

IRSN

INSTITUT
DE RADIOPROTECTION
ET DE SÛRETÉ NUCLÉAIRE

Enhancing nuclear safety

Second level of criticality modelling: beyond k-effective calculations, NCS begins...

G. CAPLIN, M. DULUC, Y. RICHET



ICNC 2015
of International Cooperation



September 13-17, Charlotte, USA

Foreword

- What this workshop is not: A code/tool demonstration/training
- What this workshop is: An introduction to the concept called “2nd level of criticality modelling”
 - Proof of concept: case study with IRSN’s R&D tool “Prométhée” (scripting tool for any kind of calculation codes)
<http://bit.ly/1gXTVkd>
- Target Audience: NCS assessors or designers using criticality codes for their work
- Presentations and practical cases study

NCS issues: parameters & problems

- Prevention of a nuclear criticality accident is based on the strict control of clearly identified parameters

“Nuclear Criticality Safety is achieved by controlling one or more parameters of the system within subcritical limits and by allowances for process contingencies”

ANS-8.1-2014

➔ Controlled parameters

- Other parameters (those which are not controlled) are assumed to take any value within a “credible” range

➔ Free / Nuisance parameters

NCS issues: parameters & problems

■ Usual parameters affecting criticality:

Mass	Absorption	Geometry
Interaction	Concentration	Moderation
Enrichment	Reflection	Volume
Chemical form	Density	Heterogeneity
...		

➔ Variable parameters through normal and credible abnormal conditions

Some of them will be “controlled parameters” others will be “nuisance parameters”

NCS issues: parameters & problems

Usual problems to solve to achieve NCS:

- Single-parameter subcritical limits

Table 1

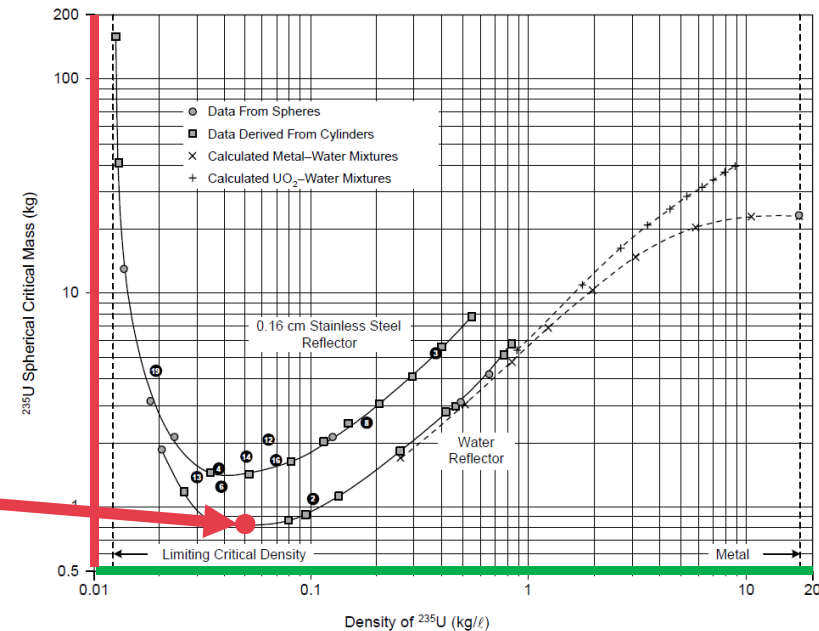
Single-Parameter Subcritical Limits for Uranium and Plutonium Solutions, Reflected by an Effectively Infinite Thickness of Water

Fissile Solute	Subcritical Limit						
	Mass of Fissile Nuclide (kg)	Diameter of Cylinder of Solution (cm)	Thickness of Slab of Solution (cm)	Volume of Solution (L)	Density of Fissile Nuclide (g/L)	Atomic Ratio ^a of Hydrogen to Fissile Nuclide	Areal Density of Fissile Nuclide (g/cm ²)
$^{233}\text{UO}_2\text{F}_2$	0.54	10.5	2.5	2.8	10.8	2390	0.35
$^{233}\text{UO}_2(\text{NO}_3)_2$	0.55	11.7	3.1	3.6	10.8	2390	0.35
$^{235}\text{UO}_2\text{F}_2$	0.76	13.7	4.4	5.5	11.6	2250	0.40
$^{235}\text{UO}_2(\text{NO}_3)_2$	0.78	14.4	4.9	6.2	11.6	2250	0.40
$^{239}\text{Pu}(\text{NO}_3)_4$	0.48	15.4	5.5	7.3	7.3	3630	0.29

from LA-12808

Note: Chemical form is also a controlled parameter

1 controlled parameter (mass)
+ 1 nuisance parameter (moderation)



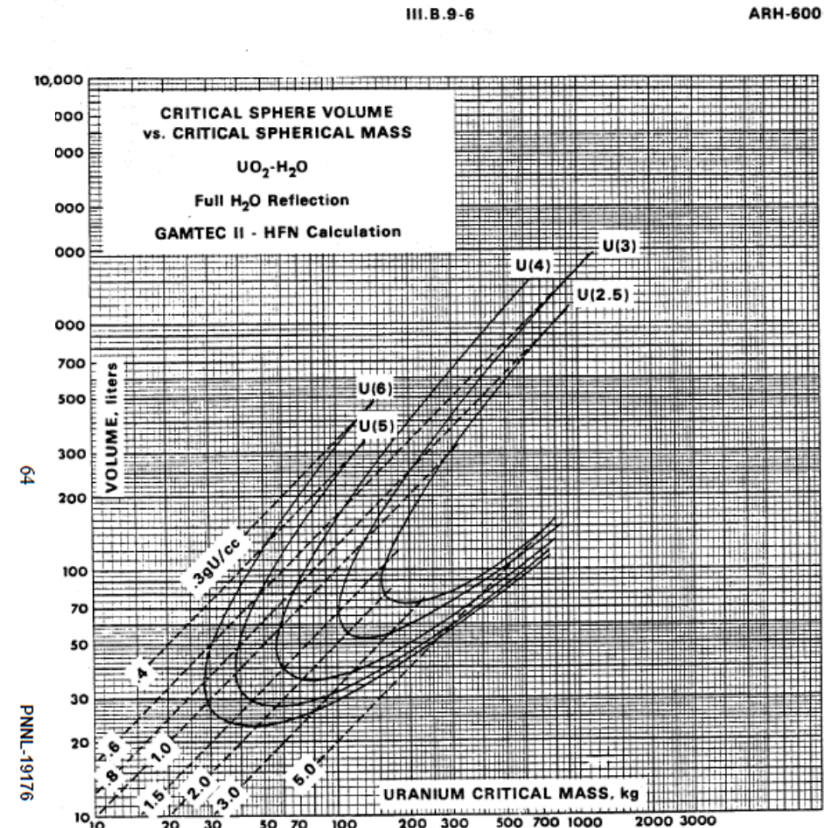
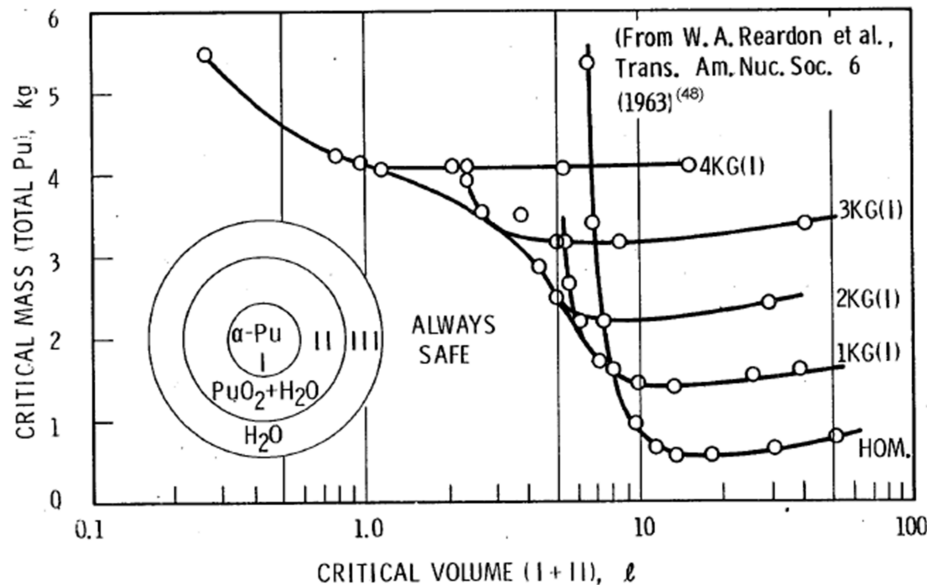
from LA-13638:2000

NCS issues: parameters & problems

Usual problems to solve to achieve NCS:

- Single-parameter subcritical limits
- Multi-parameters subcritical limits

Figure 24. Computed Critical Mass (Total ^{239}Pu in $\text{PuO}_2 + \text{H}_2\text{O}$ Solution)



NCS issues: parameters & problems

■ Usual problems to solve to achieve NCS:

- Single-parameter subcritical limits
- Multi-parameters subcritical limits
- Sensitivity to one (or more) parameter(s)
- Design (process or experiment)
- Change of an existing process
- Peer review of an NCS evaluation

➔ **Resort to computer codes** (or handbooks, hand calculations methods,...)

And also: assessing contingencies, defining what is “credible”, defining the adequate safety margin, implementing the controls, etc., etc.

NCS issues: parameters & problems

■ Resort to computer codes → modelling the problem

- Multiple times since parameters vary
- Using code requirements
 - input data of codes are **dimensions and compositions** while parameters such as **mass, moderator volume,...** are derived from those inputs
 - codes main output is **k_{eff}** while the sought limits are **mass, diameter,...** (k_{eff} is generally an input of NCS problems *via* the USL or margins of Δk)

NCS issues: parameters & problems

NRC FORM 100-1 (9-2000) 10 CFR 71		U.S. NUCLEAR REGULATORY COMMISSION			
CERTIFICATE OF COMPLIANCE FOR RADIOACTIVE MATERIAL PACKAGES					
1. CERTIFICATE NUMBER	2. REVISION NUMBER	3. DOCKET NUMBER	4. PACKAGE IDENTIFICATION NUMBER	PAGE	PAGES
9196			USA/9196/B(U)F-96	2	4

5.(a) Packaging (continued)

(3) Drawings

The Model No. UX-30 packaging is fabricated in accordance with Drawing No. C sheets 1 through 3, Rev.

(b) Contents

(1) Type and form of material

A. Unirradiated uranium, in the form of UF_6 with a U-235 mass percentage not to exceed 5 weight percent.

B. Reprocessed uranium, in the form of UF_6 with a U-235 mass percentage not to exceed 5 weight percent. The fission product gamma activity shall not exceed 10^5 Ci/gm. The alpha activity from neptunium and plutonium shall be less than 10^5 Ci/gm.

(2) Maximum quantity of material per package

5,020 pounds UF_6 contained in an ANSI Standard N14.1 30B or 30C cylinder.
The maximum H/U atomic ratio for the UF_6 is 0.088.
The total activity in the package may not exceed 10^5 A2.

(c) Criticality Safety Index (CSI)

Criticality safety index for the UX-30 overpack containing a standard ANSI N14.1 30B cylinder 5.0

Criticality safety index for the UX-30 overpack containing a standard ANSI N14.1 30C cylinder 0.0

Criticality safety index for the UX-30 overpack is not applicable to non-fissile or fissile-excepted contents.

6. The ANSI standard 30B, 30-inch diameter UF_6 cylinder, must be fabricated, inspected, tested and maintained in accordance with a) American National Standard N14.1-2001 or an earlier version of ANSI N14.1 in effect at the time of fabrication or b) American National Standard N14.1-2001 or an earlier version of ANSI N14.1 in effect at the time of fabrication and ISO 7195:1993(F). Cylinders must be fabricated in accordance with Section VIII, Division I, of the ASME (American Society of Mechanical Engineers) Boiler and Pressure Vessel Code and be ASME Code stamped.

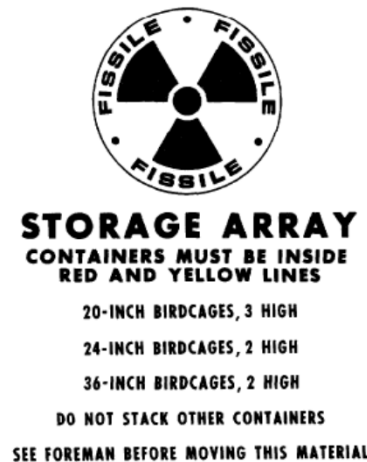


Fig. 10-12. Oak Ridge Y-12 Plant white metal sign with yellow circle, magenta fissile symbol, and black letters.

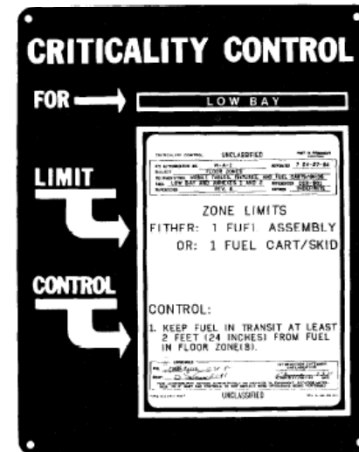


Fig. 10-13. UNC Naval Products blue metal sign with yellow letters and a white computer-generated control statement page in a clear plastic holder.

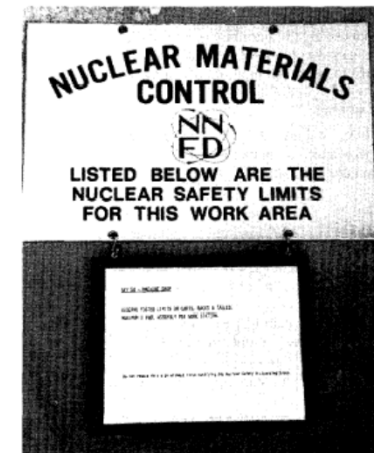


Fig. 10-14. Babcock and Wilcox Naval Nuclear Fuel Plant gold anodized aluminum sign with black letters over a paper sign in a glass-covered frame.

