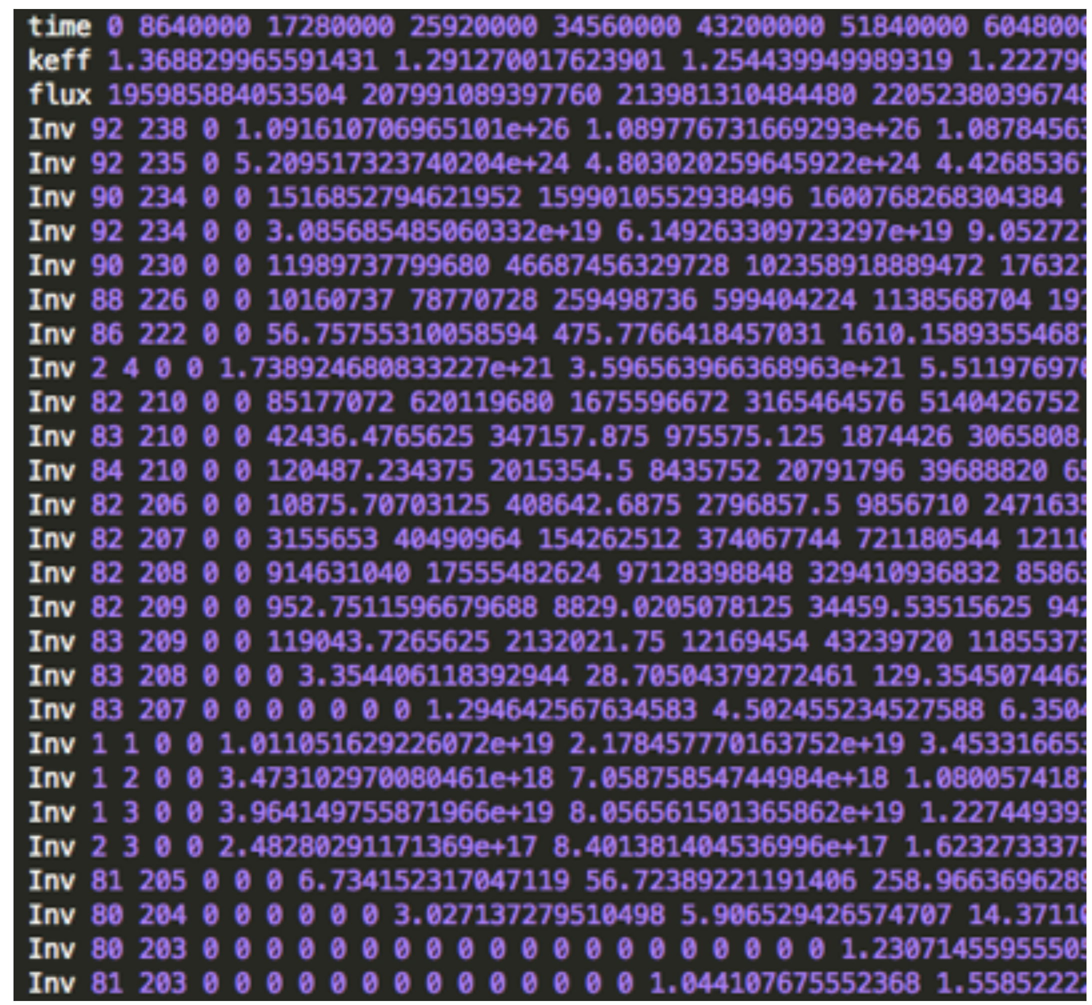
Reactor handling

Fixed fresh fuel

(recipe based)



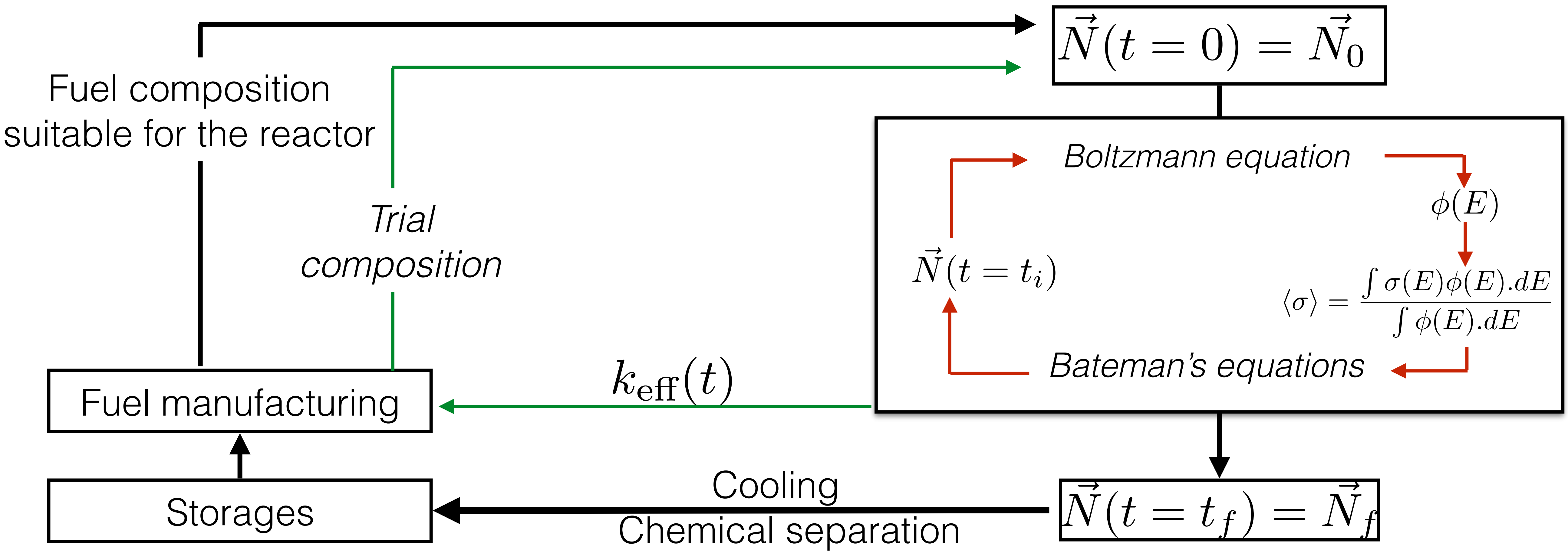
EvolutionData : pre-calculated fuel depletion



```
time 0 8640000 17280000 25920000 34560000 43200000 51840000 604800
keff 1.368829965591431 1.291270017623901 1.254439949989319 1.22279
flux 195985884053504 207991089397760 213981310484480 220523803967
Inv 92 238 0 1.091610706965101e+26 1.089776731669293e+26 1.0878456
Inv 92 235 0 5.209517323740204e+24 4.803020259645922e+24 4.4268536
Inv 90 234 0 0 1516852794621952 1599010552938496 1600768268304384
Inv 92 234 0 0 3.085685485060332e+19 6.149263309723297e+19 9.05272
Inv 90 230 0 0 11989737799680 46687456329728 102358918889472 17632
Inv 88 226 0 0 10160737 78770728 259498736 599404224 1138568704 19
Inv 86 222 0 0 56.75755310058594 475.7766418457031 1610.158935546
Inv 2 4 0 0 1.738924680833227e+21 3.596563966368963e+21 5.51197697
Inv 82 210 0 0 85177072 620119680 1675596672 3165464576 5140426752
Inv 83 210 0 0 42436.4765625 347157.875 975575.125 1874426 3065808
Inv 84 210 0 0 120487.234375 2015354.5 8435752 20791796 39688820 6
Inv 82 206 0 0 10875.70703125 408642.6875 2796857.5 9856710 247163
Inv 82 207 0 0 3155653 40490964 154262512 374067744 721180544 1211
Inv 82 208 0 0 914631040 17555482624 97128398848 329410936832 8586
Inv 82 209 0 0 952.7511596679688 8829.0205078125 34459.53515625 94
Inv 83 209 0 0 119043.7265625 2132021.75 12169454 43239720 1185537
Inv 83 208 0 0 0 3.354406118392944 28.70504379272461 129.354507446
Inv 83 207 0 0 0 0 0 0 0 1.294642567634583 4.502455234527588 6.350
Inv 1 1 0 0 1.011051629226072e+19 2.178457770163752e+19 3.45331665
Inv 1 2 0 0 3.473102970080461e+18 7.058758547449840e+18 1.08005741
Inv 1 3 0 0 3.964149755871966e+19 8.056561501365862e+19 1.22744939
Inv 2 3 0 0 2.482802911713690e+17 8.401381404536996e+17 1.62327333
Inv 81 205 0 0 0 6.734152317047119 56.72389221191406 258.966369628
Inv 80 204 0 0 0 0 0 0 3.027137279510498 5.906529426574707 14.3711
Inv 80 203 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1.230714559555
Inv 81 203 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1.044107675552368 1.5585222
```

Reactor problematic : fuel from reprocessed materials

Determine the fresh fuel composition and predict its irradiation in reactor

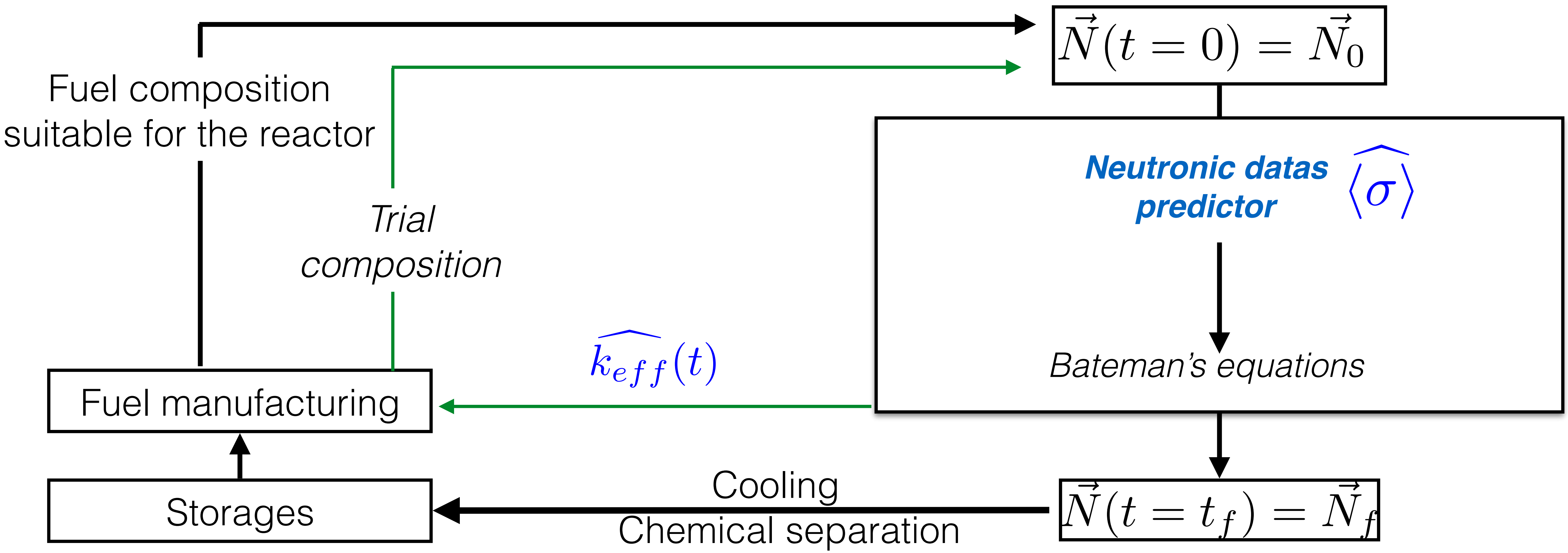


$\langle\sigma\rangle$ and $k_{\mathrm{eff}}(t)$ strongly depends on $\overrightarrow{N_0}$, which is unknown a-priori

How to avoid calls to time consuming neutron transport code ?

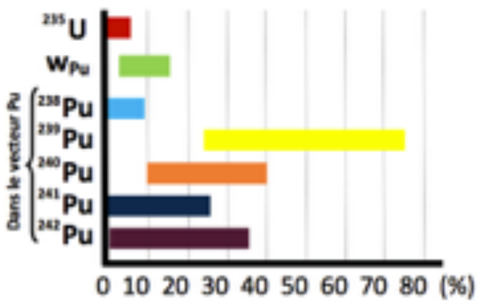
Reactor problematic : fuel from reprocessed materials

Determine the fresh fuel composition and predict its irradiation in reactor



<u>Approach used</u> : Replace the neutron transport code by statistical predictors

Methodology : Building a reactor model for CLASS

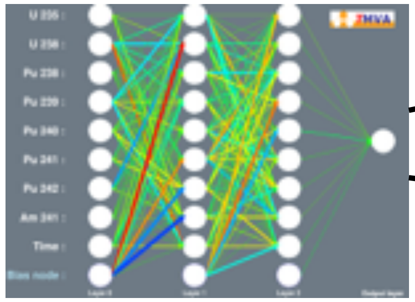


Define fresh fuel isotopic variation domain

Fuel depletion calculations (Boltzmann/ Bateman)

$\{\langle\sigma\rangle\,(t)\}$

Generation of a training data base



Multivariate regressions (Artificial neural networks)

$\widehat{k_{eff}}(\overrightarrow{N_0}, t)$ → $\overrightarrow{N_0}$

$\left\{\widehat{\langle\sigma\rangle}(\overrightarrow{N_0}, t)\right\}$ → $\frac{d\vec{N}}{dt} = A\vec{N}$

Statistical estimators

Reactor model for CLASS

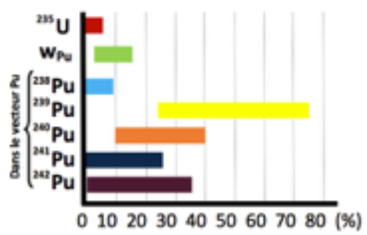
Existing models :

- PWR-UOx
- **PWR-MOX (this talk)**
- PWR-MOX on enriched uranium support. *(Fanny COURTIN's talk, this afternoon)*
- SFR MOX

In development :

- PWR-MOX with Americium *(A. Zakari's talk Friday morning)*
- ADS
- CANDU U from reprocessing