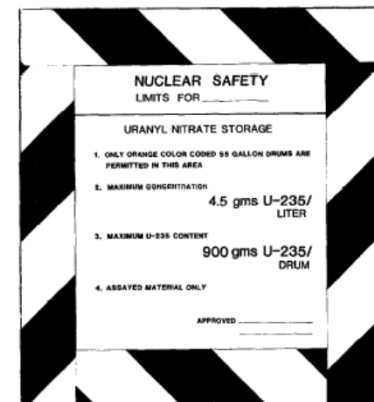


Fig. 10-15. Babcock and Wilcox Nuclear Material Division orange paper sign with black letters in a clear plastic holder bordered with orange and black tape.

from R.A. Knief "NCS – Theory and Practice"

NCS issues: parameters & problems

■ Resort to computer codes → modelling the problem

- Multiple times since parameters vary
- Using code requirements
 - input data of codes are **dimensions and compositions** while parameters such as **mass, moderator volume,...** are derived from those inputs
 - codes main output is **k_{eff}** while the sought limits are **mass, diameter,...** (k_{eff} is generally an input of NCS problems *via* the USL or margins of Δk)

→ Gap between **codes modelling** and **NCS problems**

NCS issues: parameters & problems

■ Gap between codes modelling and NCS problems

- **Many tools exist to bridge the gap...**
(home-made scripts, MS Excel™ spreadsheets, goal seeking modules, GUI,...)
- **...but generally limited to specific applications**

→ Purpose of this workshop: to share our reflections about generalizing these issues, which lead into a “2nd level of modelling”

Contents

■ Criticality safety issues: safety parameters & types of problems

■ Case study: presentation and implementation overview

■ Bridging the gap between codes modelling and NCS problems

- Basic & advanced parametrization
- Application to the case study: hands on parametric study

■ “2nd level criticality modelling”

- Discovering through the case study for typical NCS problems
- Overview of mathematics behind
- Discussions about the concept

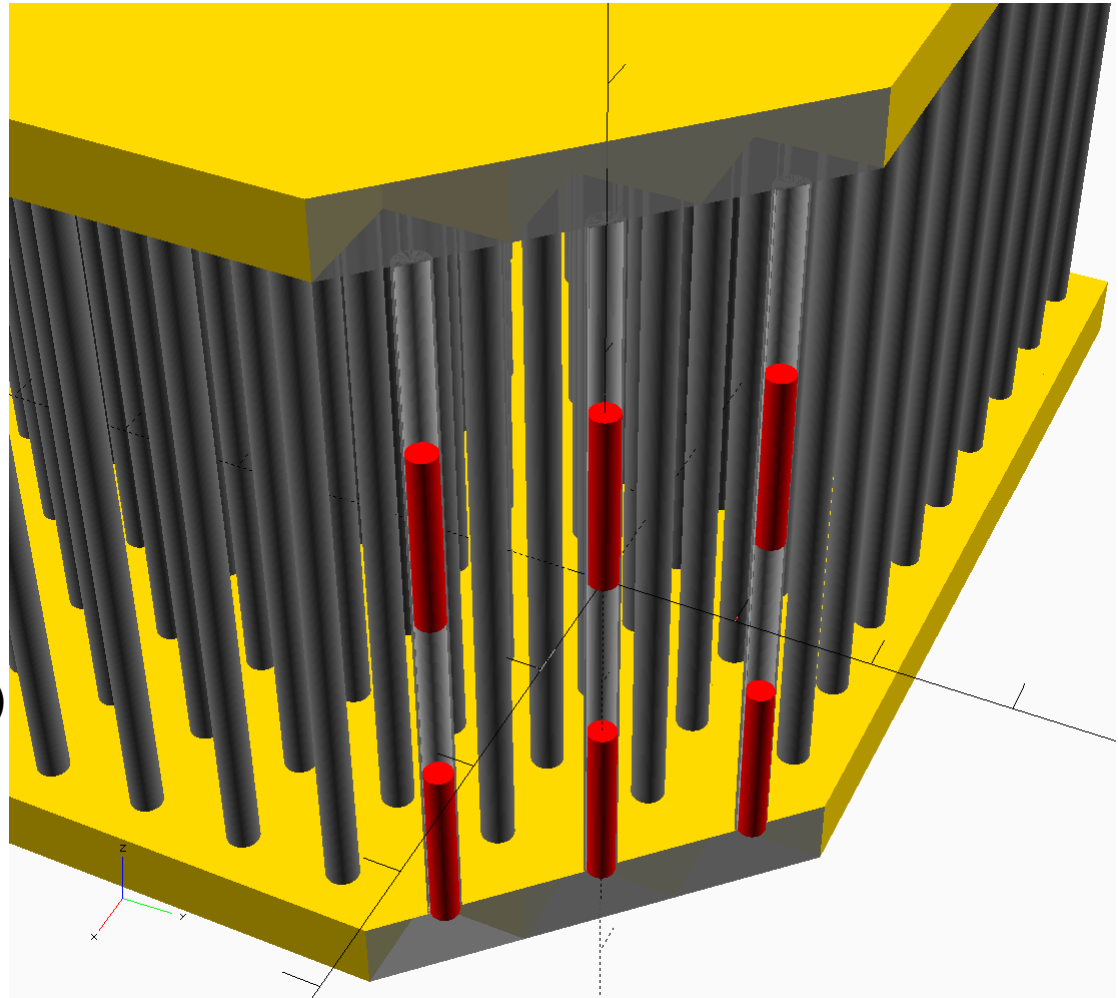
Contents

- Criticality safety issues: safety parameters & types of problems
- Case study: presentation and implementation overview
- Bridging the gap between codes modelling and NCS problems
 - Basic & advanced parametrization
 - Application to the case study: hands on parametric study
- “2nd level criticality modelling”
 - Discovering through the case study for typical NCS problems
 - Overview of mathematics behind
 - Discussions about the concept

Case Study: Presentation

■ Storage of PuO_2 powder

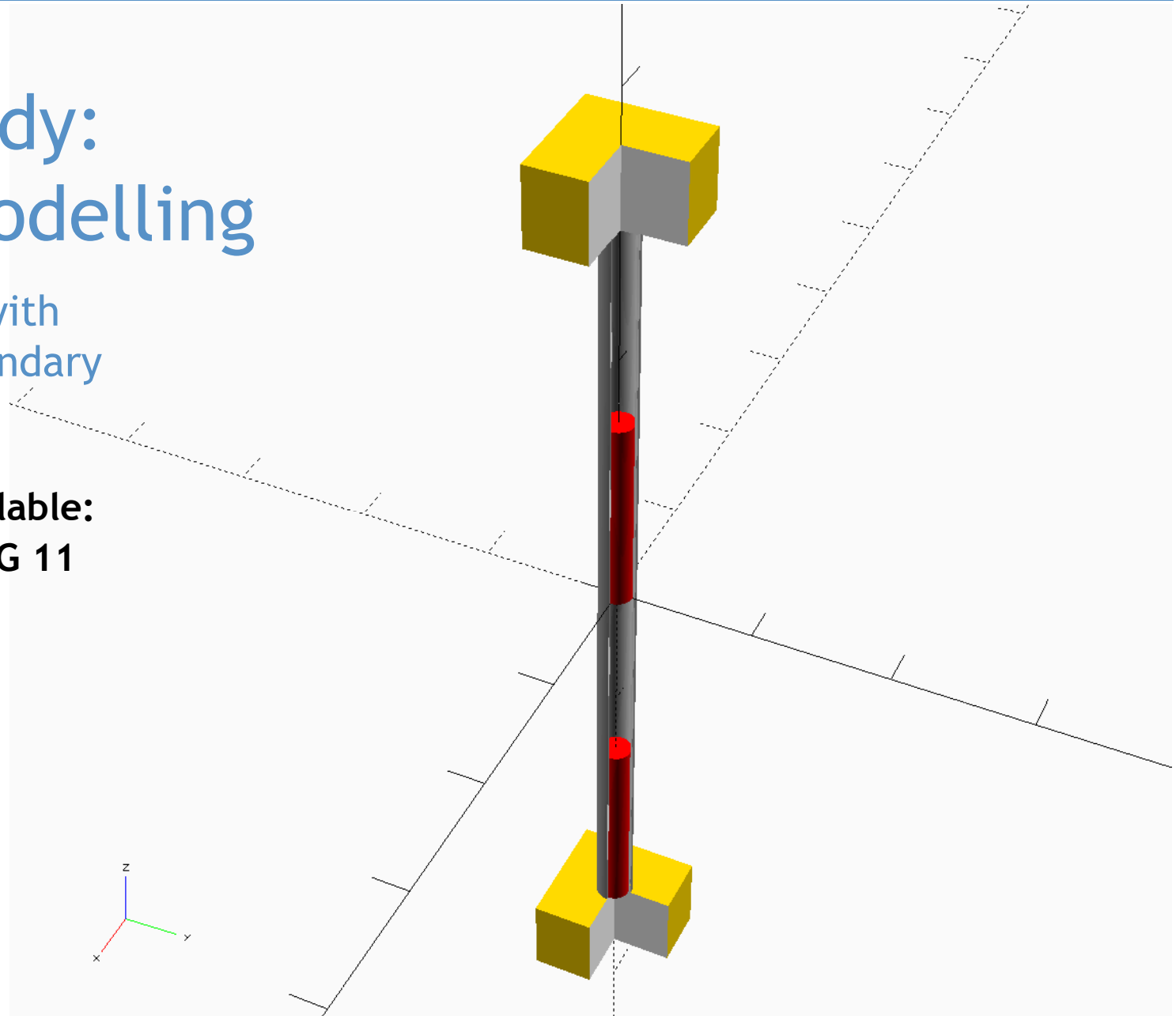
- Infinite planar array of tubes containing 2 cans of powder
- Controlled parameters:
 - **Geometry** (tubes, cans)
 - **Interaction** (spacing)
 - **Mass** (Pu per can)
 - **Moderation**
(water content inside PuO_2)
 - **Density**
(max PuO_2 density)
 - **Enrichment**
($^{239,240,241}\text{Pu}$ contents)



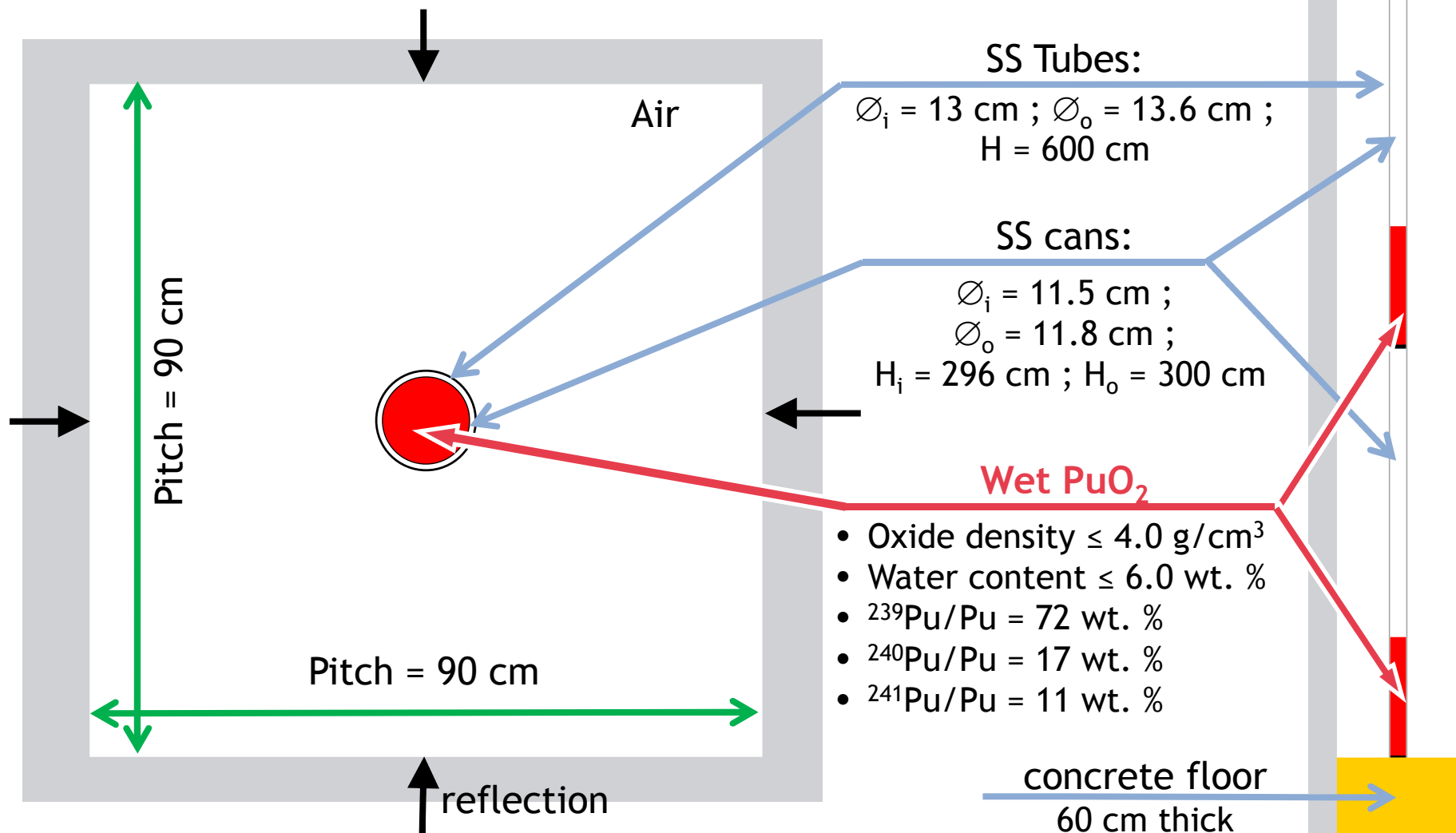
Case Study: Codes Modelling

■ A single unit with
reflective boundary
conditions


→ 2 models available:
MORET 5 and COG 11

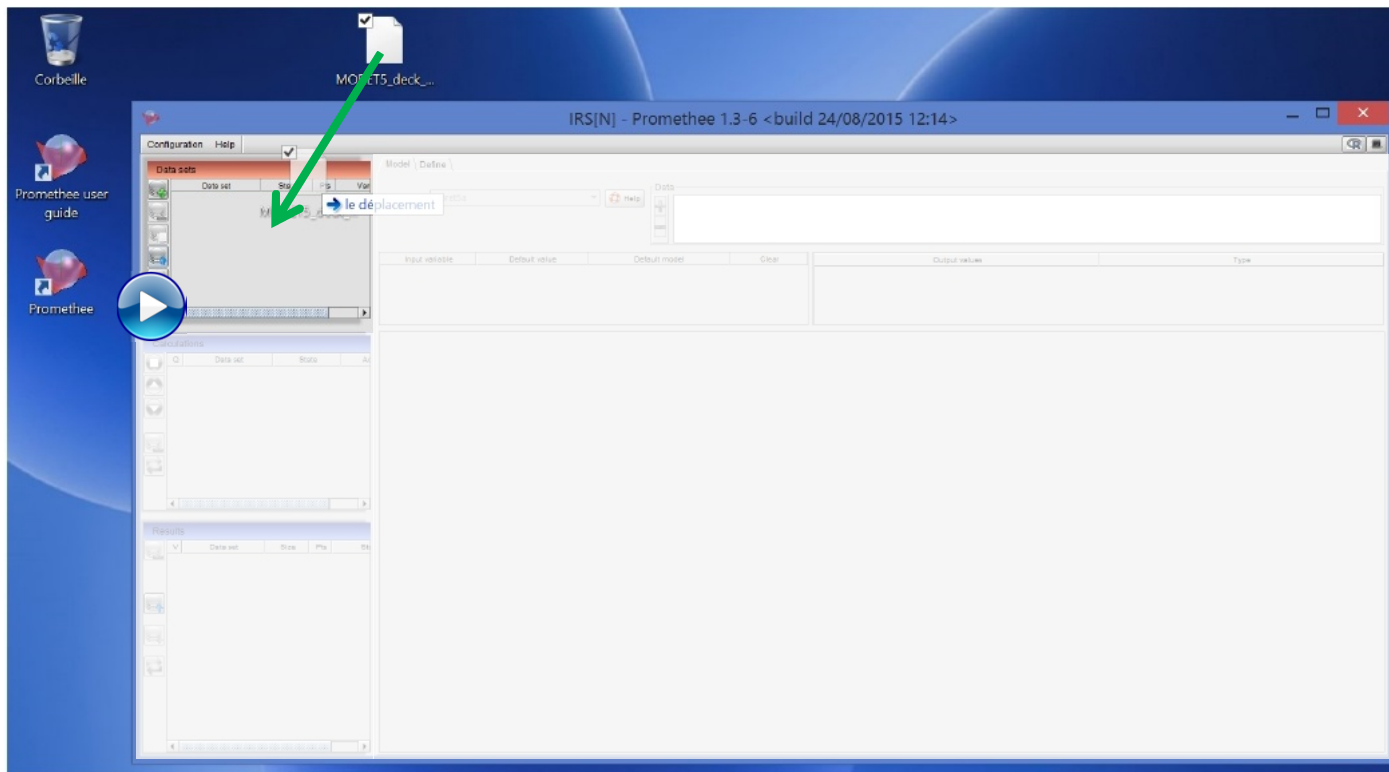


Case Study: Codes Modelling



Case Study: Starting point

- Models are given for a Pu mass per can = 50 kg
- Stage 0: Run the calculation - drag&drop the deck and click “run” 



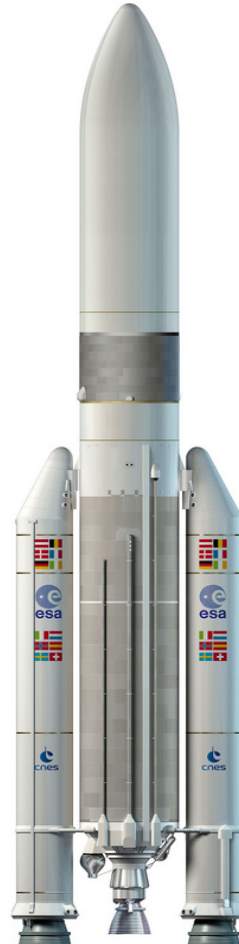
$k_{\text{eff}} \sim 1.07... !!!$

“2nd level of criticality modelling”



Single calculation: 1st level of criticality modelling

“2nd level of criticality modelling”



Managing multiple calculations

Single calculation: 1st level of criticality modelling

Contents

- Criticality safety issues: safety parameters & types of problems
- Case study: presentation and implementation overview
- Bridging the gap between codes modelling and NCS problems
 - Basic & advanced parametrization
 - Application to the case study: hands on parametric study
- “2nd level criticality modelling”
 - Discovering through the case study for typical NCS problems
 - Overview of mathematics behind
 - Discussions about the concept



Exercise #1: “design”

■ Stage 1: Search for the pitch so that $k_{\text{eff}} = 0.95$

➔ “Basic” parametrization

- To set a variable parameter to be interpreted by Prométhée

COG	MORET
<i>%param_name</i>	<i>\$param_name</i>

- To make the pitch varying

COG (in PLANES #1,2,3&4)	MORET (in TYPES #1&2)
replace 45 by <i>%half_pitch</i>	replace 45 by <i>\$half_pitch</i>
replace -45 by <i>-%half_pitch</i>	

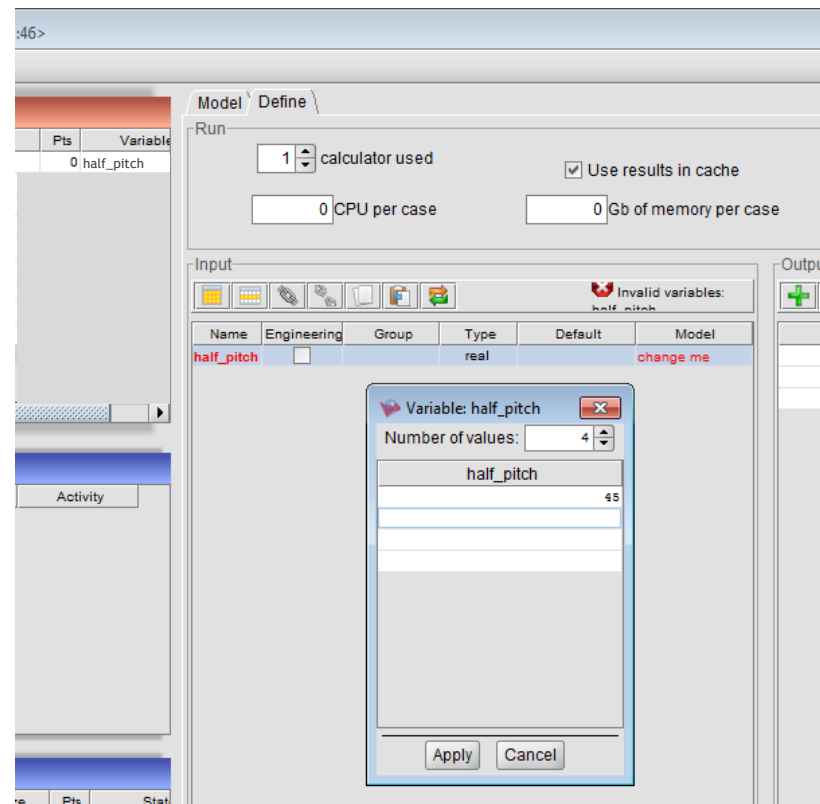
Exercise #1: “design”



■ Stage 1: Search for the pitch so that $k_{\text{eff}} = 0.95$

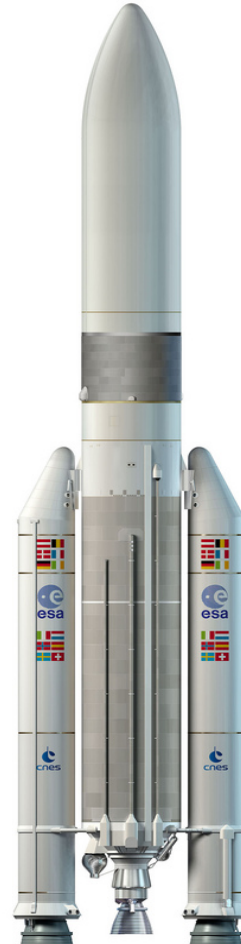
- Run multiple (not too many!) calculations with different “half_pitch” values

Try to find $k_{\text{eff}} = 0.95$



For « half_pitch » = cm, $k_{\text{eff}} \sim 0.95$

“2nd level of criticality modelling”

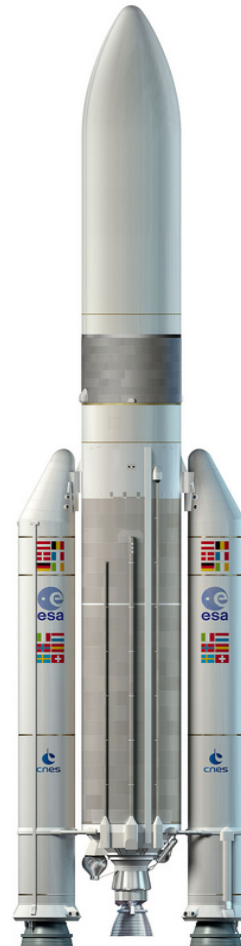


Basic parametrization

Managing multiple calculations

Single calculation: 1st level of criticality modelling

“2nd level of criticality modelling”



Basic parametrization

Self-consistent input decks
NCS oriented

Managing multiple calculations
code oriented

Single calculation: 1st level of criticality modelling



Exercise #1: “design”

■ Stage 2: Preliminary remark about basic parametrization

- Parameter implemented: “half_pitch” = code-centric view

→ Replace “half_pitch” by $\frac{\text{"pitch"}}{2}$

COG	MORET
replace <i>%half_pitch</i> by <i>@{%pitch_cm / 2}</i>	replace <i>\$half_pitch</i> by <i>@{\$pitch_cm / 2}</i>

Note: use of such expressions also useful for managing different units between **codes requirements** (cm, g/cm³) and **problem specifications** (mm, in, g/L,...)

Exercise #1: “design”

■ Stage 2: Search for the Pu mass so that $k_{\text{eff}} = 0.95$ (pitch = 90 cm)

- Pu mass is not a direct input of the code
(the direct input is the filling height of the cans)

$$m_{\text{Pu}} = \underbrace{\pi * \frac{\phi_{i(\text{cans})}^2}{4} * H_{\text{filling}}}_{\text{Volume}} * d_{\text{PuO}_2} * \% \frac{\text{Pu}}{\text{PuO}_2}$$

* density * Pu fraction in PuO₂



Exercise #1: “design”

■ Stage 2: Search for the Pu mass so that $k_{\text{eff}} = 0.95$ (pitch = 90 cm)

- Pu mass is not a direct input of the code
(the direct input is the filling height of the cans)

$$m_{\text{Pu}} = \pi * \frac{\phi_{i(\text{cans})}^2}{4} * H_{\text{filling}} * d_{\text{PuO}_2} * \% \frac{\text{Pu}}{\text{PuO}_2}$$

1. Set H_{filling} as a parameter and calculate the mass *via* a spreadsheet

→ code-centric approach

or

2. Define the relationship between m_{Pu} and H_{filling} directly in the input deck

→ Advanced parametrization NCS oriented



Exercise #1: “design”

■ Stage 2: Search for the Pu mass so that $k_{\text{eff}} = 0.95$ (pitch = 90 cm)

- **Advanced parametrization:** Declaration of a formula

Add the following line anywhere:

COG	<code>\$@: H_fiss_cm <- function(Pu_mass_kg) { 1000 * Pu_mass_kg / (pi/4 * 11.5^2 * 4.0 * 0.88211) }</code>
MORET	<code>*@: H_fiss_cm <- function(Pu_mass_kg) { 1000 * Pu_mass_kg / (pi/4 * 11.5^2 * 4.0 * 0.88211) }</code>

Where 1000 is a conversion factor due to chosen units (cm, kg and g/cm³)
 11.5 is $\varnothing_{i(cans)}$ (in cm)
 4.0 is d_{PuO_2} (in g/cm³)
 0.88211 is the weight fraction of Pu in PuO₂



Exercise #1: “design”

■ Stage 2: Search for the Pu mass so that $k_{\text{eff}} = 0.95$ (pitch = 90 cm)

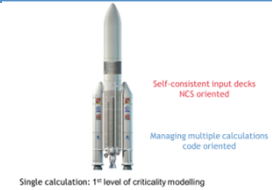
- **Advanced parametrization:** Call of a formula

Change both lower and upper can filling

COG	in PLANE #12	replace -161.5726 by @{-298 + H_fiss_cm(%Pu_mass_kg)}
	in PLANE #22	replace 138.4274 by @{ 2 + H_fiss_cm(%Pu_mass_kg)}
MORET	in TYPE #9	replace 68.2137 by @{H_fiss_cm(\$Pu_mass_kg)/2}
	in VOLUME #91	replace -229.7863 by @{-298 + H_fiss_cm(\$Pu_mass_kg)/2}
	in VOLUME #92	replace 70.2137 by @{ 2 + H_fiss_cm(\$Pu_mass_kg)/2}

➔ **Clarification of the deck:** ‘-161.5726’ has less meaning than ‘-298+H_fiss_cm’

Exercise #1: “design”



■ Stage 2: Search for the Pu mass so that $k_{\text{eff}} = 0.95$ (pitch = 90 cm)

- Run multiple (not too many!) calculations with different “Pu_mass_kg” values (and a “pitch_cm” value = 90 cm): Try to find $k_{\text{eff}} = 0.95$

For « Pu_mass_kg » = kg & « pitch » = 90 cm, $k_{\text{eff}} \sim 0.95$

Exercise #1: “design”



■ Stage 2 (advanced parametrization) remarks (1)

- The source position is in the middle of the PuO_2 VOLUME:
Beware of parameters dependences on all code inputs
→ Change the source position as-well-as the fissile height



COG	in <i>CRITICALITY</i> <i>nsource</i> replace -229.7863 by @{-298 + H_fiss_cm(%Pu_mass_kg)/2} replace 70.2137 by @{ 2 + H_fiss_cm(%Pu_mass_kg)/2}
MORET	in <i>SOURCE</i> replace -229.7863 by @{-298 + H_fiss_cm(\$Pu_mass_kg)/2} replace 70.2137 by @{ 2 + H_fiss_cm(\$Pu_mass_kg)/2}



Exercise #1: “design”

■ Stage 2 (advanced parametrization) remarks (2)

- **Beware of formulae results formatting**
(in particular when codes require integers or
if results need scientific formatting to be significant, eg. **0.00001** \neq **1.499E-5**)

➔ **Necessity of adding a format specification**

COG	in PLANE #12, PLANE #22 and <i>CRITICALITY nsource</i> add 0.0000 in @{...} such as @{... 0.0000}
MORET	in TYPE #9, VOLUME #91, VOLUME #92 and <i>SOURCE</i> add 0.0000 in @{...} such as @{... 0.0000}

Exercise #1: “design”



■ Stage 2 (advanced parametrization) remarks (3)

- To clarify an input deck with multiple parameters: **Benefit of being able to declare (and comment) the parameters**

Add comment lines anywhere:	
COG	<i>\$ Storage pitch (cm): %pitch_cm</i> <i>\$ Plutonium mass per can (kg): %Pu_mass_kg</i>
MORET	<i>* Storage pitch (cm): \$pitch_cm</i> <i>* Plutonium mass per can (kg): \$Pu_mass_kg</i>

Exercise #1: “design”



■ Stage 2 (advanced parametrization) remarks (4)

- For limiting errors (and for clarity), benefit of declaring “constants” for once (eg. the weight fraction of Pu in PuO₂)

	Add the following line anywhere:	In “H_fiss_cm” formula declaration
COG	<code>\$@: Pu_in_PuO2 = 0.88211</code>	Replace 0.88211 by Pu_in_PuO2
MORET	<code>*@: Pu_in_PuO2 = 0.88211</code>	Replace 0.88211 by Pu_in_PuO2



Exercise #1: “design”

■ Stage 2 (advanced parametrization) remarks (5)

- For complex formulae, great benefit of checking its results
(limiting errors when the deck is used by someone else or after a long period)

$$\pi * \frac{\phi_{i(cans)}^2}{4} * H_fiss_cm(32) * d_{PuO_2} * \% \frac{Pu}{PuO_2} =? 32$$

Add the following line anywhere:

COG	<code>\$@? round(pi/4*11.5^2 *H_fiss_cm(32)*4.0*Pu_in_PuO2) == 32000</code>
MORET	<code>*@? round(pi/4*11.5^2 *H_fiss_cm(32)*4.0*Pu_in_PuO2) == 32000</code>