Example : PWR MOX Cross section predictors

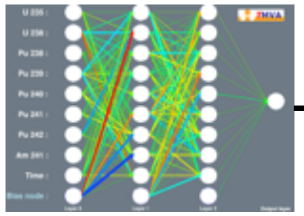


Hyperspace of isotopic composition of the fresh fuel

Depletion calculations (coupled Bateman/ Boltzmann solvers)

$\{\langle\sigma\rangle(t)\}$

Data base generation



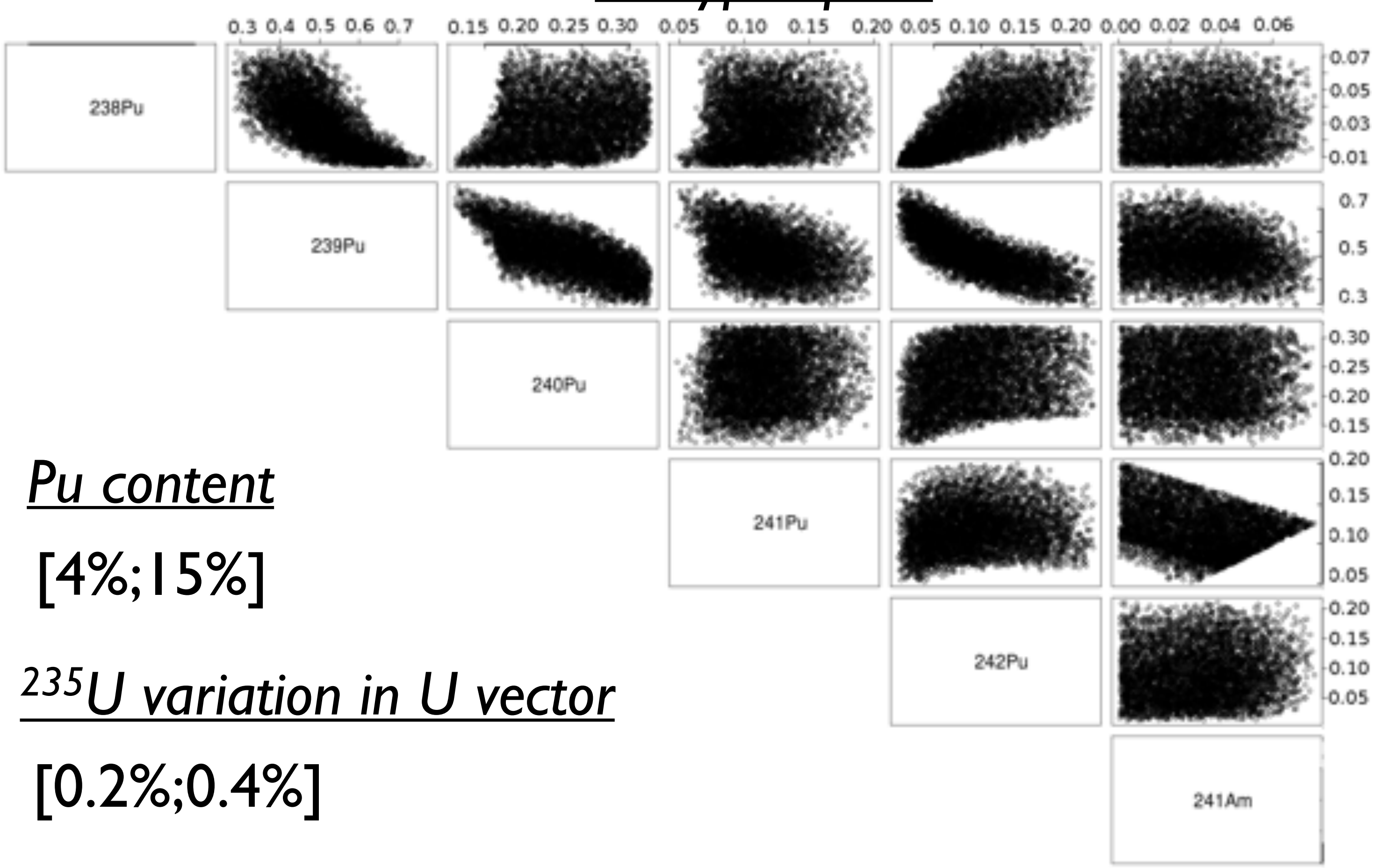
Multivariate interpolation (Artificial neural network)

$\{\widehat{\langle\sigma\rangle}(\vec{N_0},t)\}$

Statistical estimators

$$\frac{d\vec{N}}{dt} = A\vec{N}$$

Pu hyperspace



Pu content

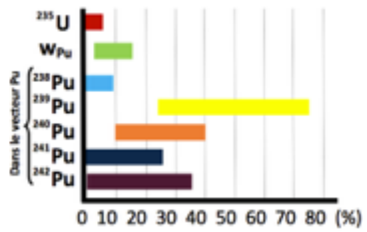
[4%;15%]

^{235}U variation in U vector

[0.2%;0.4%]

The plutonium hyperspace is generated using 6 different PWR-UOx fuel depletion calculation (representative of the french fleet) at difference irradiation time.

Example : PWR MOX Cross section predictors

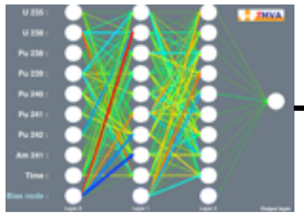


Hyperspace of isotopic composition of the fresh fuel

Depletion calculations (coupled Bateman/ Boltzmann solvers)

$$\{\langle\sigma\rangle\,(t)\}$$

Data base generation



Multivariate interpolation (Artificial neural network)

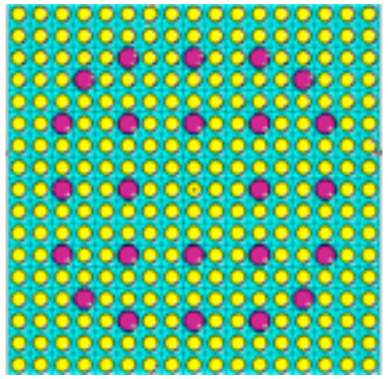
$$\{\widehat{\langle\sigma\rangle}(\vec{N_0},t)\}$$

Statistical estimators

$$\frac{d\vec{N}}{dt} = A\vec{N}$$

Main characteristics :

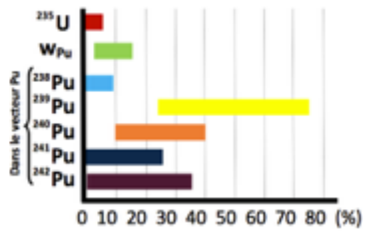
Assembly 17x17 lattice
100% MOX
Fixed power density : 30 W/g
MOX composition : Variable



Main simulation parameters :

Infinite calculation (mirror boundaries)
Impact of UOx neighbor not taken into account
Fixed boron content : 600ppm
MCNP steps : 11
Multi-group : yes (17900 groups)
Data Bases (XS,FPYields,S(α,β)) : JEFF 3.1.1
Half life threshold : 1 hour
Running time (1cpu) : ~2 hours

Example : PWR MOX Cross section predictors

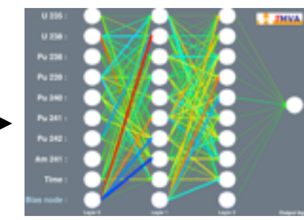


Hyperspace of isotopic composition of the fresh fuel

Depletion calculations (coupled Bateman/ Boltzmann solvers)

$\{\langle\sigma\rangle\,(t)\}$

Data base generation



Multivariate interpolation (Artificial neural network)

$\{\widehat{\langle\sigma\rangle}(\overrightarrow{N_0},t)\}$

Statistical estimators

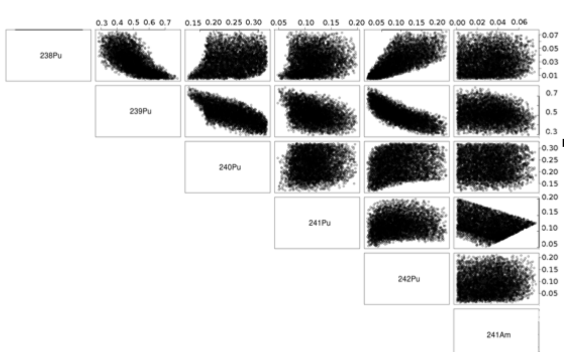
$\frac{d\vec{N}}{dt} = A\vec{N}$

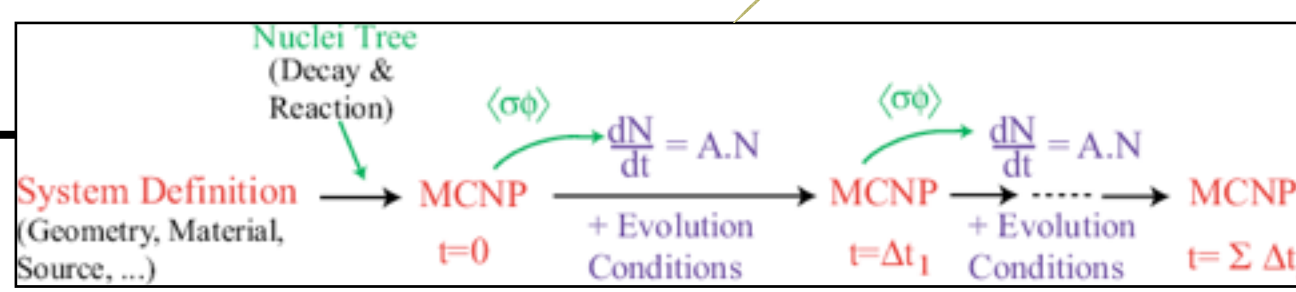
Depletion calculations

N different fresh fuel compositions

N fuel depletion calculations using the code :

N sets of results:

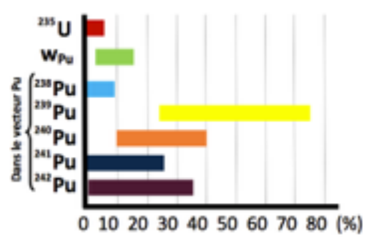




$$\{\langle\sigma\rangle\,(t)\}$$

~2h per depletion calculation on 1 CPU

Example : PWR MOX Cross section predictors

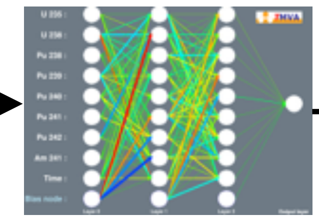


Hyperspace of isotopic composition of the fresh fuel

Depletion calculations (coupled Bateman/ Boltzmann solvers)

$\{\langle\sigma\rangle(t)\}$

Data base generation



Multivariate interpolation (Artificial neural network)

$\{\widehat{\langle\sigma\rangle}(\vec{N_0},t)\}$

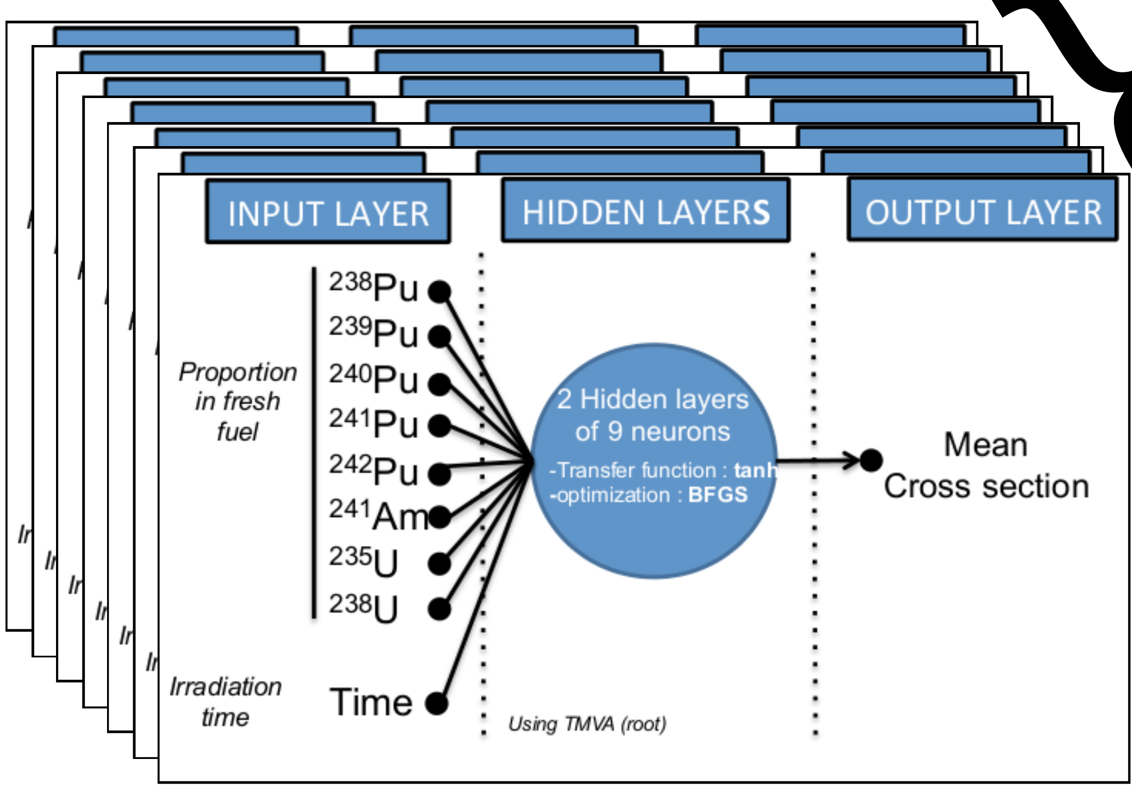
Statistical estimators

$$\frac{d\vec{N}}{dt} = A\vec{N}$$

Using the depletion calculations to train ANNs to predict :

$\widehat{\langle\sigma\rangle} = f(\vec{N_0},t)$ *(Power is constant)*

One neural network per reaction…



Number of reactions :

- Nuclei considered : 365
- Reaction type handle : (n,γ), (n,f) and (n,2n)
- Number of reactions : 700