If the test is not passed, running calculation is not authorized

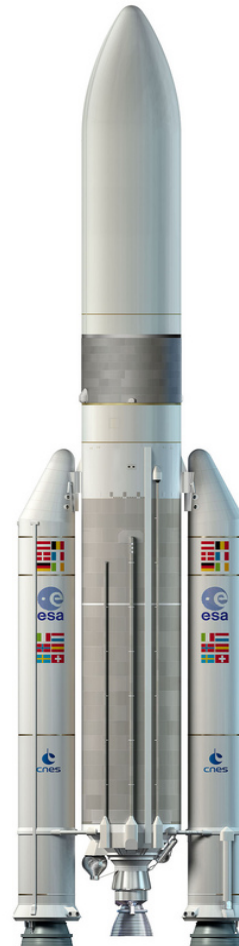
“2nd level of criticality modelling”

Advanced parametrization

Self-consistent input decks
NCS oriented

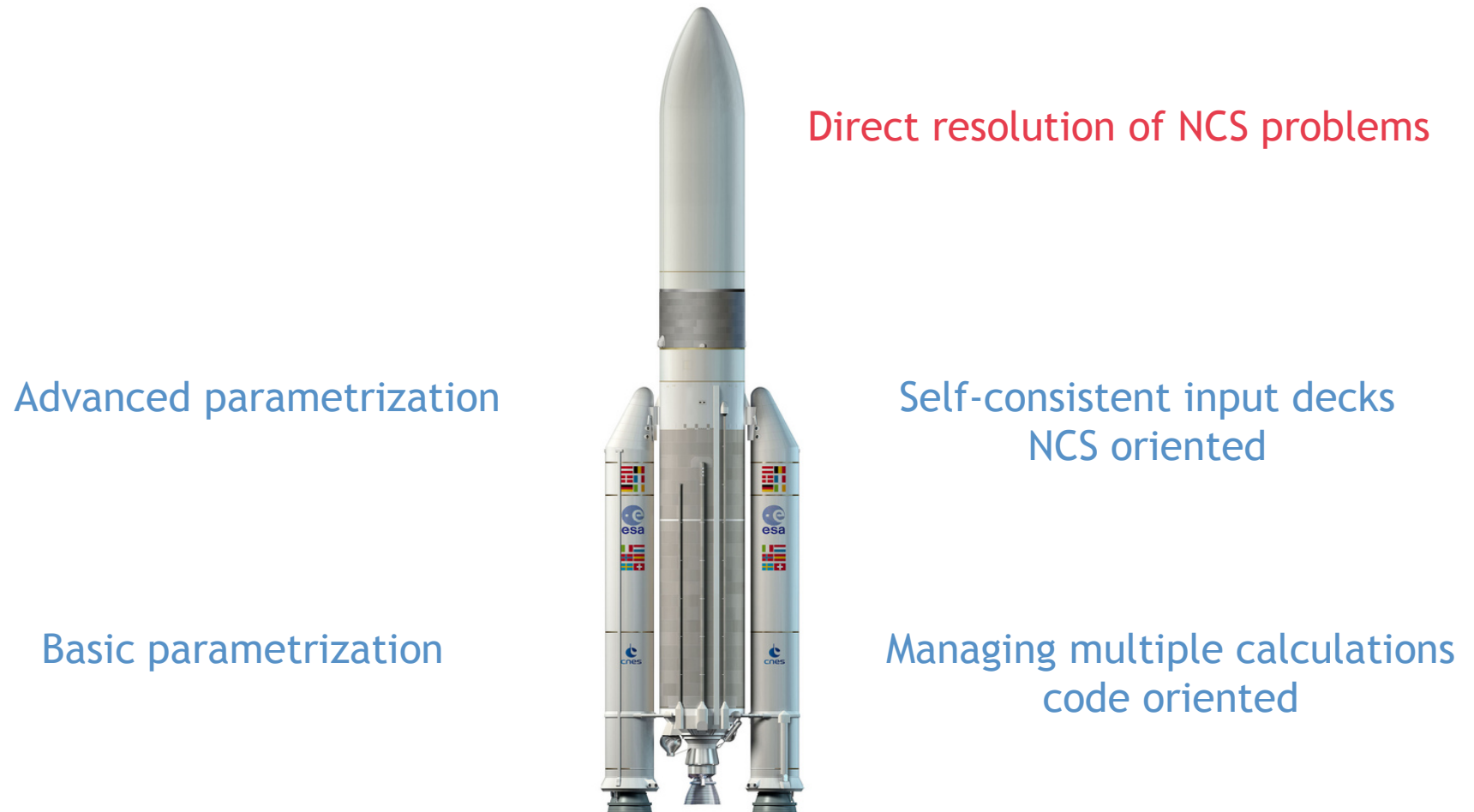
Basic parametrization

Managing multiple calculations
code oriented



Single calculation: 1st level of criticality modelling

“2nd level of criticality modelling”



Single calculation: 1st level of criticality modelling

Contents

- Criticality safety issues: safety parameters & types of problems
- Case study: presentation and implementation overview
- Bridging the gap between codes modelling and NCS problems
 - Basic & advanced parametrization
 - Application to the case study: hands on parametric study
- “2nd level criticality modelling”
 - Discovering through the case study for typical NCS problems
 - Overview of mathematics behind
 - Discussions about the concept

Exercise #1: “design”



■ Stage 3: “Full design” what are the {pitch, m_{Pu} } where $k_{eff} \leq 0.95$?

- Resort to a Design of Experiments

- Multiple strategies possible

- Randomly!
 - Estimation from the 2 known results
 - Pitch-by-pitch (or mass-by-mass)
 - Full factorial plan (n_p pitches x n_m masses)

Share the work!

Pitch (cm)	Pu mass (kg)
40	
50	
60	
70	
80	
90	36
100	
109	50
120	

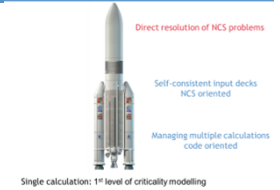
Exercise #1: “design”



■ Stage 3: “Full design” what are the $\{\text{pitch}, m_{\text{Pu}}\}$ where $k_{\text{eff}} \leq 0.95$?

- Resort to an algorithm (deck with advanced parametrization is essential)
 - Multiple strategies possible (dichotomy, genetic algorithms,...)
 - Example of one algorithm we have found well adapted for NCS problems:
SUR
 - Select “engineering” for both “pitch_cm” and “Pu_mass_kg”
 - Set the lower and upper bounds (e.g. [13.6,150] and [10,50])
 - Select the “inversion / SUR” algorithm
 - Specify the target k_{eff} value : Tlim = 0.95 (default : NULL)
 - Run the project
 - Look at the results...

Exercise #1: “design”



■ Stage 3: “Full design” what are the $\{\text{pitch}, m_{pu}\}$ where $k_{\text{eff}} \leq 0.95$?

Stepwise Uncertainty Reduction (SUR) algorithm (inversion algorithm)

1. calculate k_{eff} for first few points (pitch, m_{pu}), randomly chosen plus bounds
 2. generate a surrogate function K_{eff} :
interpolating previous $\{k_{\text{eff}}(\text{pitch}, m_{pu})\}$ calculations
 3. search the next most “valuable” points (pitch, m_{pu})
 4. perform these k_{eff} calculations, stack with previous $\{k_{\text{eff}}(\text{pitch}, m_{pu})\}$
- repeat steps 2-3-4 as needed to get a reliable definition of the safety area

- Key details:

- Surrogate function: allows to estimate $\text{Prob}[K_{\text{eff}}(\text{pitch}, m_{pu}) > 0.95]$
- “valuable” point: helps to predict where $k_{\text{eff}} > 0.95$

Exercise #1: “design”

Surrogate function $K_{\text{eff}}(\text{pitch}, m_{\text{Pu}})$

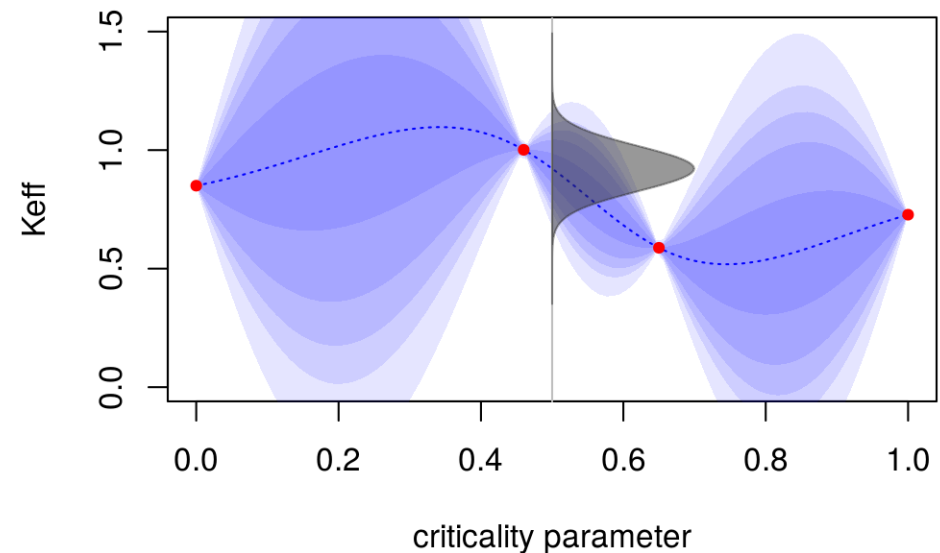
Random function

Interpolates measures
(even imprecise measures)

Gaussian predictor mean, sd:

$$E[K_{\text{eff}}(\text{pitch}, m_{\text{Pu}})] = \text{mean}(\text{pitch}, m_{\text{Pu}})$$

$$\text{Var}[K_{\text{eff}}(\text{pitch}, m_{\text{Pu}})] = \text{sd}(\text{pitch}, m_{\text{Pu}})^2$$



Exercise #1: “design”

Surrogate function $K_{\text{eff}}(\text{pitch}, m_{\text{pu}})$

Random function

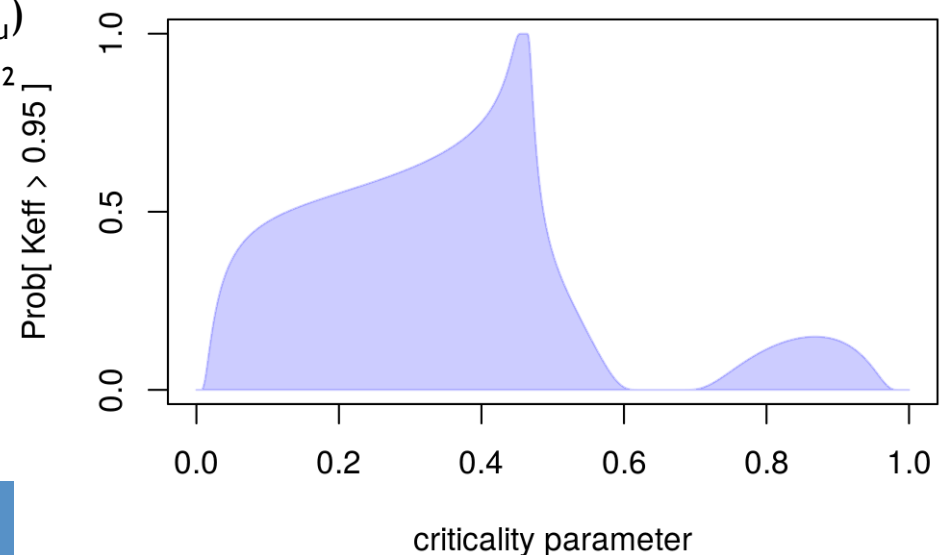
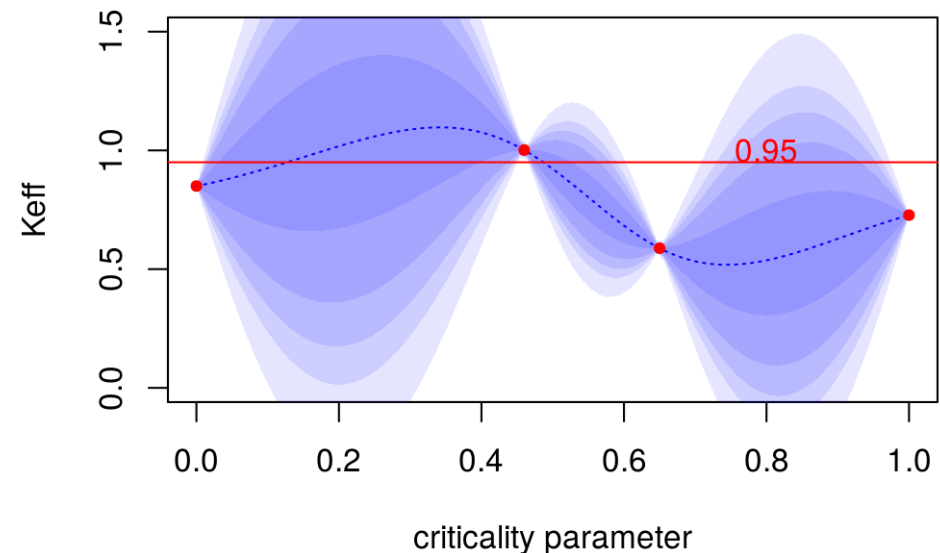
Interpolates measures
(even imprecise measures)

Gaussian predictor mean, sd:

$$E[K_{\text{eff}}(\text{pitch}, m_{\text{pu}})] = \text{mean}(\text{pitch}, m_{\text{pu}})$$

$$\text{Var}[K_{\text{eff}}(\text{pitch}, m_{\text{pu}})] = \text{sd}(\text{pitch}, m_{\text{pu}})^2$$

→ Convenient to estimate:
 $\text{Prob}[K_{\text{eff}}(\text{pitch}, m_{\text{pu}}) > 0.95]$



Exercise #1: “design”

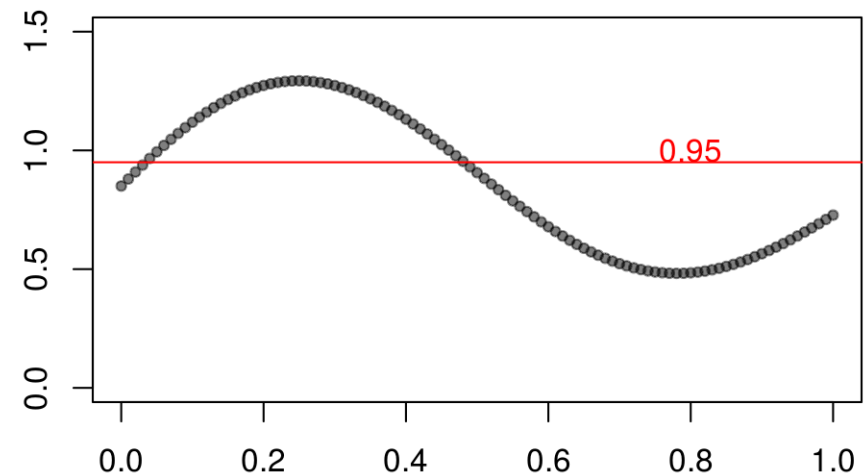
“valuable” point (pitch, m_{pu})

We aim to reach full certainty about: k_{eff}

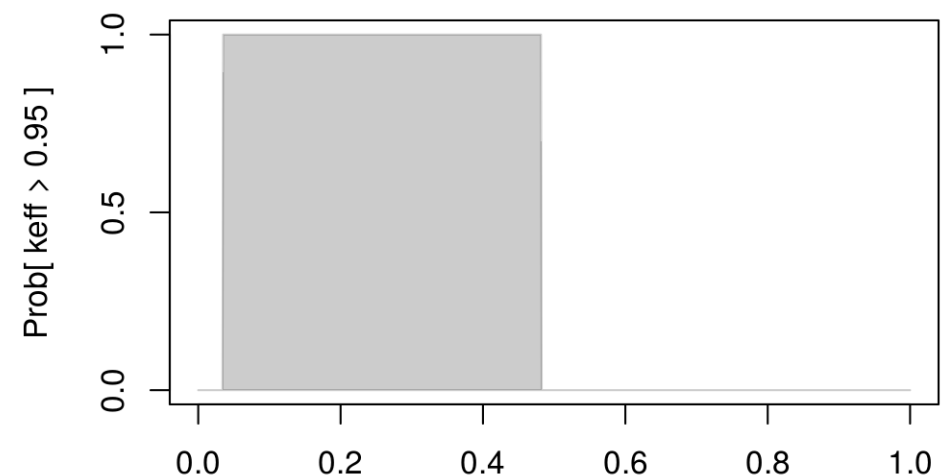
$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$



$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$



criticality parameter



criticality parameter

Exercise #1: “design”

“valuable” point (pitch, m_{pu})

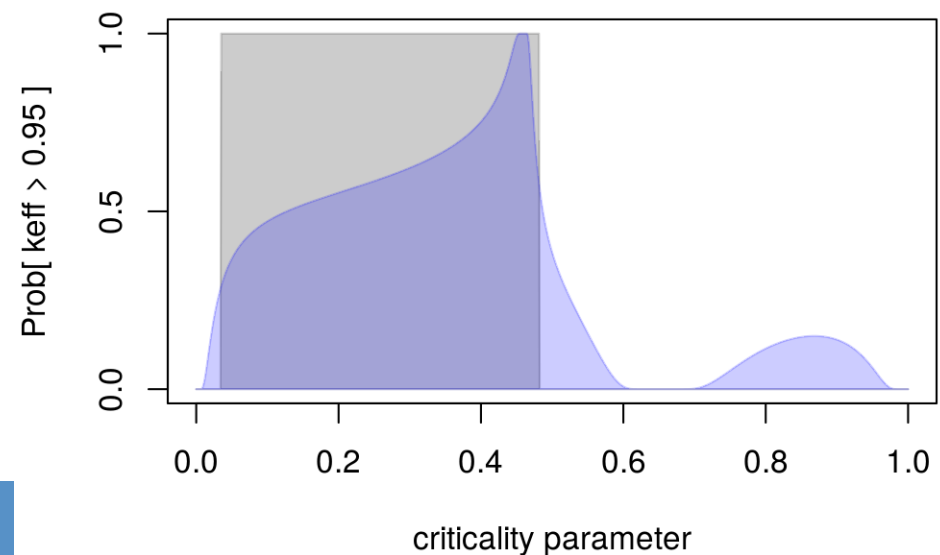
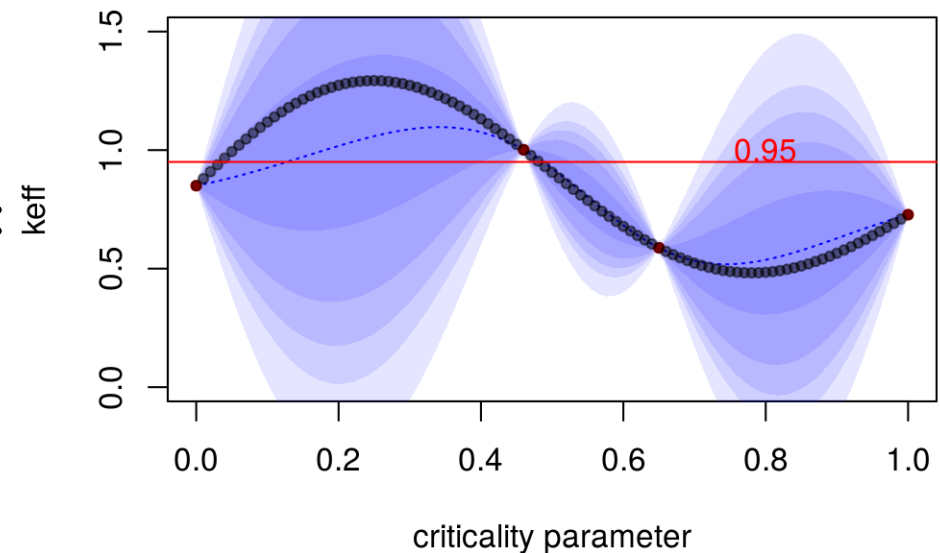
We aim to reach full certainty about: k_{eff}

$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$



$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?



Exercise #1: “design”

“valuable” point (pitch, m_{pu})

We aim to reach full certainty about: k_{eff}

$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$

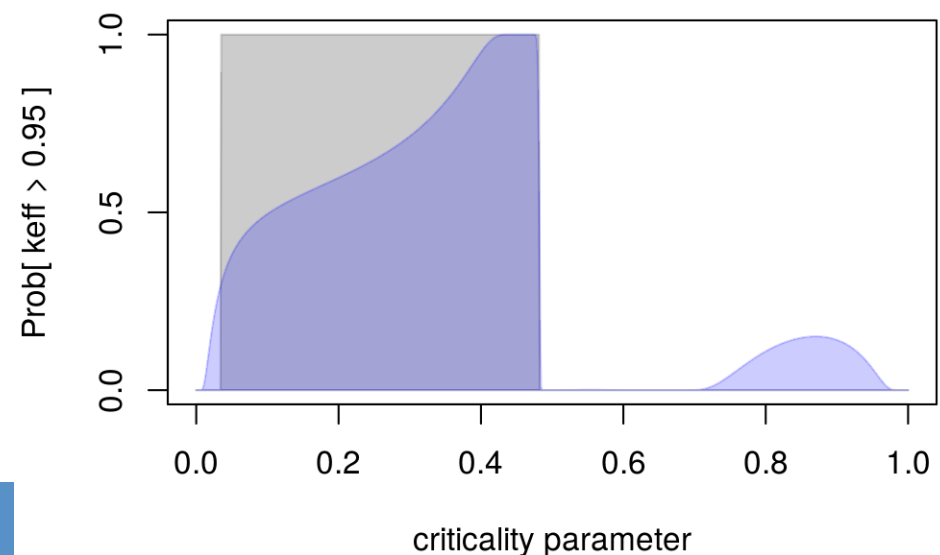
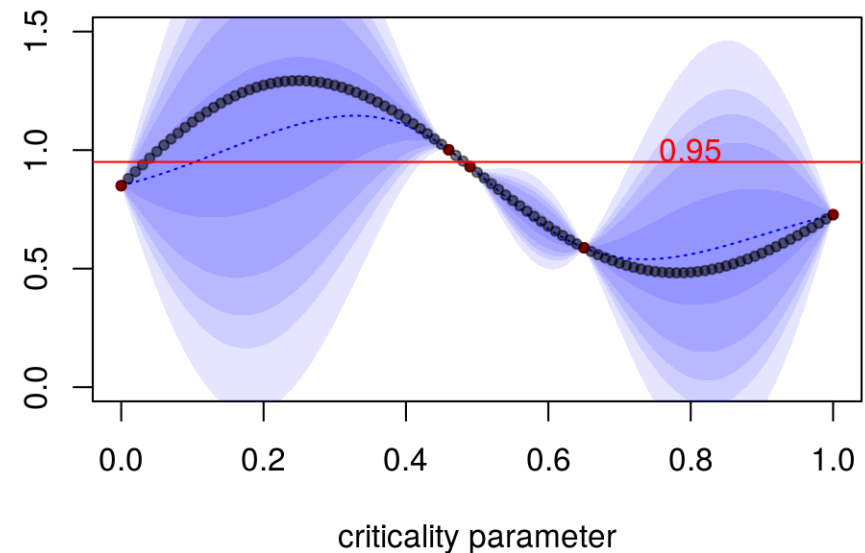


$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?

where, once added,

$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] \rightarrow \{0, 1\}$$



Exercise #1: “design”

“valuable” point (pitch, m_{pu})

We aim to reach full certainty about: k_{eff}

$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$

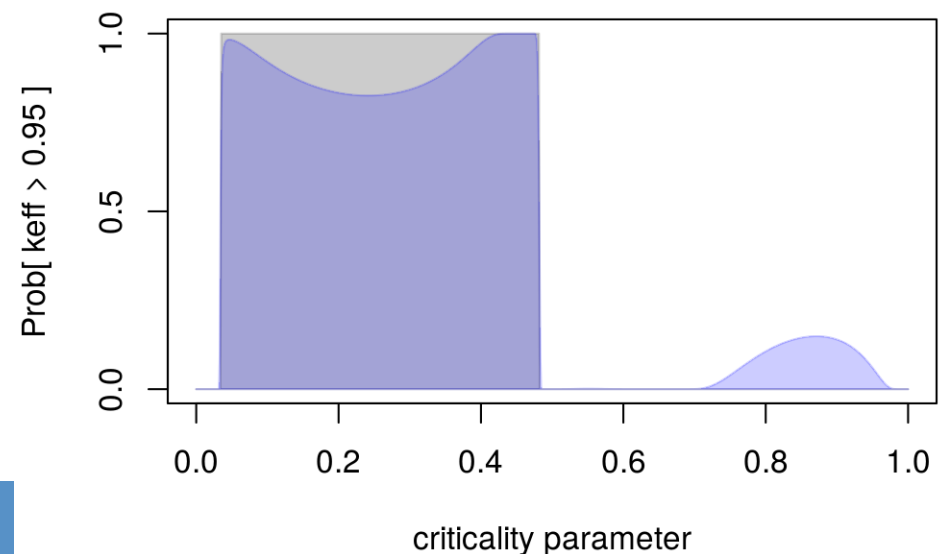
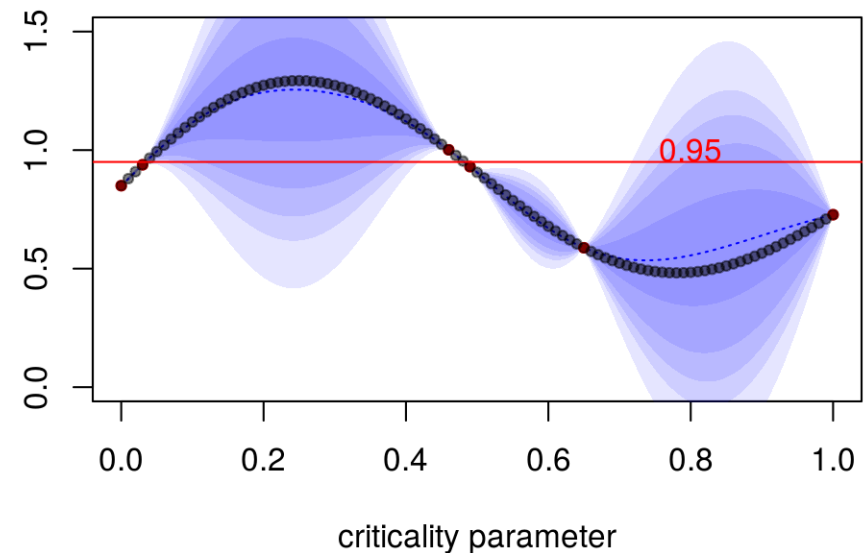


$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?

where, once added,

$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] \rightarrow \{0, 1\}$$



Exercise #1: “design”

“valuable” point (pitch, m_{pu})

We aim to reach full certainty about: k_{eff}

$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$

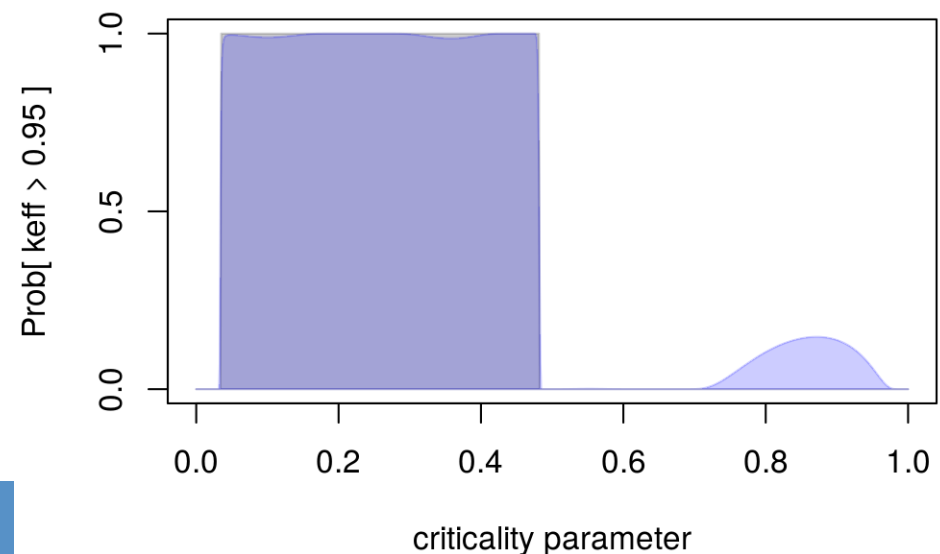
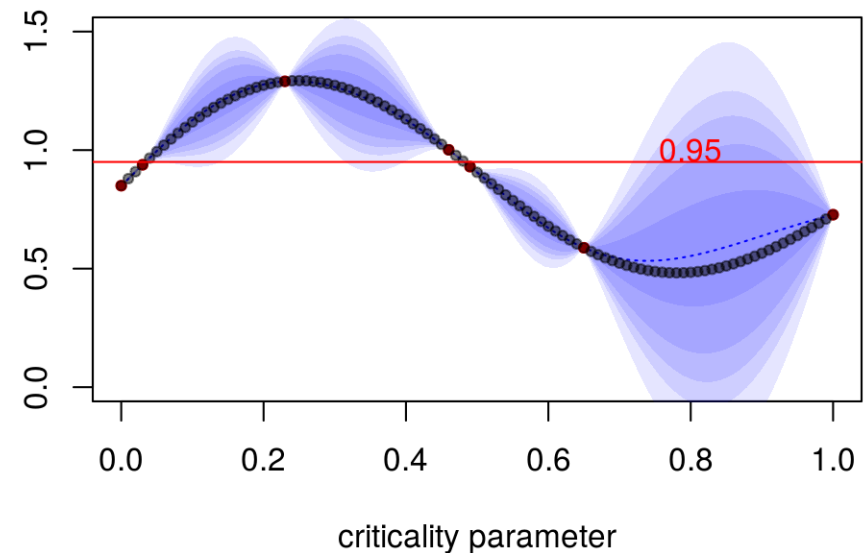