PWR MOX model testing performances

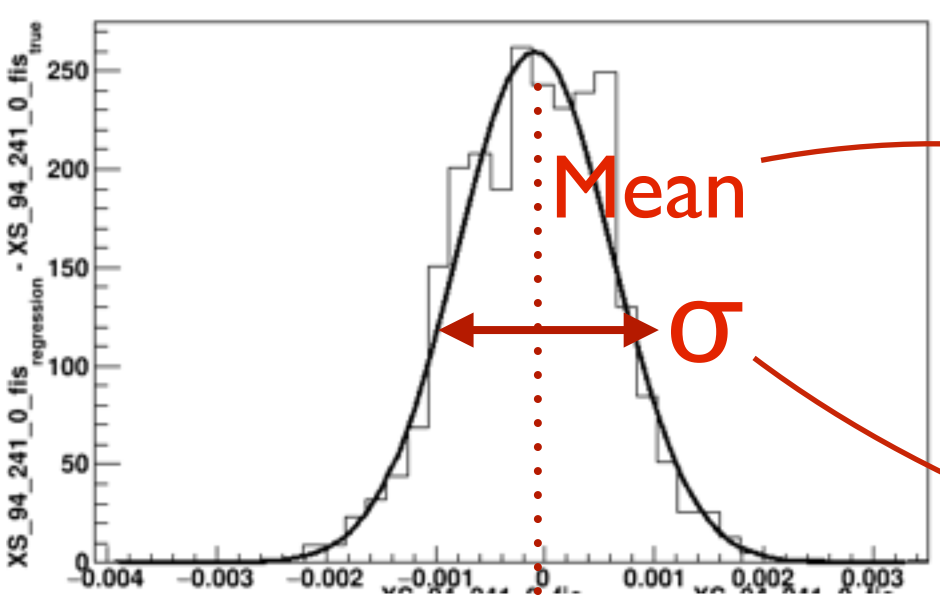
Testing procedure :

$$\text{ANN Error} = \frac{\widehat{\langle\sigma\rangle}^{N} - \langle\sigma\rangle^{\text{test}}}{\langle\sigma\rangle^{\text{test}}}$$

$\widehat{\langle\sigma\rangle}^{N}$ Mean cross section predict by ANN trained with ***N*** MURE calculations results

$\langle\sigma\rangle^{\text{test}}$ Mean cross section calculated with MURE NOT USED FOR TRAINING

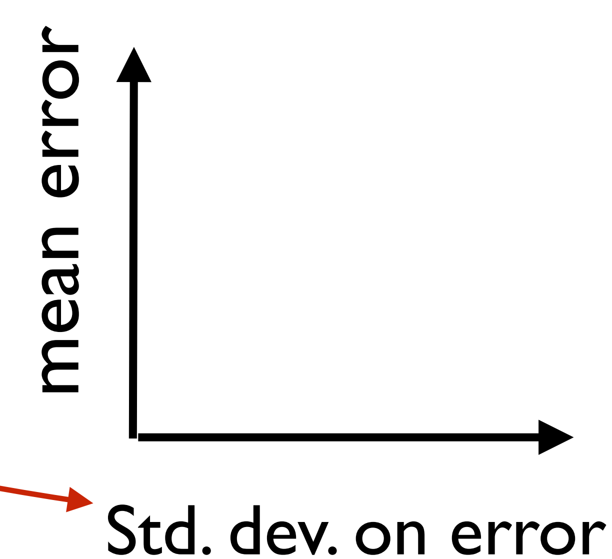
For each reaction and for each training sample size :



Example :**Error** distribution on ^{241}**Pu(n,f)** prediction

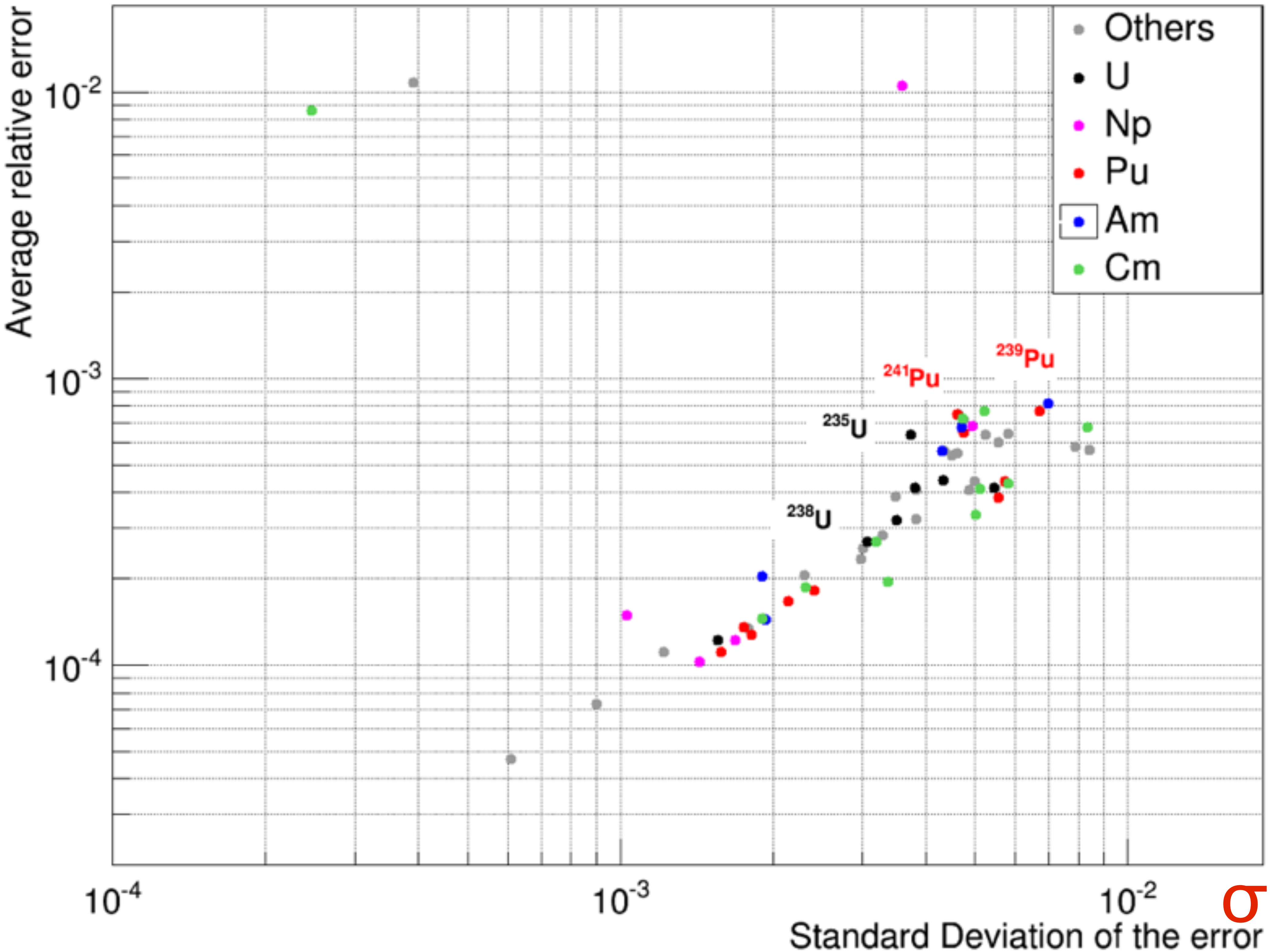
ANN Error

Plot each (mean,σ) on a graph



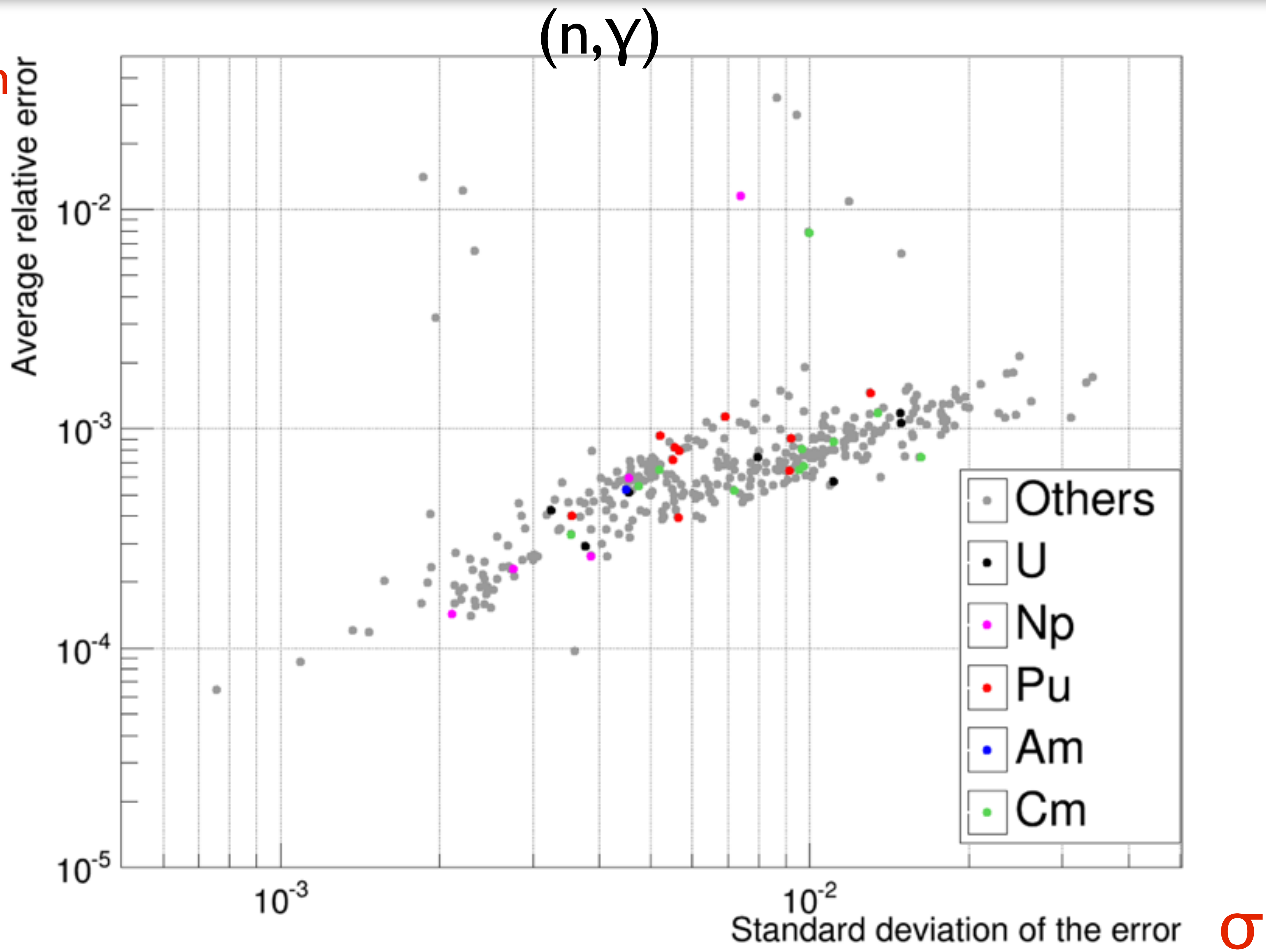
Testing performances : fission

Mean



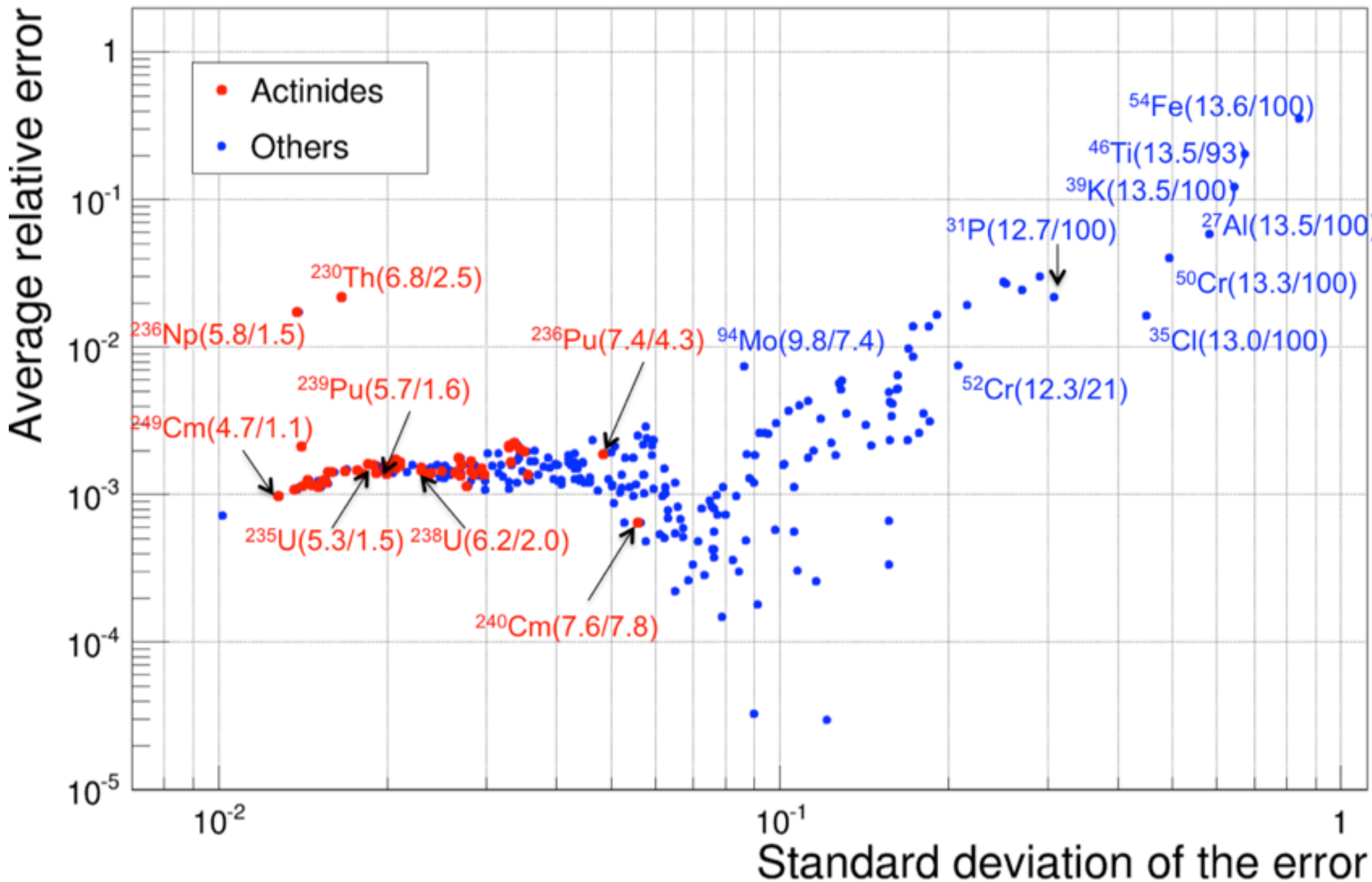
σ

Testing performances : capture



Testing performances : (n,2n)