$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?

where, once added,

$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] \rightarrow \{0, 1\}$$



Exercise #1: “design”

“valuable” point (pitch, m_{pu})

We aim to reach full certainty about: k_{eff}

$$k_{eff}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$

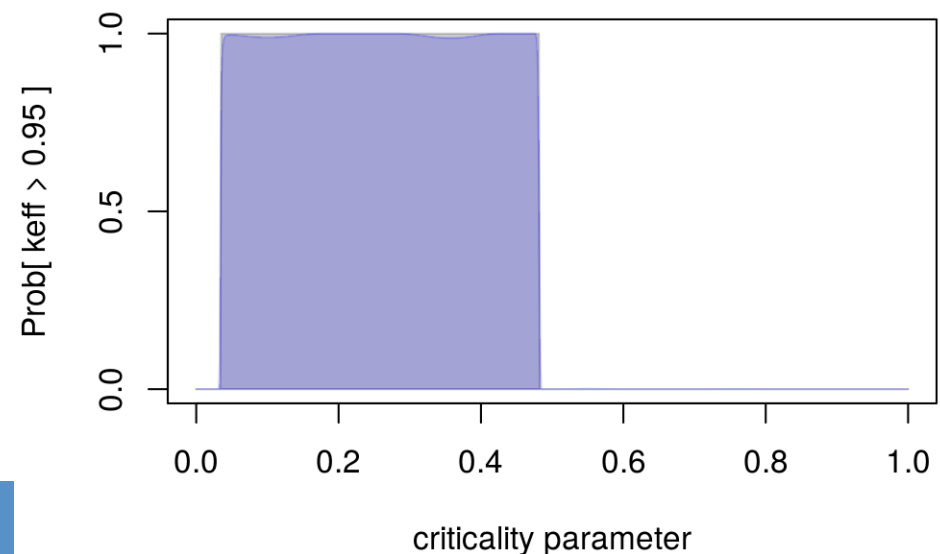
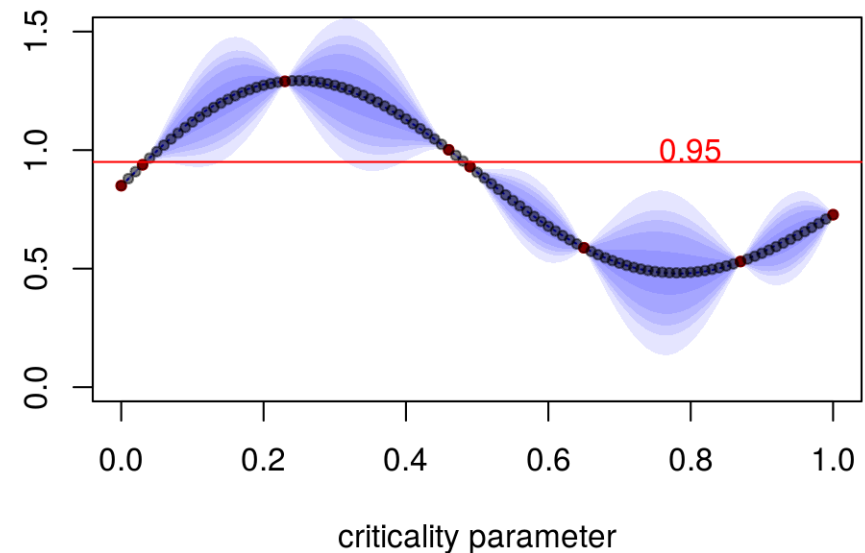


$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?

where, once added,

$$\text{Prob}[K_{eff}(\text{pitch}, m_{pu}) > 0.95] \rightarrow \{0, 1\}$$



Exercise #1: “design”

“valuable” point (pitch, m_{pu})

We aim to reach full certainty about:

$$k_{\text{eff}}(\text{pitch}, m_{pu}) > \text{or} < 0.95$$



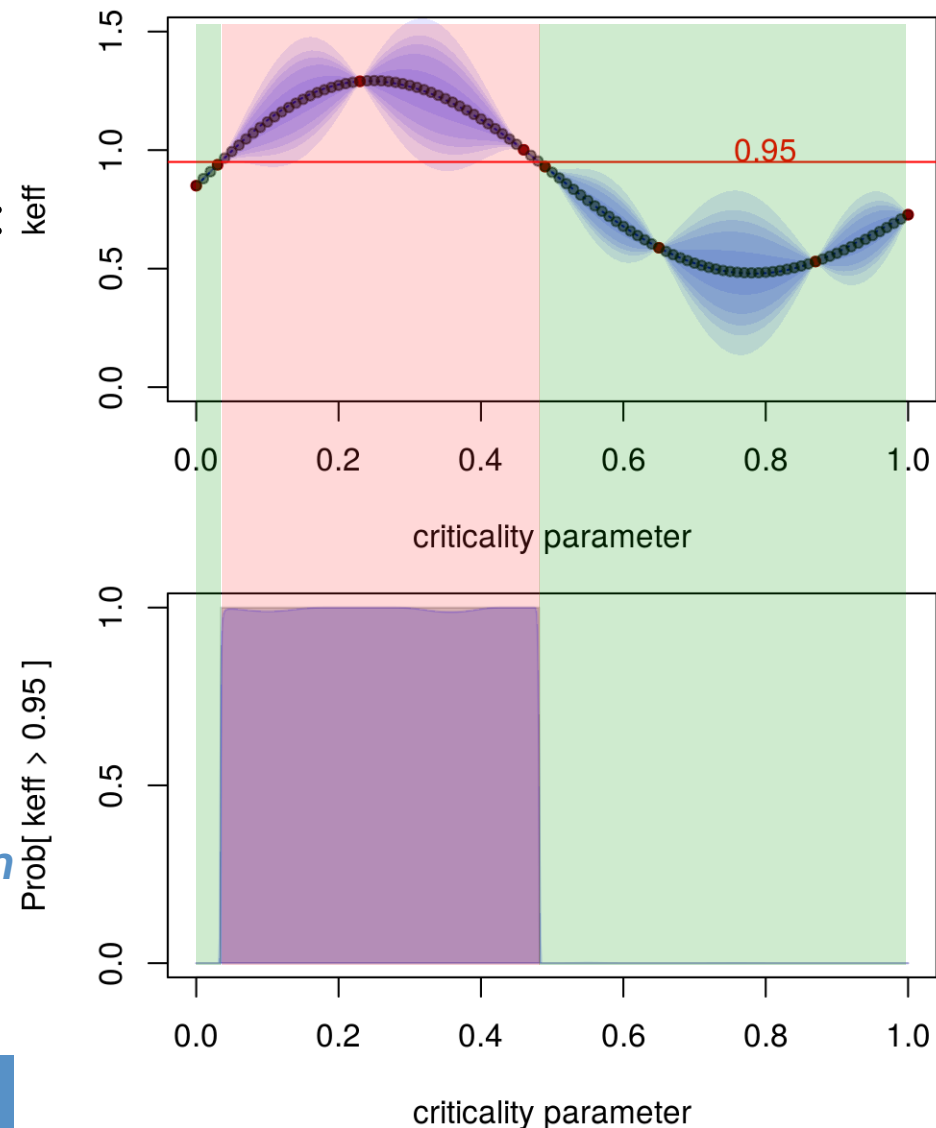
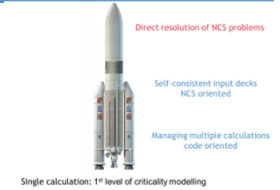
$$\text{Prob}[K_{\text{eff}}(\text{pitch}, m_{pu}) > 0.95] = 0 \text{ or } 1$$

Where to add next points/calculations?

where, once added,

$$\text{Prob}[K_{\text{eff}}(\text{pitch}, m_{pu}) > 0.95] \rightarrow \{0, 1\}$$

In the end, the safety area is precisely known thanks to the *surrogate function* (even for more than 1-dimensional area)



“2nd level of criticality modelling”

Use of statistical learning algorithm

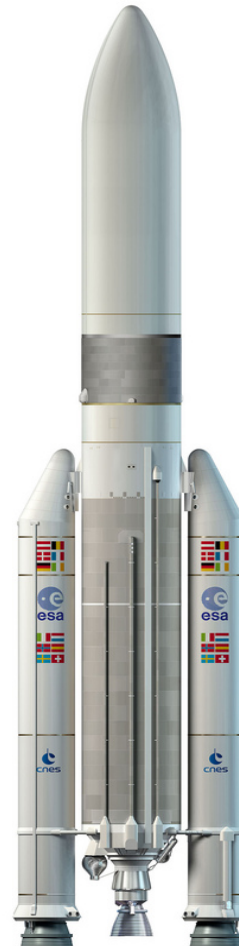
Direct resolution of NCS problems

Advanced parametrization

Self-consistent input decks
NCS oriented

Basic parametrization

Managing multiple calculations
code oriented



Single calculation: 1st level of criticality modelling

Exercise #2: “penalization”



■ What about “nuisance parameters”?

- Among all possible “nuisance parameters” in this problem, let’s consider **interaction**:

- **Is interstitial moderation increase interaction?**

→ Modelling a water layer around the tubes with a variable thickness

Thickness can take any value: **nuisance parameter**

- **Is decreasing PuO_2 density increase interaction?**

→ Change the PuO_2 density

Density can’t take any value, because **controlled parameter**: **[0,4]**

Exercise #2: “penalization”

Exercise 2.1: Modelling a variable water layer around the tubes

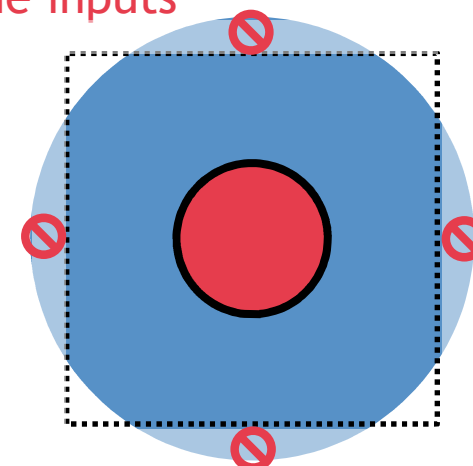
Water layer already included in your decks, just make its thickness varying

COG	MORET
<i>In SURFACE #16, replace 6.8 by $@\{6.8 + \%water_thick_cm \mid 0.0000\}$</i>	<i>In TYPE #3 replace 6.8 by $@\{6.8 + \\$water_thick_cm \mid 0.0000\}$</i>

- Reminder: Beware of parameters dependences on all code inputs

What about a large thickness and a small pitch?

- ➔ Limit the water thickness variation range
or Account for a possible intersection in the deck

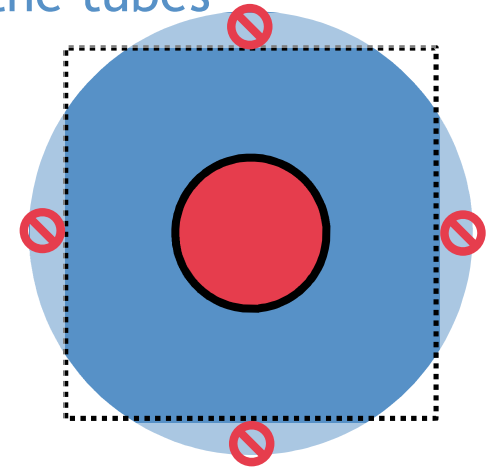


Exercise #2: “penalization”



Exercise 2.1: Modelling a variable water layer around the tubes

- Account for a possible intersection in the deck:
Define the “water” VOLUME as being truncated
by the “storage unit” VOLUME



COG	Done in the SECTOR #3 definition (SURFACES #1,2,3,4 are boundaries of SECTOR #3)
MORET	in VOLUME #3, replace the comment mark * by <i>@{ if(6.8+\$water_thick_cm > \$pitch_cm/2) print (" ") else print("*") }</i>

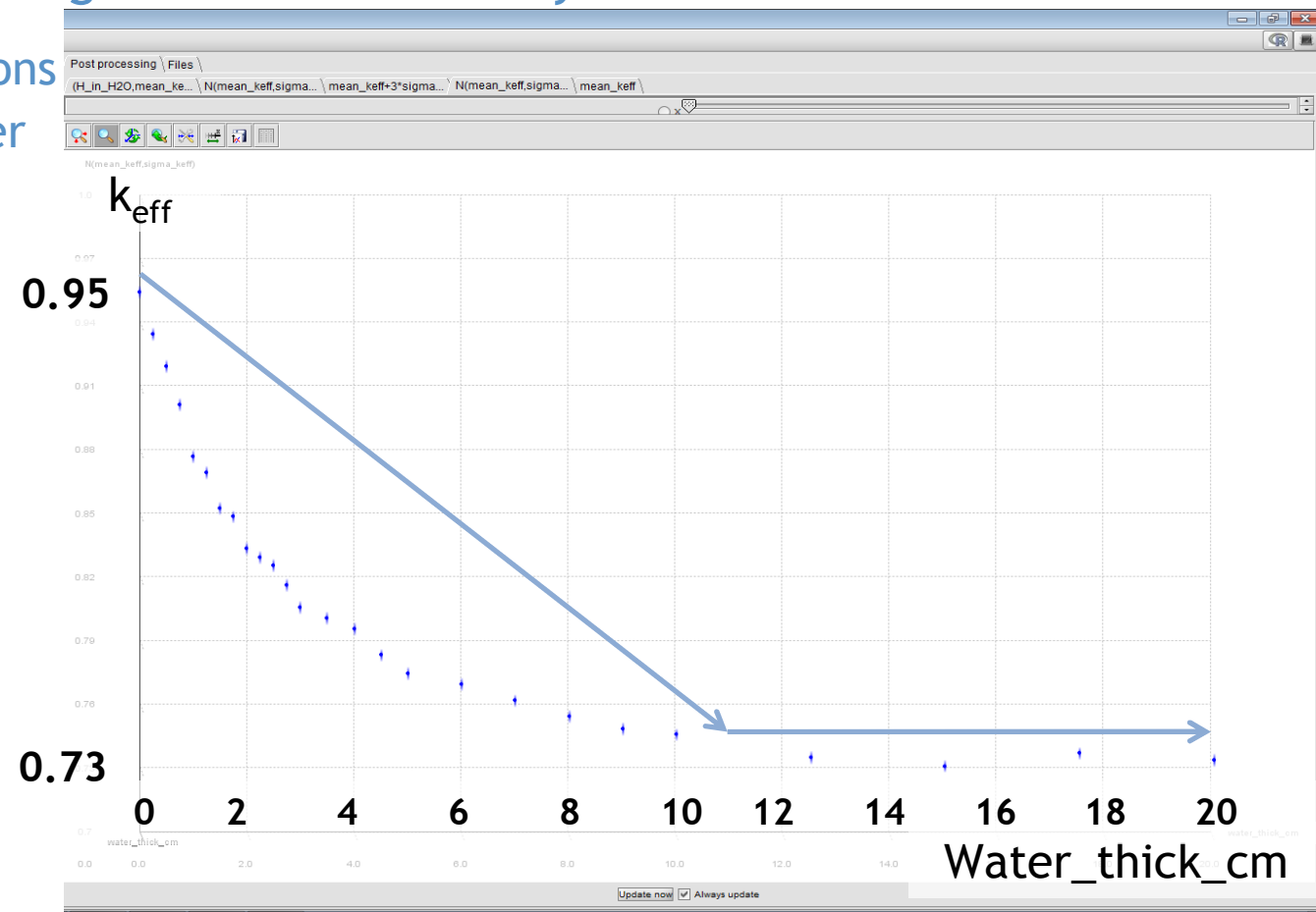
Exercise #2: “penalization”