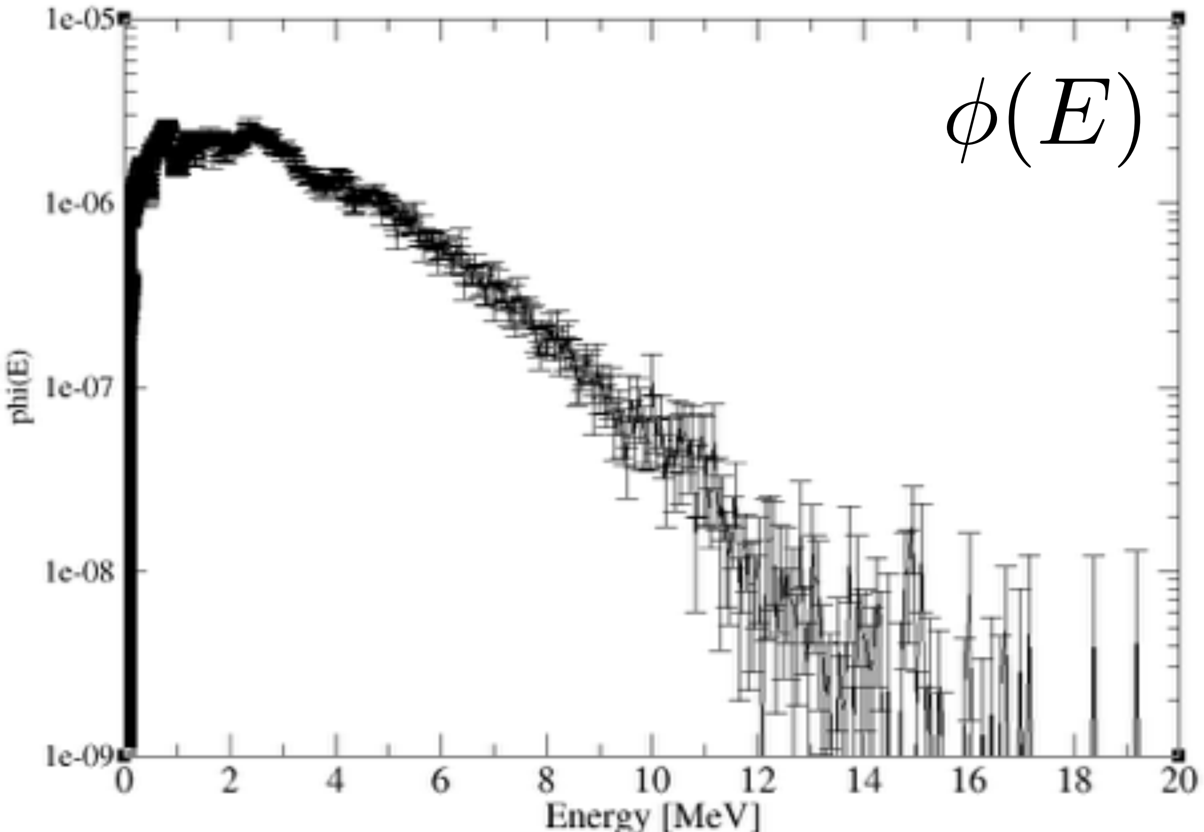
Mean

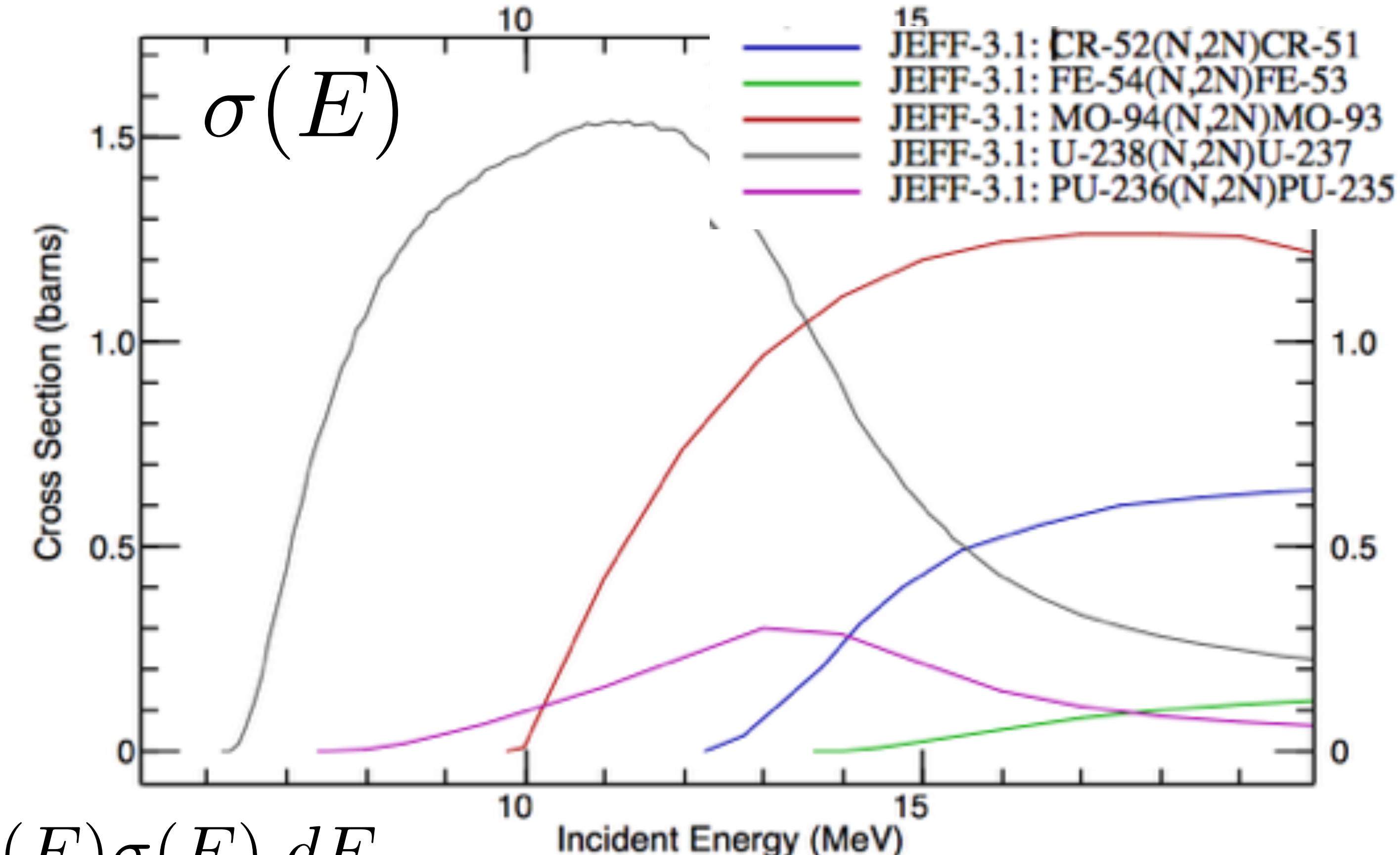


σ

^{A}X(Energy threshold, MCNP statistical error)

(n,2n) statistical error





$$\langle\sigma\rangle = \frac{\int \phi(E)\sigma(E).dE}{\int \phi(E).dE}$$

Error more important for (n,2n) reactions :

(n,2n) : Threshold Reaction

->Low statistic in fast region of neutron spectrum (Monte Carlo !)

->High statistical error on (n,2n)

-> Noisy mean cross section

->ANN can't be better than statistical error on the training sample !

Induce errors in fuel depletion calculation

Methodology :

Reference

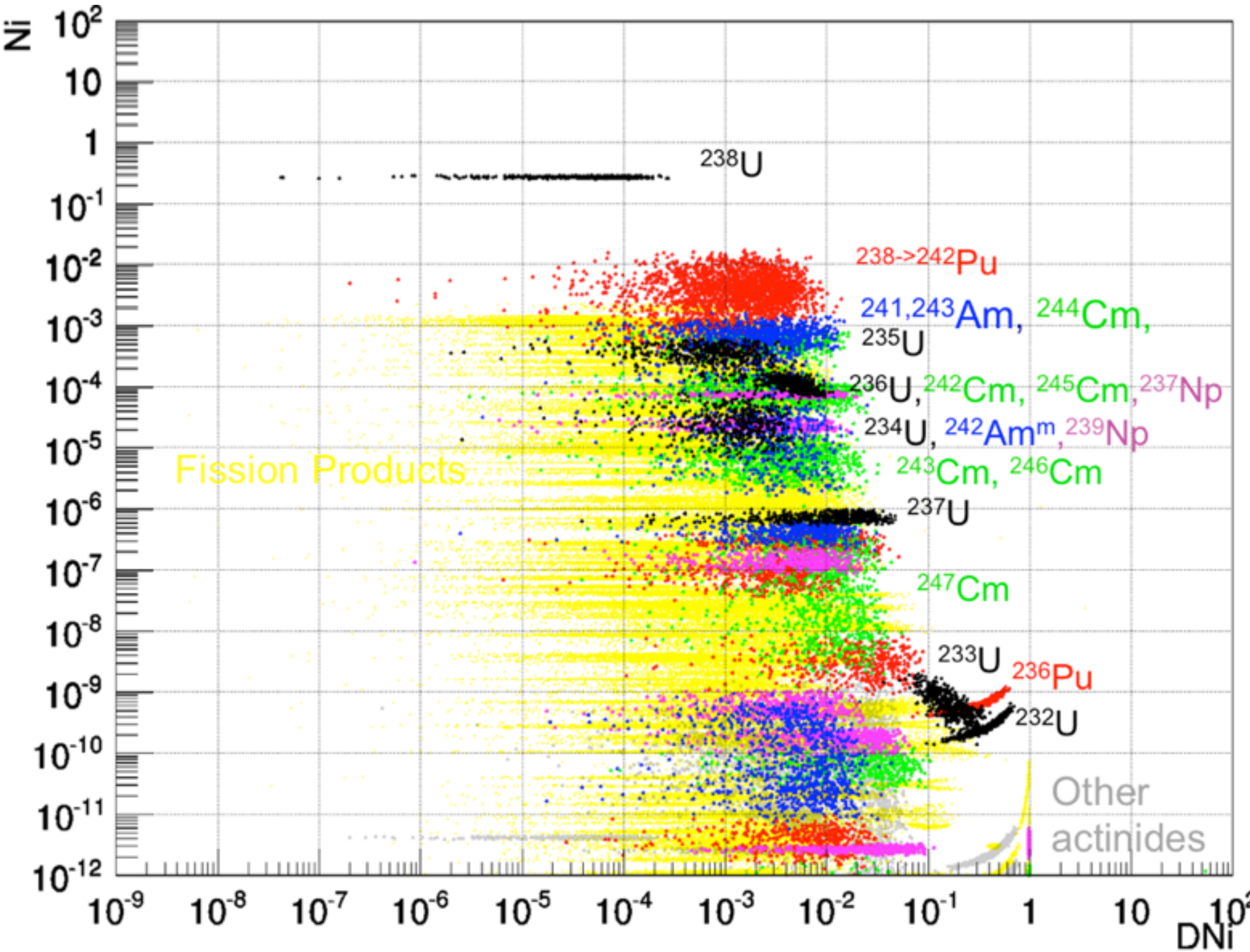
- Generation of 500 random fresh fuel compositions in the hyperspace previously defined.
- Depletion calculations (MURE) with these 500 different fresh fuels

CLASS models output

- The same 500 fresh fuel compositions are used as input for the mean cross sections predictor
- These cross sections are used by the Bateman solver of CLASS

The end of cycle inventories are compared

Induce errors in fuel depletion calculation



How to read ?

Y axis:

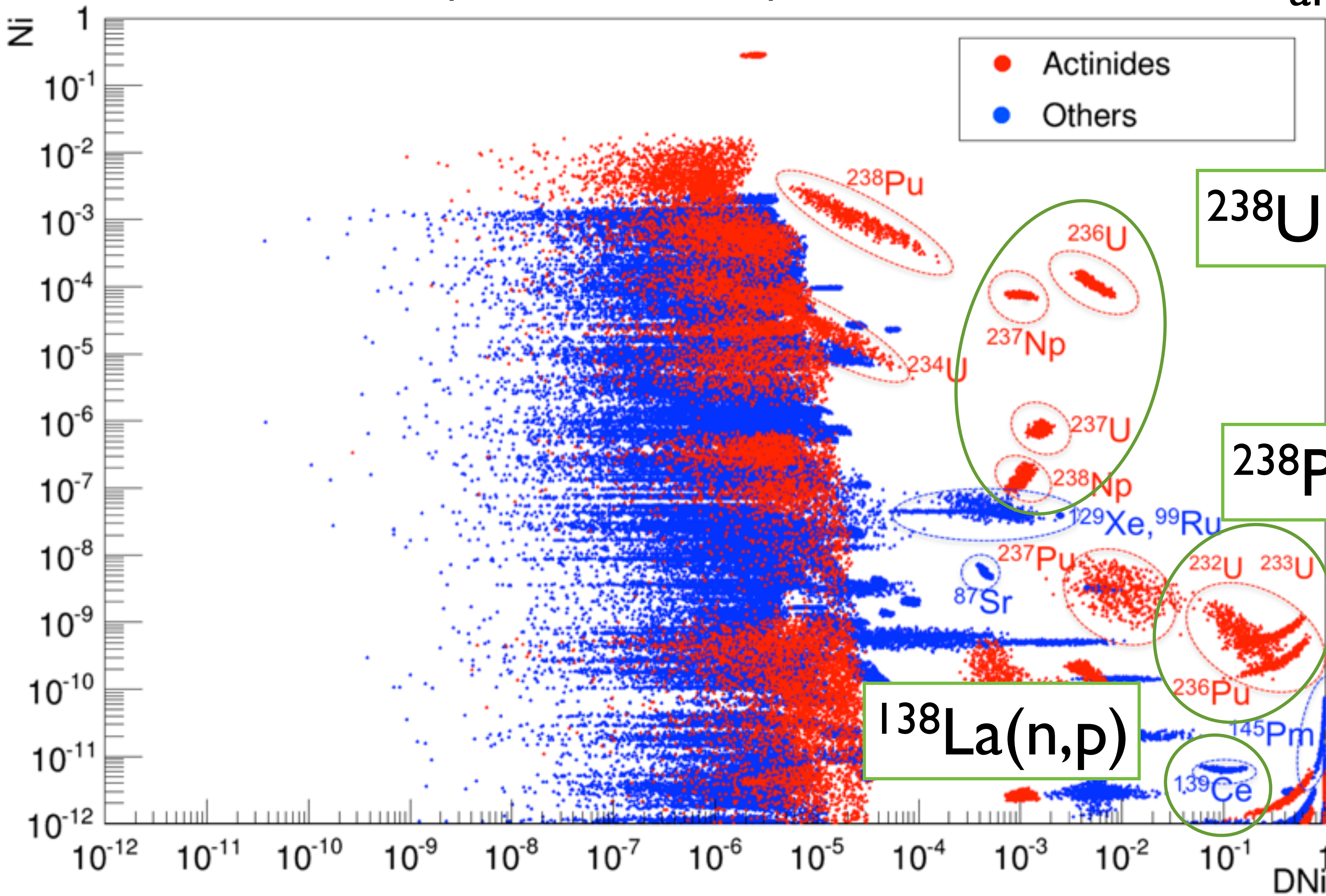
Proportion in spent fuel

X axis:

Relative difference between a MURE calculation and a depletion calculation with predicted <σ> via ANN

Impact of the 3 reactions assumption

CLASS 3 reactions (XS from MURE) VS MURE all reactions



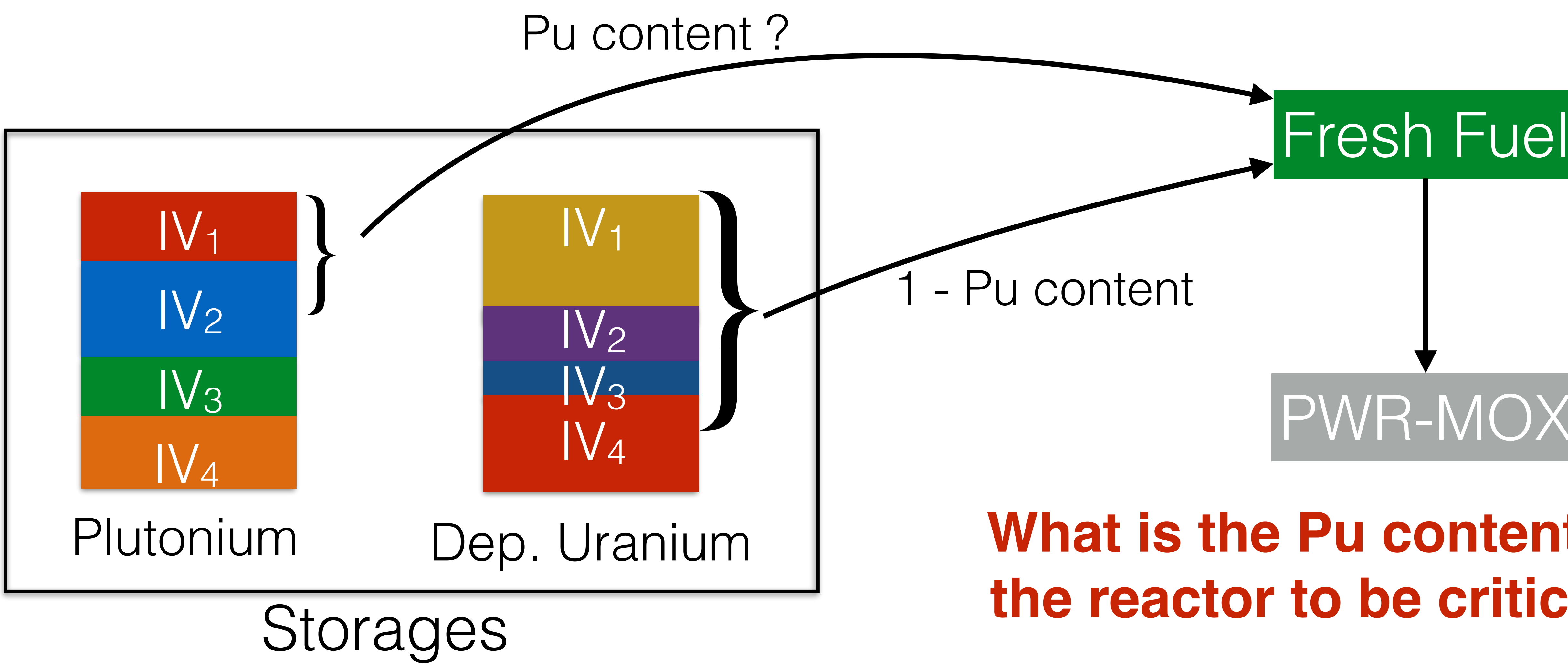
all reaction :
3 reactions
+
(n,p)
...

$^{238}U(n,3n)$

$^{238}Pu(n,3n)$

$^{138}La(n,p)$

Example of building a MOX fuel for a PWR



What is the Pu content for the reactor to be critical ?

Isotopic compositions determined during fuel cycle simulation

The averaged k_∞ model

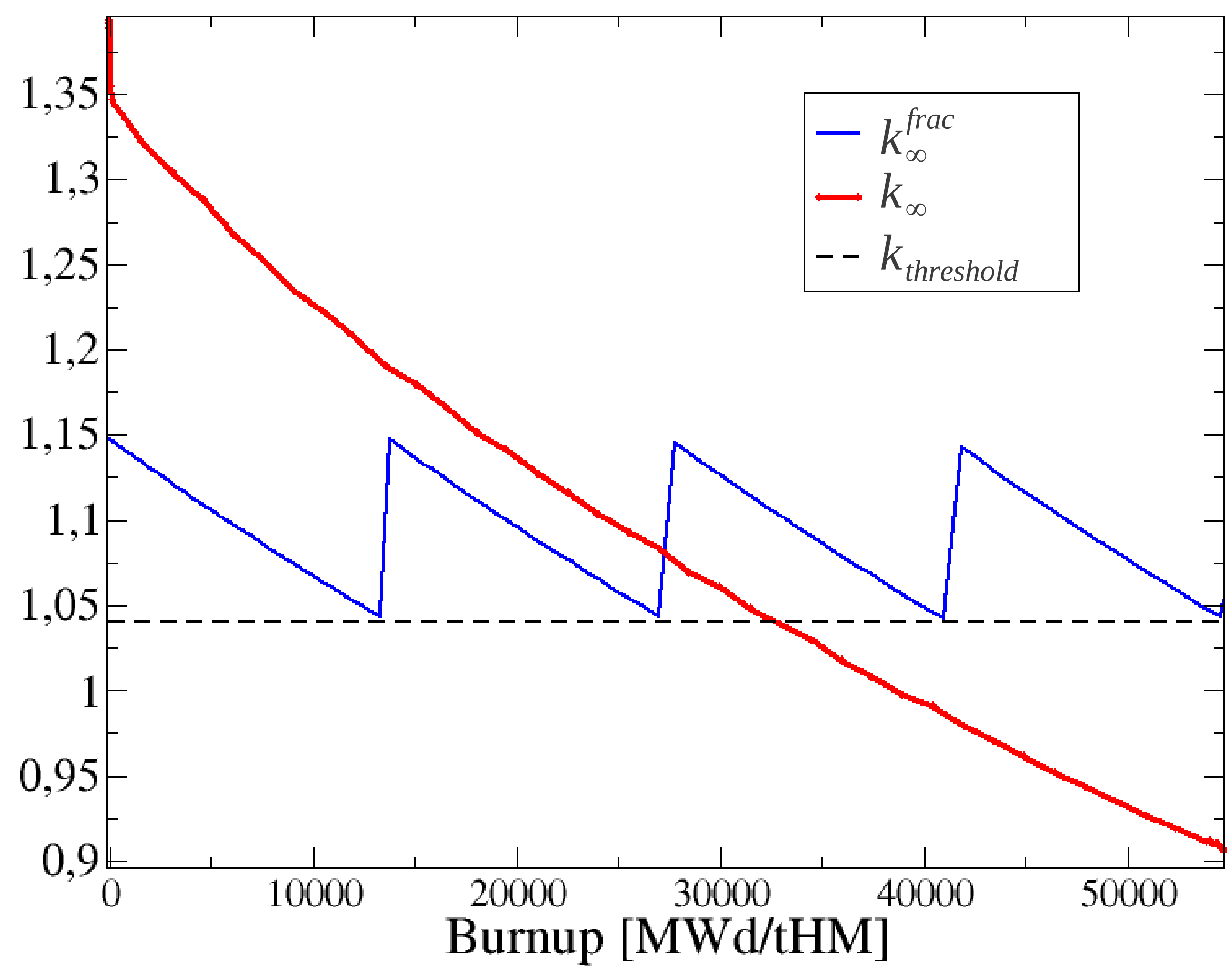
•The **fuel management in CANDU/PWR/BWR/SFR…** uses **batches** to flatter the flux profile and to **flatter the k_{eff} evolution** (fresh fuel assemblies compensate de lake of criticality of the old ones)
@ a fuel loading : 1/N assemblies are loaded, 1/N are moved in the core,...,1/N are unloaded
For PWR : N = 3 or 4.

How to mimic batch management with infinite calculation ?

Considering that the behavior of an assembly isn't impacted by its surrounding media, we defined the average k_∞ over the batches as :

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_{i=0}^{N-1} k_\infty(t + \frac{iT}{N})$$

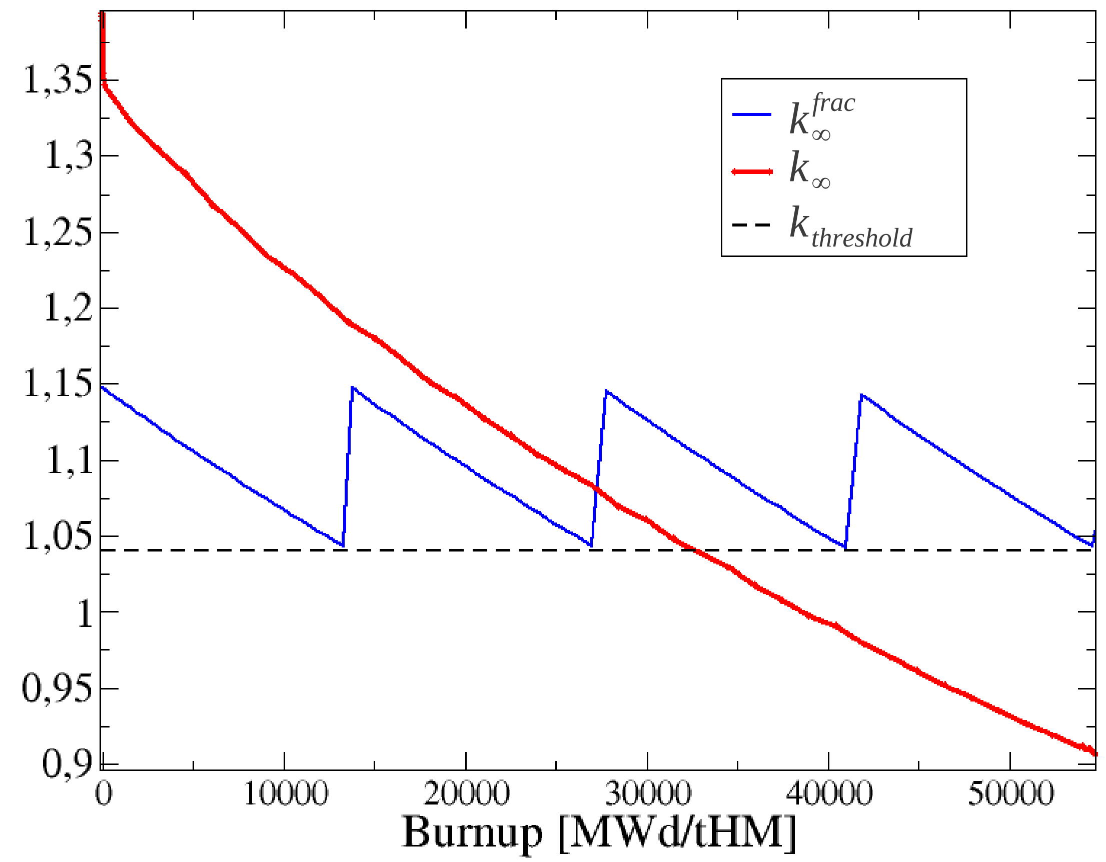
T : Irradiation time (N loading/unloading)



The averaged k_∞ model

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_{i=0}^{N-1} k_\infty (t + \frac{iT}{N})$$

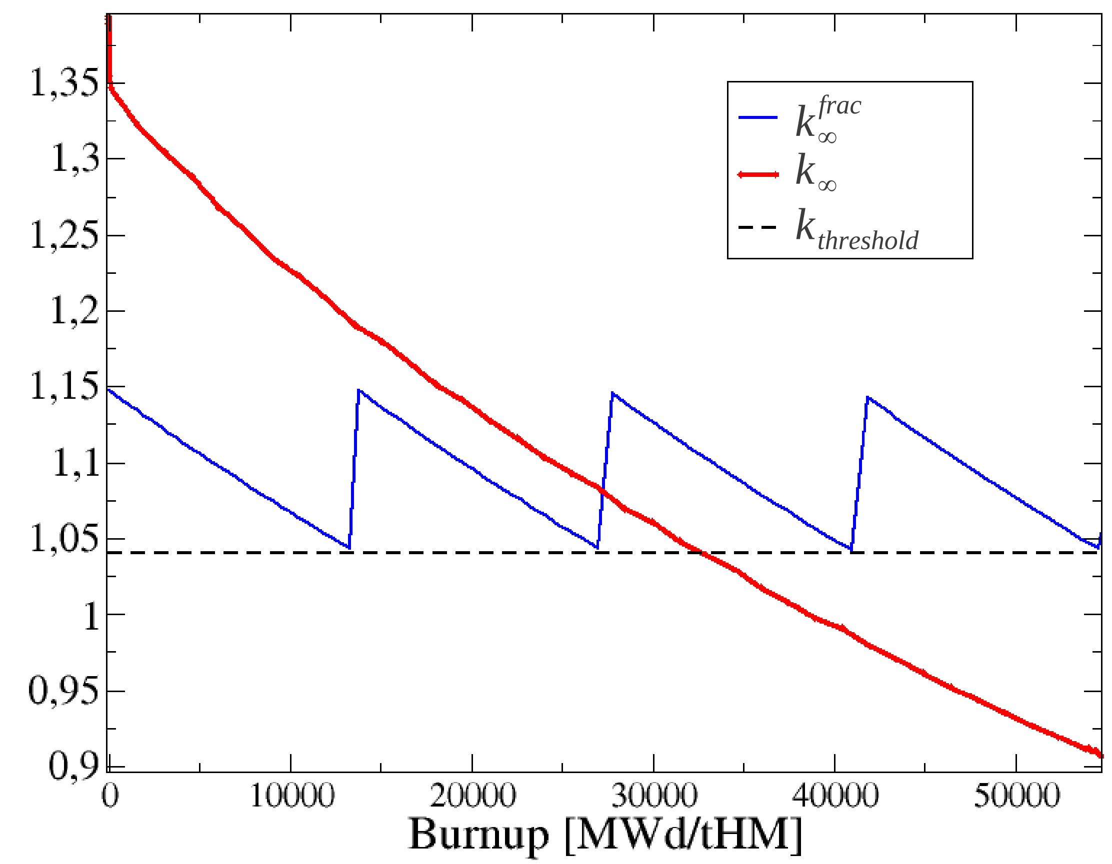
T : Irradiation time (N loading/unloading)