Exercise 2.1: Modelling a variable water layer around the tubes

- Results: k_{eff} variations with the water layer thickness

For $m_{\text{pu}} = 36 \text{ kg}$
pitch = 90 cm





Exercise #2: “penalization”

Exercise 2.2: Changing the PuO₂ density

- Change the Material Definition
 - PuO₂ density d_{PuO_2} is not a direct input of the code...
... because PuO₂ contains water (direct input: total material density d_{powder})

$$d_{\text{powder}} = \frac{d_{\text{PuO}_2}}{\left(1 - \% \frac{\text{water}}{\text{powder}}\right)}$$

Reminder:
Powder = Wet PuO₂

- Oxide density $\leq 4.0 \text{ g/cm}^3$
- Water content $\leq 6.0 \text{ wt. \%}$

Exercise #2: “penalization”



Exercise 2.2: Changing the PuO₂ density

- Change the Material Definition

COG	in MATERIAL #5 replace 4.2553 by $@\{\%PuO2_dens_gcm3 / (1-6/100) \mid 0.0000\}$
MORET	in MATERIAL “FISSILE” replace 4.2553 by $@\{\$PuO2_dens_gcm3 / (1-6/100) \mid 0.0000\}$

- Is it sufficient?
Is there any thing else impacted by setting the density as a variable parameter?

H_{fiss_cm} formula now depends on both mass and density

Exercise #2: “penalization”



Exercise 2.2: Changing the PuO₂ density

- Change the H_fiss_cm formula (because depends on both mass and density)

Modify the H_fiss_cm formula as follow:

COG	<pre>\$@: H_fiss_cm <- function(Pu_mass_kg, PuO2_dens_gcm3) { 1000 * Pu_mass_kg / (pi/4 * 11.5^2 * PuO2_dens_gcm3 * Pu_in_PuO2) }</pre>
MORET	<pre>\$@: H_fiss_cm <- function(Pu_mass_kg, PuO2_dens_gcm3) { 1000 * Pu_mass_kg / (pi/4 * 11.5^2 * PuO2_dens_gcm3 * Pu_in_PuO2) }</pre>

→ Change also all the calls to the formula

COG	<pre>@{... H_fiss_cm(%Pu_mass_kg,%PuO2_dens_gcm3) ... }</pre>
MORET	<pre>@{... H_fiss_cm(\$Pu_mass_kg,\$PuO2_dens_gcm3) ... }</pre>

Exercise #2: “penalization”

Exercise 2.2: Changing the PuO₂ density

- Anything else? What about dependences on other code inputs?

Low density → $H_{\text{fiss_cm}} > H_{\text{cans}}$

→ Limit the filling height variation range **Truncate $H_{\text{fiss_cm}}$ to H_{cans} (=296 cm)**
or Account for a possible intersection in the deck

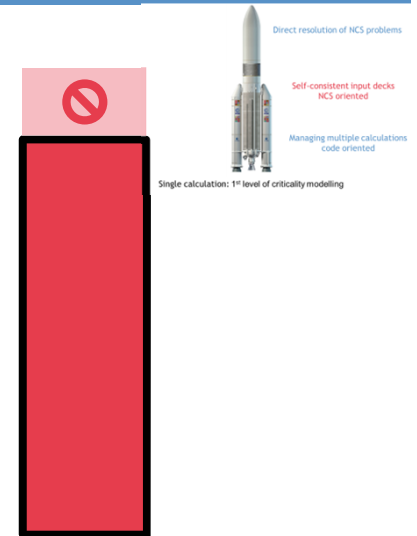
Modify the $H_{\text{fiss_cm}}$ formula as follow:

COG

```
$@: H_fiss_cm <- function(Pu_mass_kg, PuO2_dens_gcm3)
{ min(296,
      1000 * Pu_mass_kg / (pi/4 * 11.5^2 * PuO2_dens_gcm3 * 0.88211)) }
```

MORET

```
*@: H_fiss_cm <- function(Pu_mass_kg, PuO2_dens_gcm3)
{ min(296,
      1000 * Pu_mass_kg / (pi/4 * 11.5^2 * PuO2_dens_gcm3 * 0.88211)) }
```



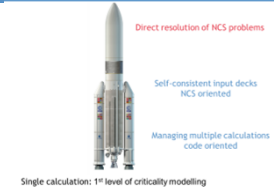
Exercise #2: “penalization”



- **Challenge:** We found that $k_{\text{eff}}(36 \text{ kg ; } 90 \text{ cm}) \sim 0.95$,
is it still true considering interaction as a nuisance?
- **Exercise 2.3:** Resort to your own Design of Experiments to find $\max(k_{\text{eff}})$
for any water layer thickness and PuO_2 density $\leq 4.0 \text{ g/cm}^3$

$\max(k_{\text{eff}}) \sim$ for $\text{PuO}_2\text{_dens_gcm3} =$
 $\text{water_thick_cm} =$

Exercise #2: “penalization”



■ **Challenge:** We found that $k_{\text{eff}}(36 \text{ kg} ; 90 \text{ cm}) \sim 0.95$,
is it still true considering interaction as a nuisance?

- **Exercise 2.4:** Resort to an algorithm
 - Multiple strategies possible (gradient descent, genetic algorithms,...)
 - Example of one algorithm we have found well adapted for NCS problems:
EGO
 - Set “pitch_cm = 90” and “Pu_mass_kg = 36”
 - Select “engineering” for both “PuO2_dens_gcm3” and “water_thick_cm”
 - Set the lower and upper bounds (e.g. [0.2,4.0] and [0.,5.])
 - Select the “optimization / EGO” algorithm
 - Run the project and Look at the results...

Exercise #2: “penalization”



■ **Challenge:** search the most penalizing k_{eff} using $\{d_{PuO_2}, \text{water}\}$

Efficient Global Optimization (EGO) algorithm (optimization algorithm)

1. calculate k_{eff} for first few points $(d_{PuO_2}, \text{water})$, randomly chosen plus bounds
 2. generate a **surrogate function** K_{eff} :
interpolating previous $\{k_{eff}(d_{PuO_2}, \text{water})\}$ calculations
 3. search the next **most “valuable” points** $(d_{PuO_2}, \text{water})$
 4. perform these k_{eff} calculations, stack with previous $\{k_{eff}(d_{PuO_2}, \text{water})\}$
- repeat steps 2-3-4 as needed to find highest k_{eff}

- **Key details:**

- **surrogate function:** allows to estimate $E[K_{eff}(d_{PuO_2}, \text{water}) > \max\{k_{eff}\}]$
- **“valuable” point:** helps to predict where $k_{eff} > \max\{k_{eff}\}$

Exercise #2: “penalization”



Surrogate function $K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})$

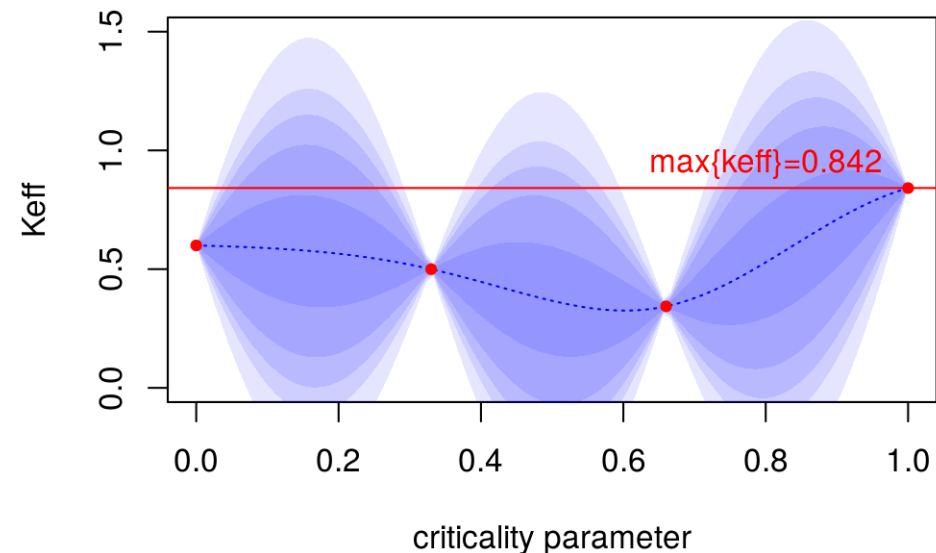
Random function

Interpolates measures
(even imprecise measures)

Gaussian predictor mean,sd:

$$E[K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})] = \text{mean}(d_{\text{PuO}_2}, \text{water})$$

$$\text{Var}[K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})] = \text{sd}(d_{\text{PuO}_2}, \text{water})^2$$



Exercise #2: “penalization”



Surrogate function $K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})$

Random function

Interpolates measures
(even imprecise measures)

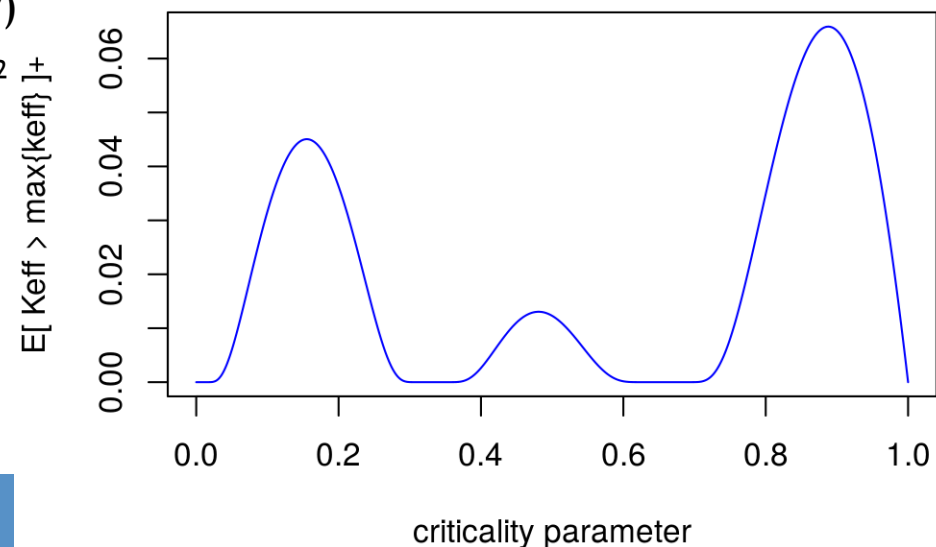
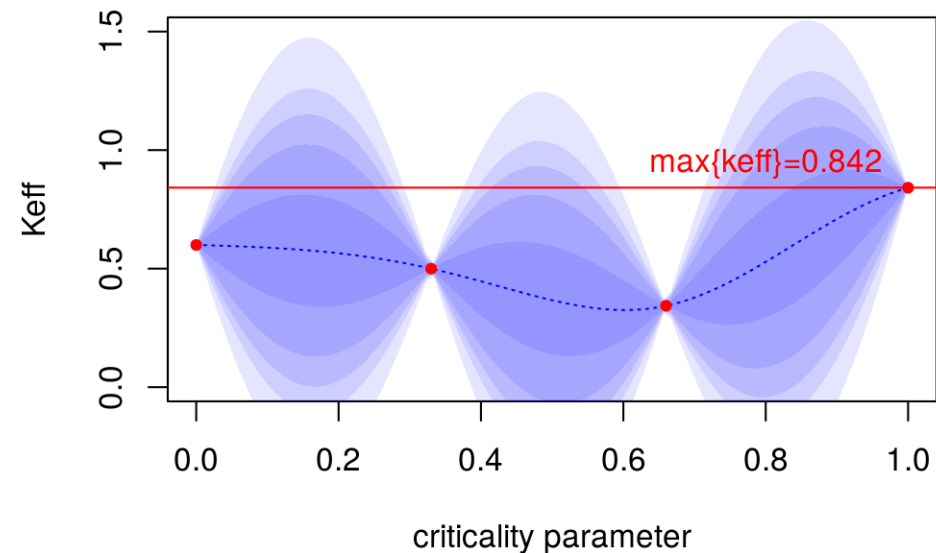
Gaussian predictor mean,sd:

$$E[K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})] = \text{mean}(d_{\text{PuO}_2}, \text{water})$$

$$\text{Var}[K_{\text{eff}}(d_{\text{PuO}_2}, \text{water})] = \text{sd}(d_{\text{PuO}_2}, \text{water})^2$$

→ Convenient to estimate:

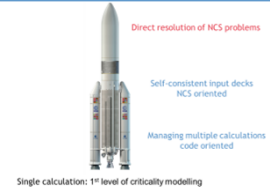
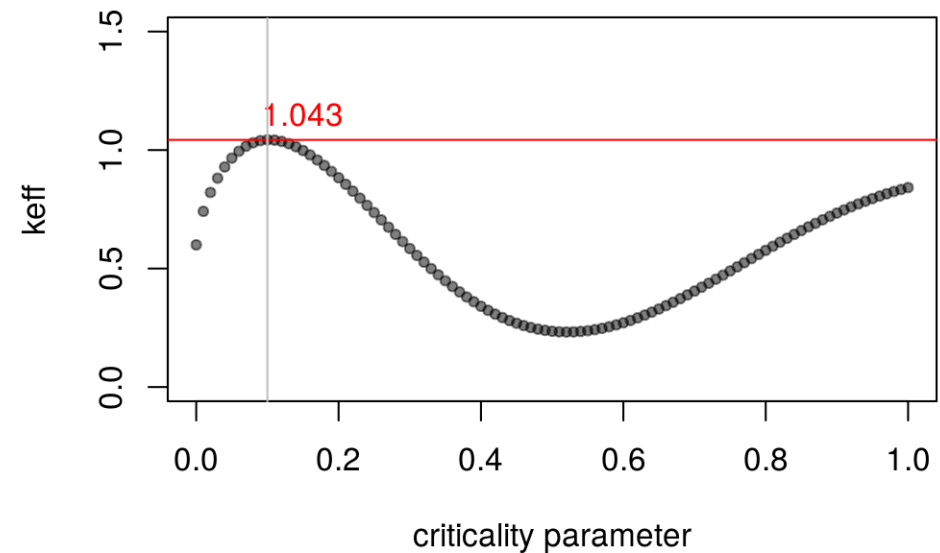
$$E[K_{\text{eff}}(d_{\text{PuO}_2}, \text{water}) > \max\{\text{keff}\}]^+$$