The averaged k_∞ model

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_{i=0}^{N-1} k_\infty (t + \frac{iT}{N})$$

T : Irradiation time (N loading/unloading)



<u>The reactor has to verify</u> :

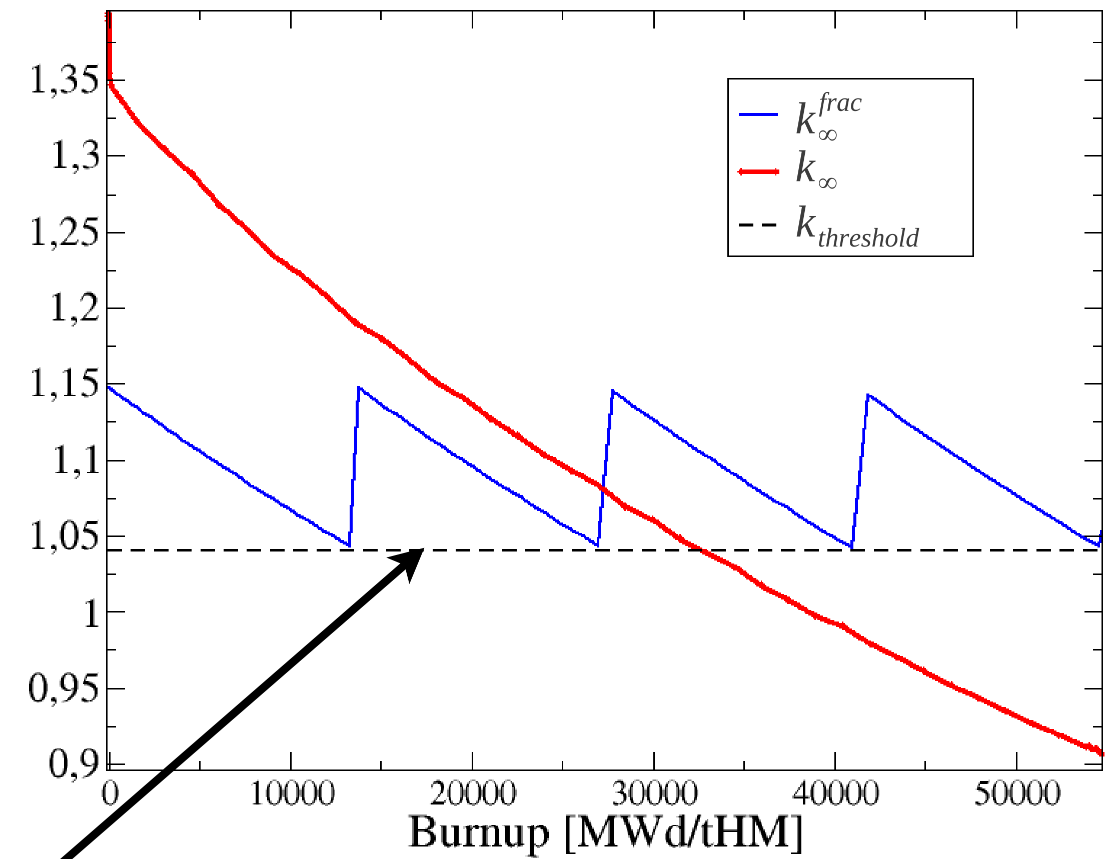
$$k_{eff} = 1 \ \forall t \ \in [0, T]$$

$$\rightarrow < k_\infty >^{batch} = 1 + Leaks + CRods$$

The averaged k_∞ model

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_{i=0}^{N-1} k_\infty(t + \frac{iT}{N})$$

T : Irradiation time (N loading/unloading)



The reactor has to verify :

$$k_{eff} = 1 \;\; \forall t \;\; \in [0, T]$$

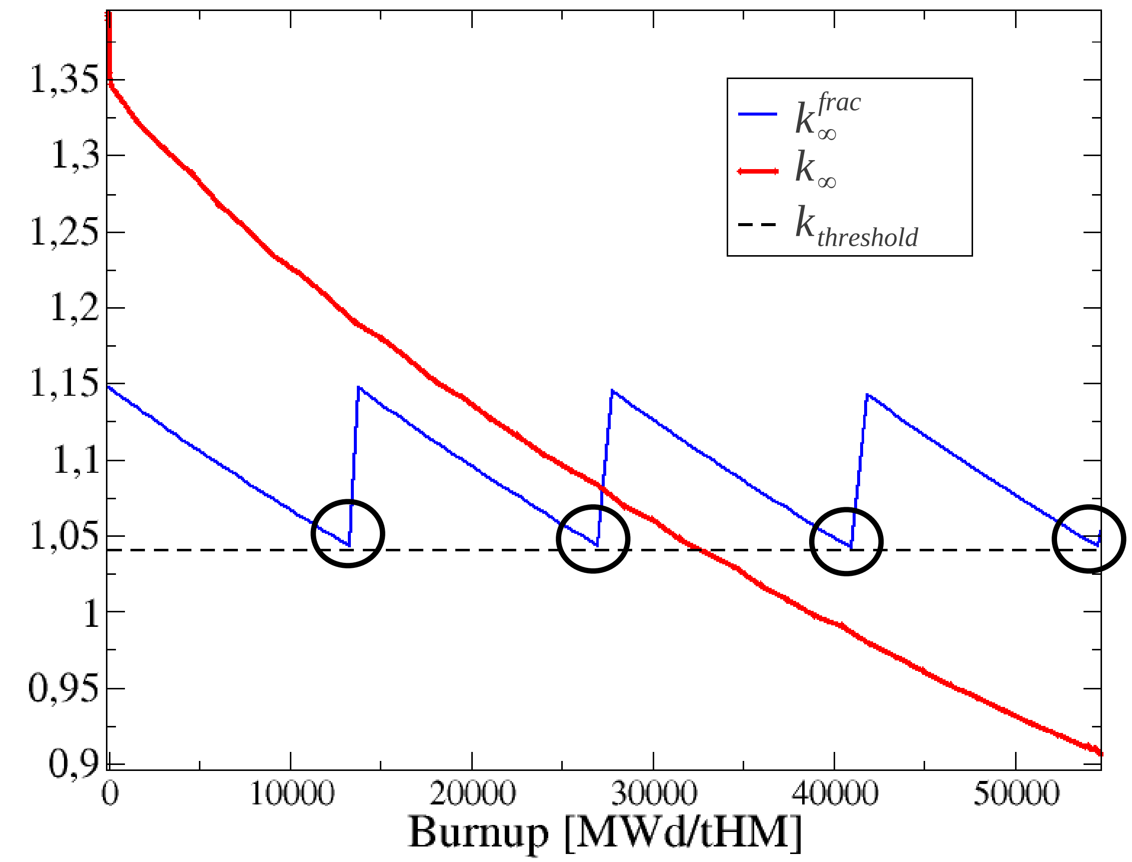
$$\rightarrow < k_\infty >^{batch} = 1 + Leaks + CRods$$

$$< k_\infty >^{batch} = k_{threshold}$$

The averaged k_∞ model

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_{i=0}^{N-1} k_\infty(t + \frac{iT}{N})$$

T : Irradiation time (N loading/unloading)



The reactor has to verify :

$$k_{eff} = 1 \;\; \forall t \;\; \in [0, T]$$

$$\rightarrow < k_\infty >^{batch} = 1 + Leaks + CRods$$

$$< k_\infty >^{batch} = k_{threshold}$$

as [boron] is kept constant in simulation the criticality conditions is :

$$< k_\infty >^{batch} (T/N) = < k_\infty >^{batch} (2T/N) = ... = k_{threshold}$$

The averaged k_∞ model

The plutonium content in a fresh fuel is such as the criticality condition is verified :

Criticality condition :

$$< k_\infty >^{batch} (T/N) = k_{threshold}$$

The time T is the maximum irradiation time (Burnup) a given fuel can reach : BU_{max} (Given by user)

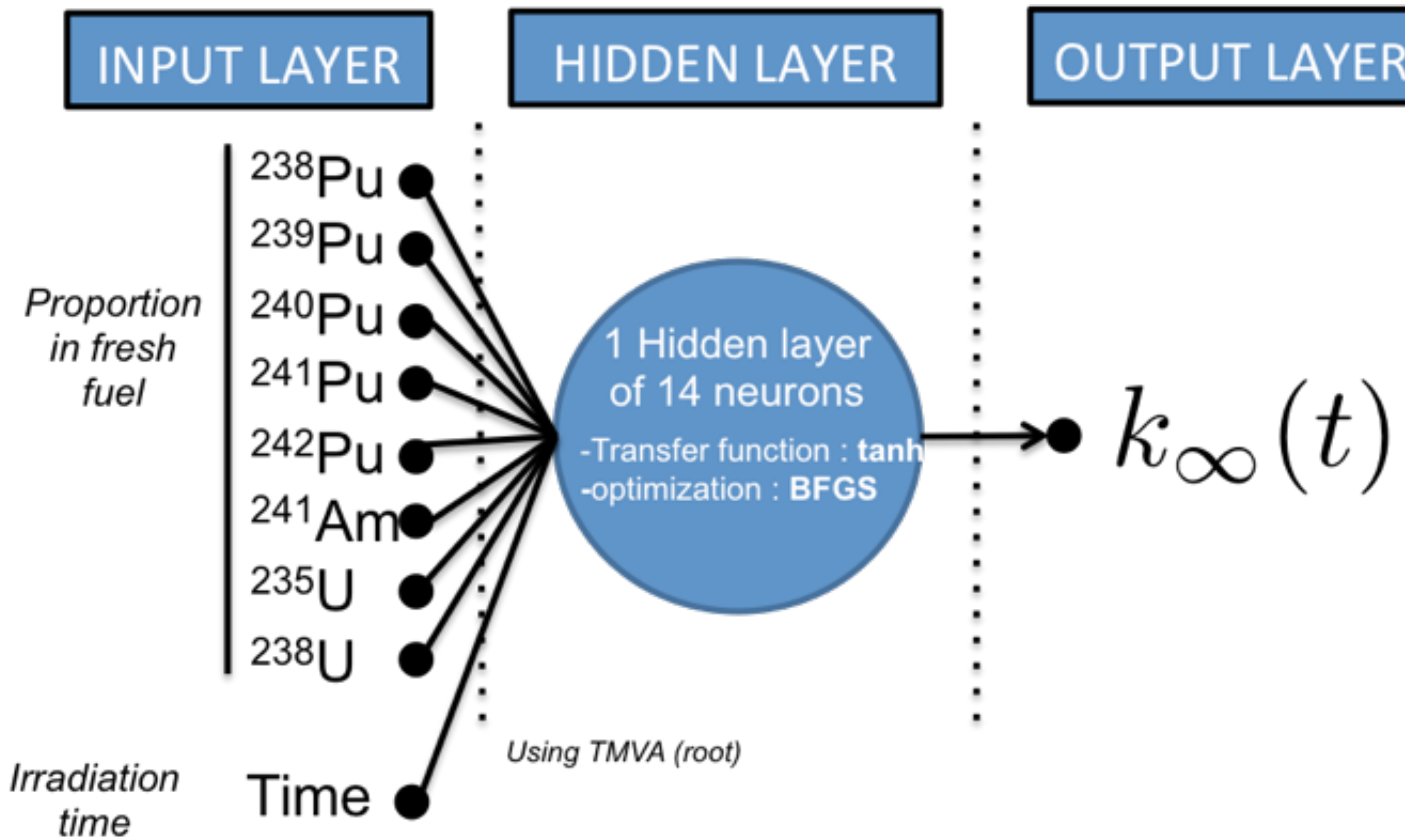
Method used to find maximum average discharged burnup in candu & pwr :

Nuttin, A., et al. "Comparative analysis of high conversion achievable in thorium-fueled slightly modified CANDU and PWR reactors." Annals of Nuclear Energy 40.1 (2012): 171-189.

This model involves to build :

- → a $k_\infty(t)$ predictor : $\widehat{k}_\infty(t)$
- → an algorithm to find the Pu content according to the criticality condition

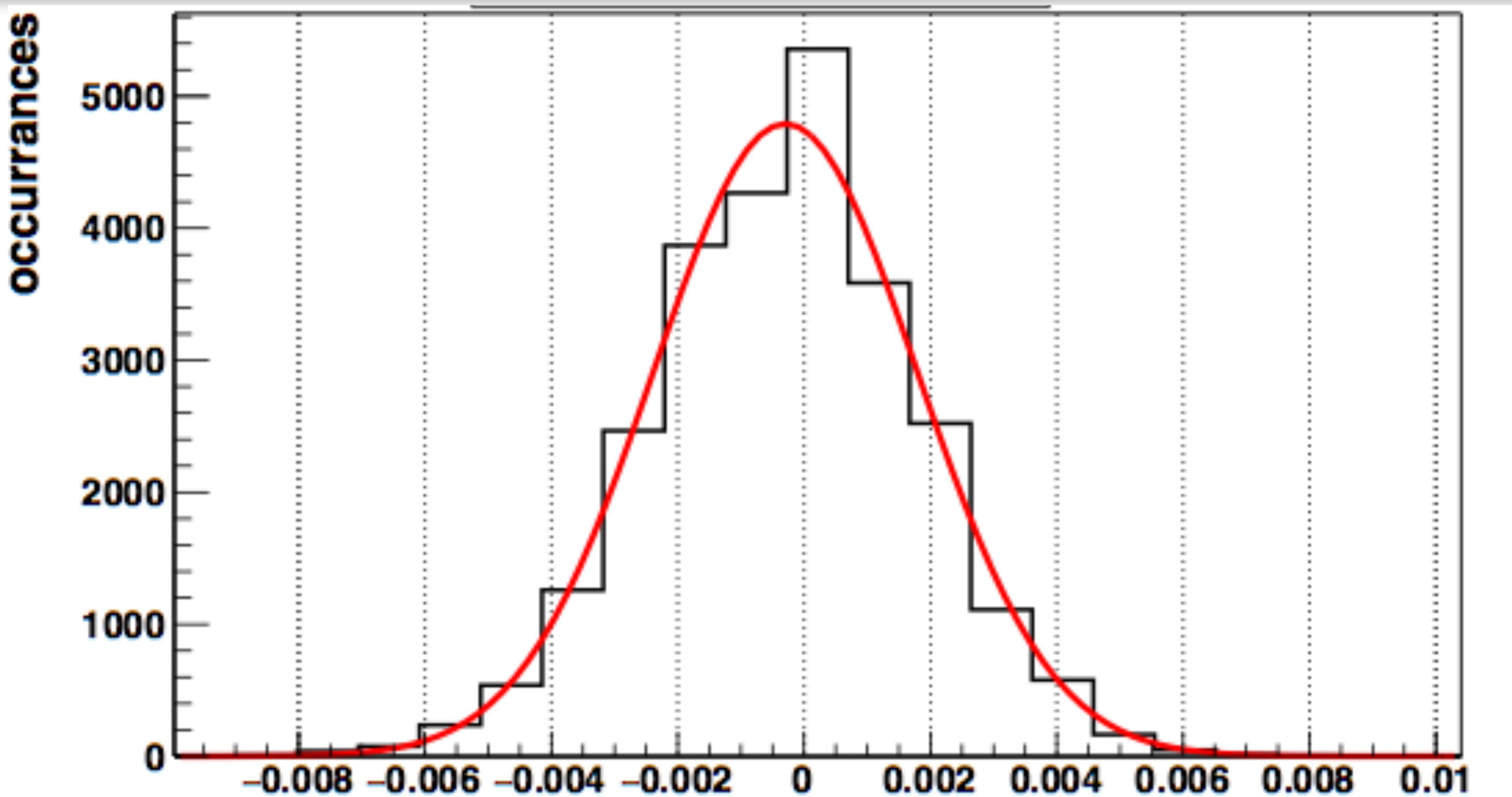
$\widehat{k_\infty(t)}$: Artificial neural network (ANN) approach



<u>Total number of events :</u>

5000 depletion calculation & 11 time step each : 55000 events.
Train with half (randomly picked) and tested with the other half

$\widehat{k}_\infty(t)$ Results of the ANN testing procedure

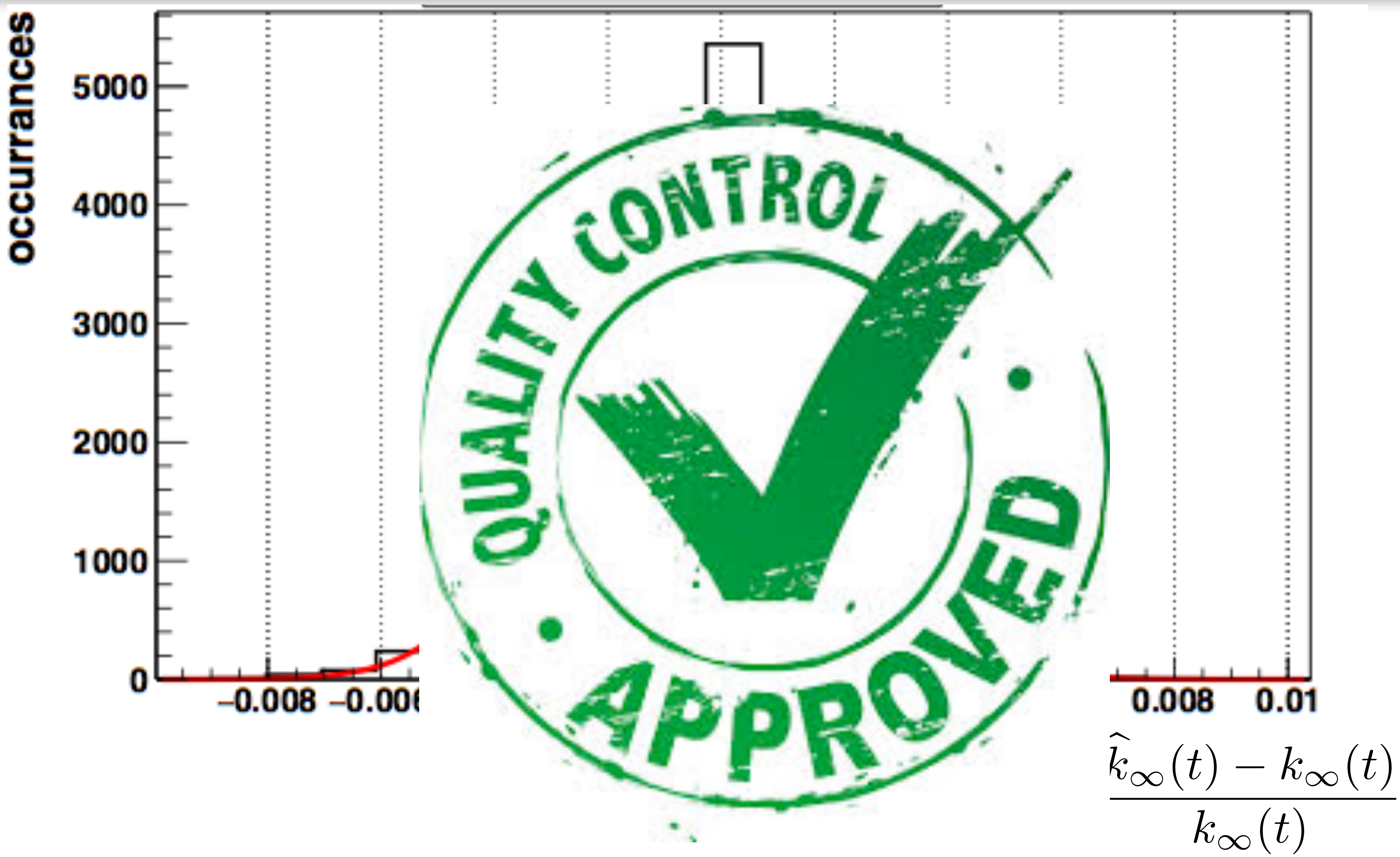


$$\frac{\widehat{k}_\infty(t) - k_\infty(t)}{k_\infty(t)}$$

Mean error : 30 pcm
Std. dev. on error: 200 pcm

MCNP Statistical error : ~ 150 pcm

$\widehat{k}_\infty(t)$ Results of the ANN testing procedure



Mean error : 30 pcm
Std. dev. on error: 200 pcm

MCNP Statistical error : ~ 150 pcm

- Model algorithm

Which Pu content

$$\boxed{\%Pu =}$$

Which Pu content to reach a given burnup

$$\boxed{\%Pu = f(BU_{target},}$$

Inputs

BU_{target} : wanted burnup

• Model algorithm

Which Pu content to reach a given burnup with a given batch management,

$$\boxed{\%Pu = f(BU_{target}, N,}$$

Inputs

BU_{target} : wanted burnup

N : # of batches

- Model algorithm

Which Pu content to reach a given burnup with a given batch management, knowing isotopic compositions of Pu et U

$$\boxed{\%Pu = f(BU_{target}, N, \overrightarrow{Pu}, \overrightarrow{U}, k_{th})}$$

{

}

Inputs

BU_{target} : wanted burnup

N : # of batches

$\overrightarrow{Pu}$: Fissile composition

$\overrightarrow{U}$: Fertile composition

- Model algorithm

Which Pu content to reach a given burnup with a given batch management, knowing isotopic compositions of Pu et U

$$\boxed{\%Pu = f(BU_{target}, N, \overrightarrow{Pu}, \overrightarrow{U}, k_{th})}$$

```
{
    %Pu = P0            //Initialization

    do
    {
      varying BU until <k>(BU/N)= kth
      using eq 1 & 2 : BUmax
}
```

Inputs

BU_{target} : wanted burnup

N : # of batches

$\overrightarrow{Pu}$: Fissile composition

$\overrightarrow{U}$: Fertile composition

eq 1 :

$$< k_\infty >^{batch} (t) = \frac{1}{N}\sum_i^N k_\infty(t + \frac{iT}{N})$$

eq 2 : $\widehat{k_\infty}(BU, \vec{N})$

$$\vec{N} = \%Pu\vec{Pu} + (1 - \%Pu)\vec{U}$$

- Model algorithm

Which Pu content to reach a given burnup with a given batch management, knowing isotopic compositions of Pu et U

$$\boxed{\%Pu = f(BU_{target}, N, \overrightarrow{Pu}, \overrightarrow{U}, k_{th})}$$

```
{
    %Pu = P0              //Initialization

    do
    {
      varying BU until <k>(BU/N)= kth
      using eq 1 & 2 : BUmax

      if(BUmax > BUtarget)
        decrease %Pu    //%Pu too high
      else
        increase %Pu    //%Pu too low
    }
}
```

Inputs

BU_{target} : wanted burnup

N : # of batches

$\overrightarrow{Pu}$: Fissile composition

$\overrightarrow{U}$: Fertile composition

eq 1 :

$$< k_\infty >^{batch} (t) = \frac{1}{N}\sum_i^N k_\infty(t + \frac{iT}{N})$$

eq 2 : $\widehat{k}_\infty(BU, \vec{N})$

$$\vec{N} = \%Pu\vec{Pu} + (1 - \%Pu)\vec{U}$$

- Model algorithm

Which Pu content to reach a given burnup with a given batch management, knowing isotopic compositions of Pu et U

$$\boxed{\%Pu = f(BU_{target}, N, \overrightarrow{Pu}, \overrightarrow{U}, k_{th})}$$

```
{
    %Pu = P0            //Initialization

    do
    {
      varying BU until <k>(BU/N)= kth
      using eq 1 & 2 : BUmax

      if(BUmax > BUtarget)
        decrease %Pu    //%Pu too high
      else
        increase %Pu    //%Pu too low
    }
    while ( BUmax != BUtarget )   //accept %Pu or
                                   continue looping
   return %Pu
}
```

Inputs

BU_{target} : wanted burnup

N : # of batches

$\overrightarrow{Pu}$: Fissile composition

$\vec{U}$: Fertile composition

eq 1 :

$$< k_\infty >^{batch} (t) = \frac{1}{N} \sum_i^N k_\infty (t + \frac{iT}{N})$$

eq 2 : $\widehat{k}_\infty (BU, \vec{N})$

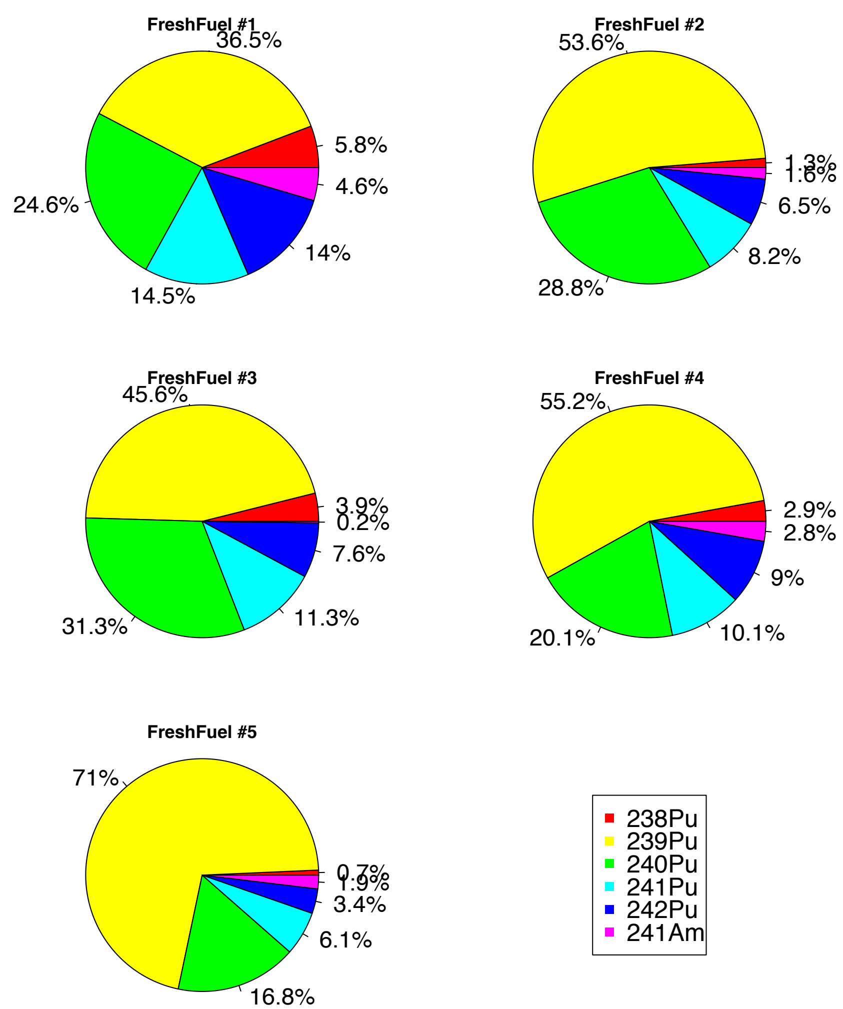
$$\vec{N} = \%Pu\vec{Pu} + (1 - \%Pu)\vec{U}$$

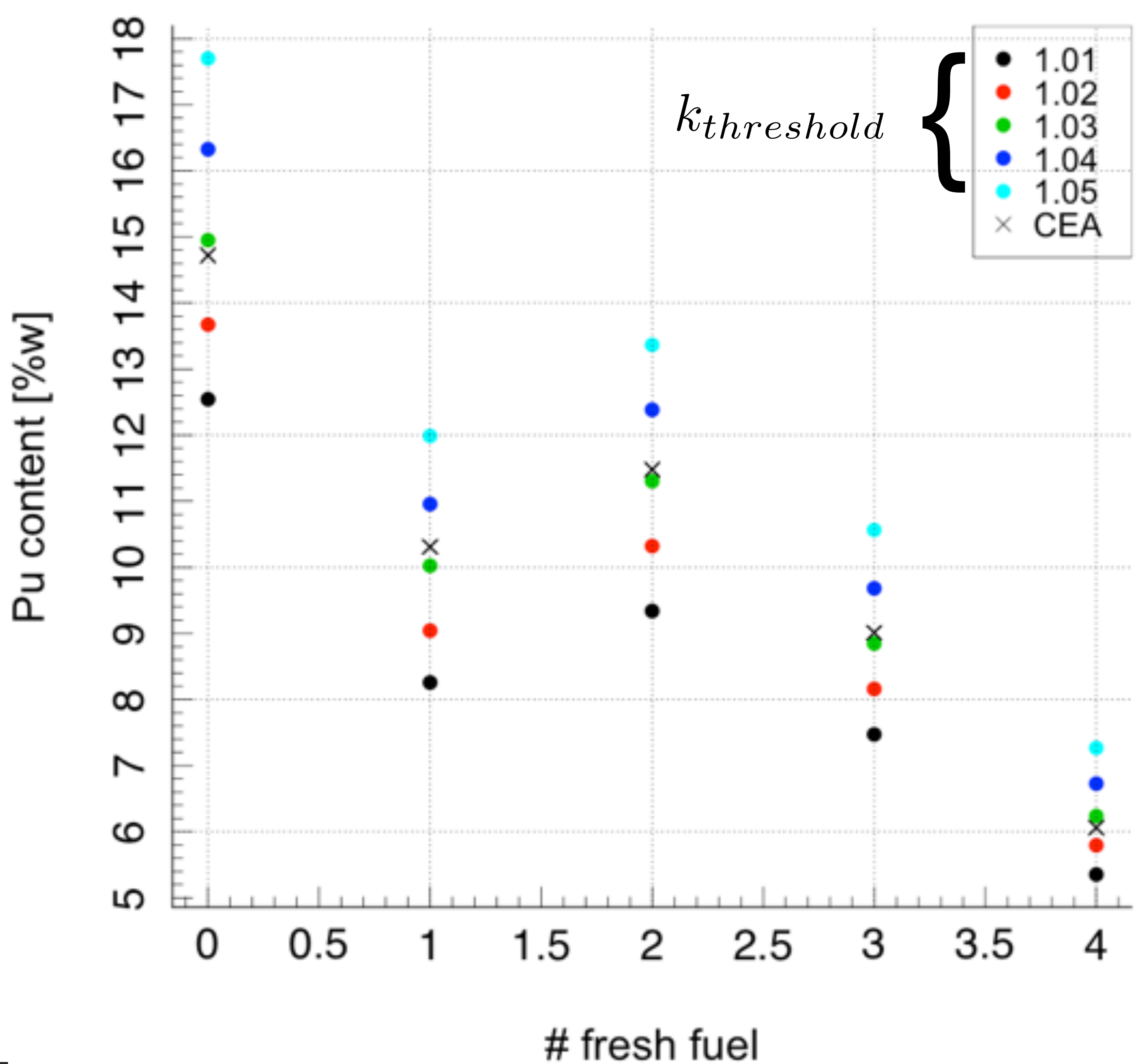
Benchmark with COSI PWR-MOX Model

Work accomplished in the framework of a CEA/CNRS collaboration (2014)

Pu content predicted by the two codes for 5 different Pu isotopy :

$$N = 3$$
$$k_{threshold} = [1.01\ ;\ 1.05] \qquad \overrightarrow{Pu} = variable$$
$$BU_{target} = 50\,GWd/t \qquad \overrightarrow{U} = 0.25\%\ {}^{235}U$$





Conclusions & outlooks

Mean cross sections interpolation :

➡ Good precision of the ANN approach on <σ> prediction. For PWR-MOX :
-(n,f) : std. dev. on error < 1 %
-(n,γ) : std. dev. on error < 2 %
-(n,2n) : std. dev. on error < 5 % (for actinides)
Discrepancy due to Monte Carlo statistical error

➡ Performances in a depletion calculation:
Error (compared as MURE calc.) ~3% max @ EOC for main actinides
Computational time ~30 s

This methodology is used for all reactor description in CLASS

Conclusions & outlooks

Fresh fuel construction

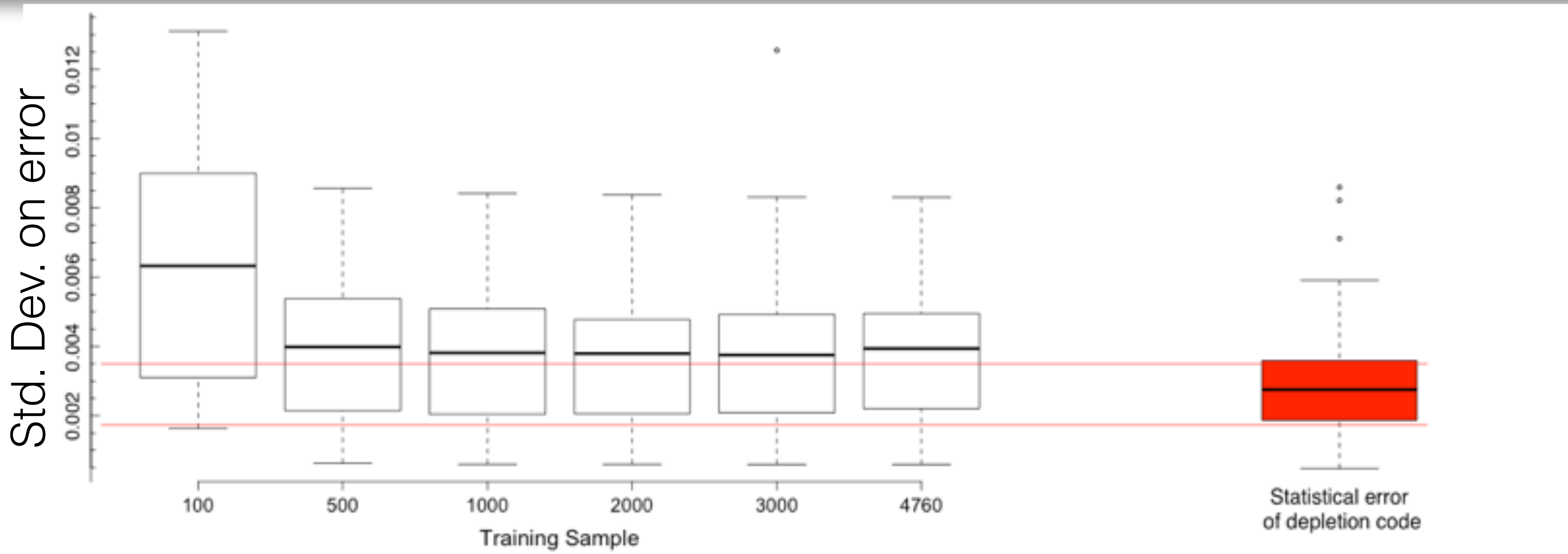
➡ PWR-MOX model :

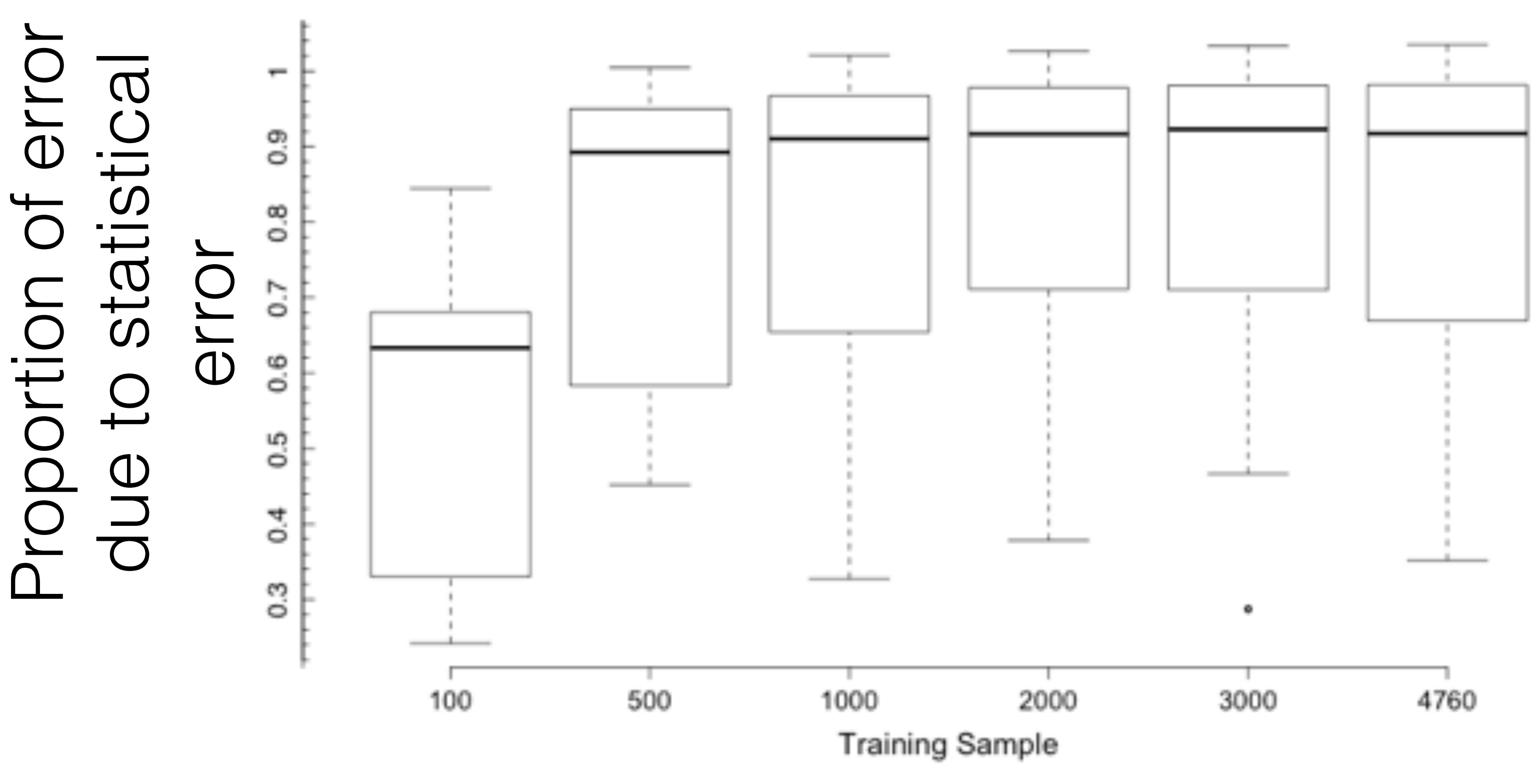
- Based on batch averaged $k_{infinite}$.
- $k_{infinite}(t)$ prediction using MLP (std. dev. 200pcm)
- Flexible : possibility to change $k_{threshold}$ and number of batches
- Pu content prediction close to COSI(CEA) with $k_{threshold}$ = 1.03

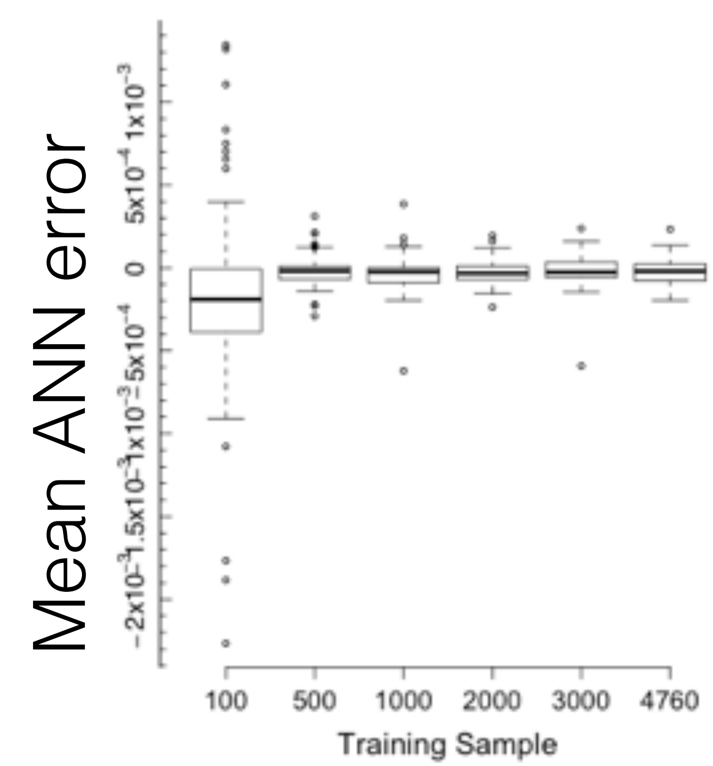
➡ List of CLASS equivalence models :

Model Name	Reactor	Fuel	Spectrum	Target Burnup	Batch Management	Regression Method	Predicted value	Input parameters
PWR_POL_UO2	PWR	UOX	Thermal	any	fixed	Polynomial	% 235U	Target burnup
PWR_MLP_MOX	PWR	MOX	Thermal	any	fixed	MLP	Pu content	Target burnup
PWR_MLP_MOXAm	PWR	MOX-Am	Thermal	any	fixed	MLP	(Pu+Am) content	Target burnup
MLP_Kinf	any (not breeder)	any		any	any	MLP	K_∞	Target BU / Kthreshold/ N Batch
PWR_MLP_MOXEUS	PWR	MOX EUS	Thermal	any	any	MLP	K_∞	Target BU / Kthreshold/ N Batch/ Max Pu Content
FBR_BakerAndRoss	any (breeder)	MOX	fast	fixed	none	Linear (pert. theory)	Pu content	reactivity weights
FBR_MLP_Keff	any (breeder)	any	fast	fixed	none	MLP	keff(t=0)	none

Testing performances : Fission prediction







Testing performances : (n,2n) prediction