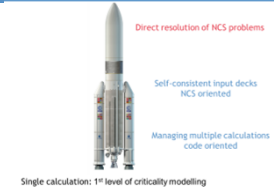
Exercise #2: “penalization”

“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}



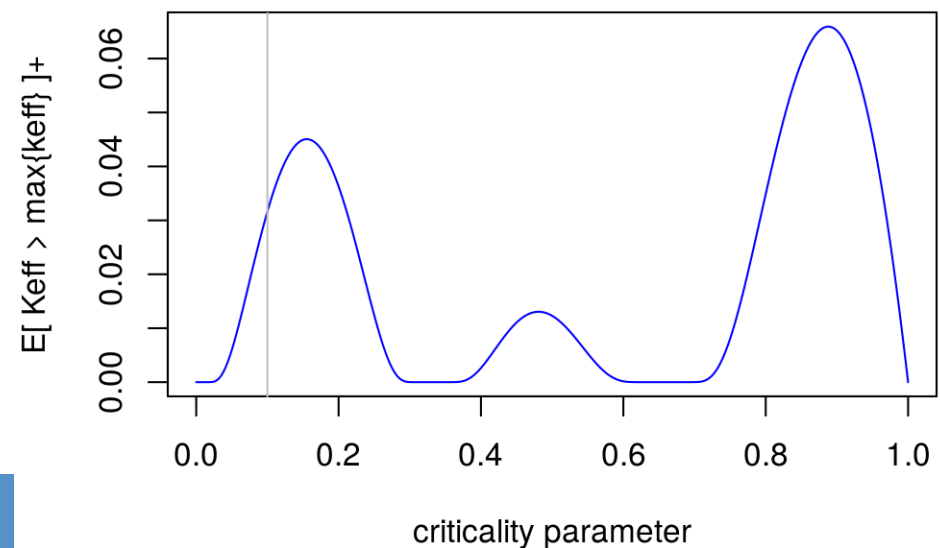
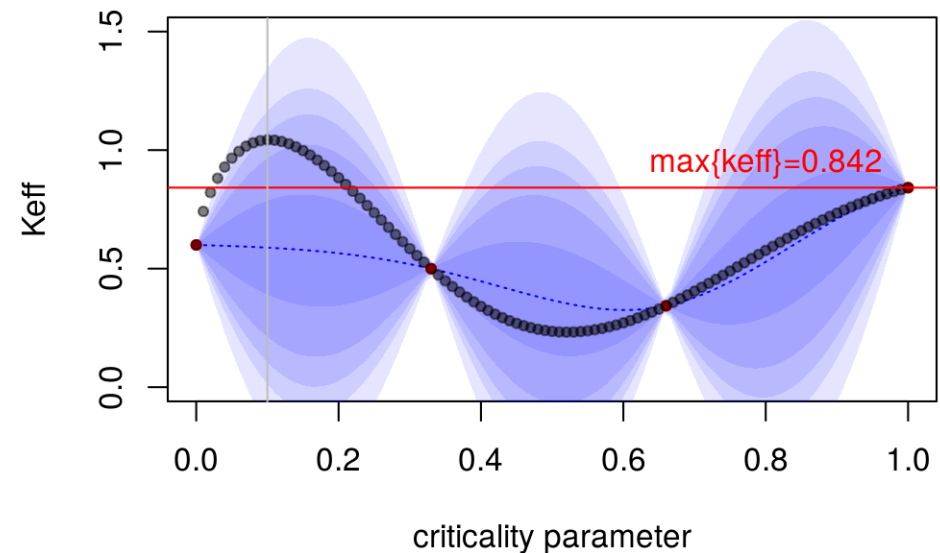
Exercise #2: “penalization”



“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}

Where to add next points/calculations?



Exercise #2: “penalization”

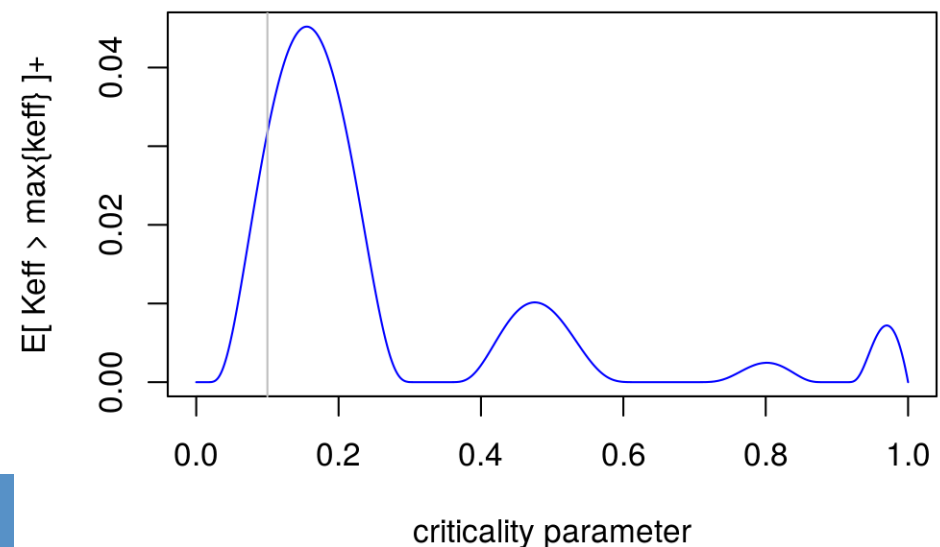
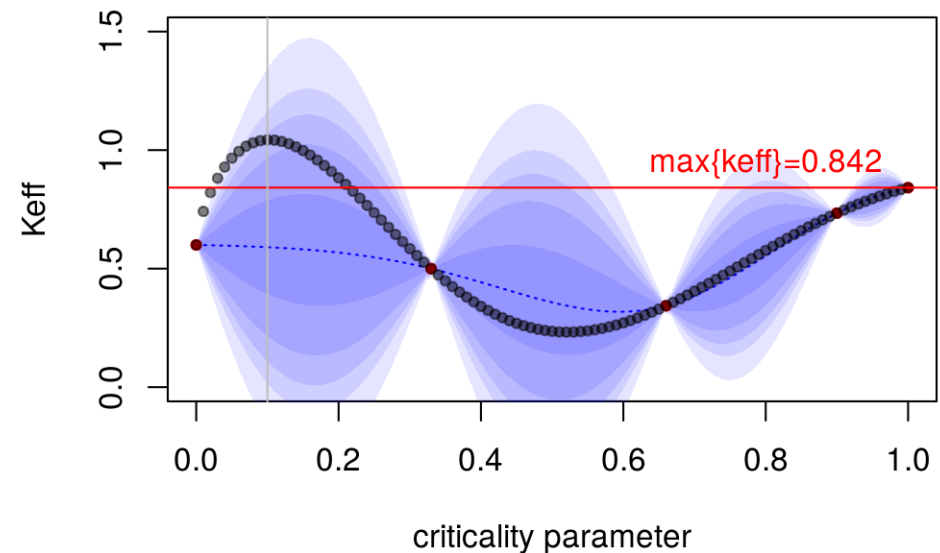


“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}

Where to add next points/calculations?

where is the highest $E [K_{\text{eff}} > \max\{k_{\text{eff}}\}]^+$



Exercise #2: “penalization”

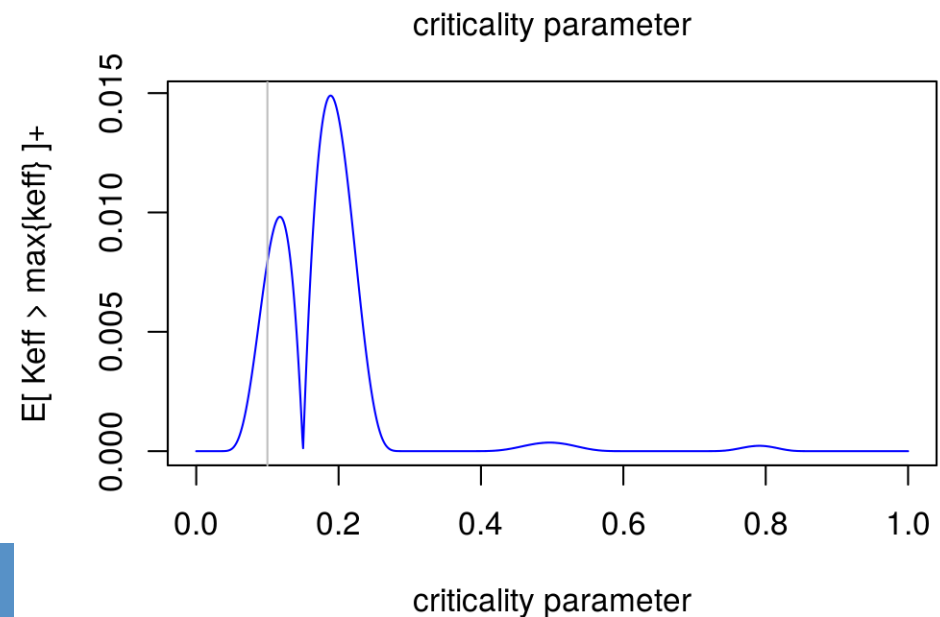
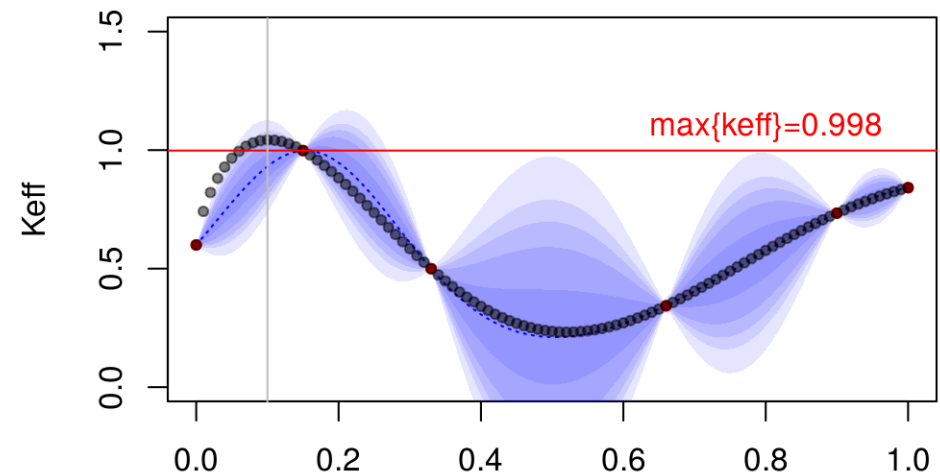


“valuable” point (d_{PuO_2} , water)

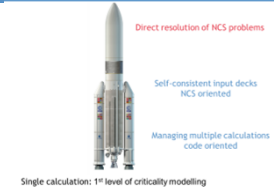
We aim to reach highest k_{eff}

Where to add next points/calculations?

where is the highest $E [K_{\text{eff}} > \max\{k_{\text{eff}} \}]^+$



Exercise #2: “penalization”

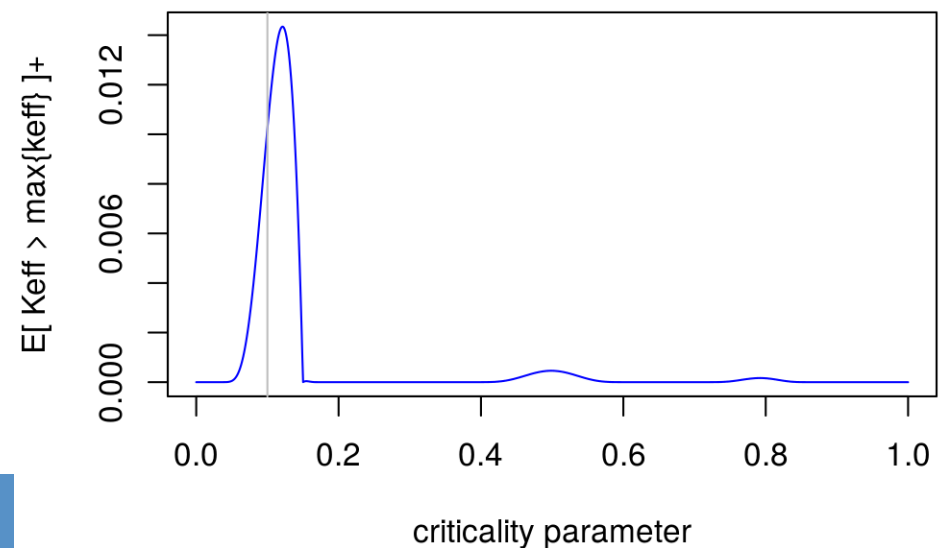
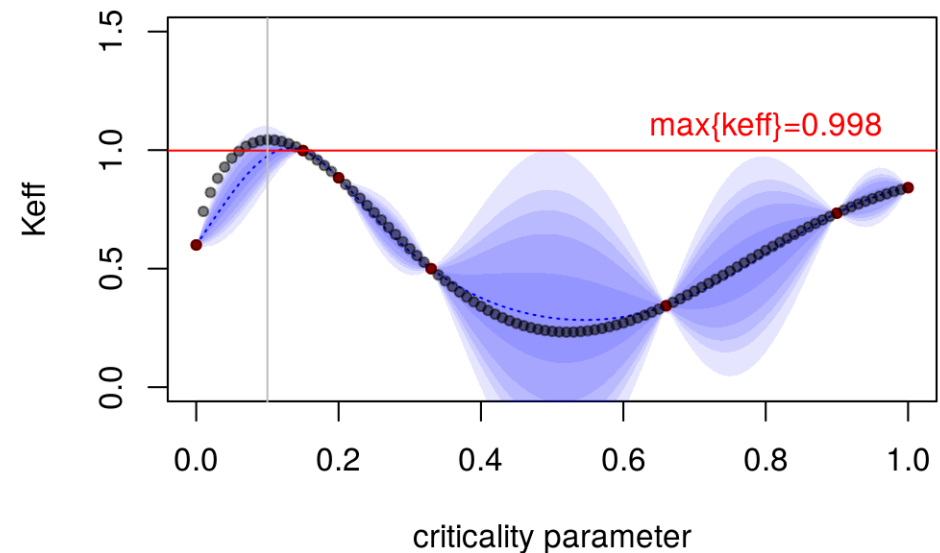


“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}

Where to add next points/calculations?

where is the highest $E [K_{\text{eff}} > \max\{k_{\text{eff}} \}]^+$



Exercise #2: “penalization”

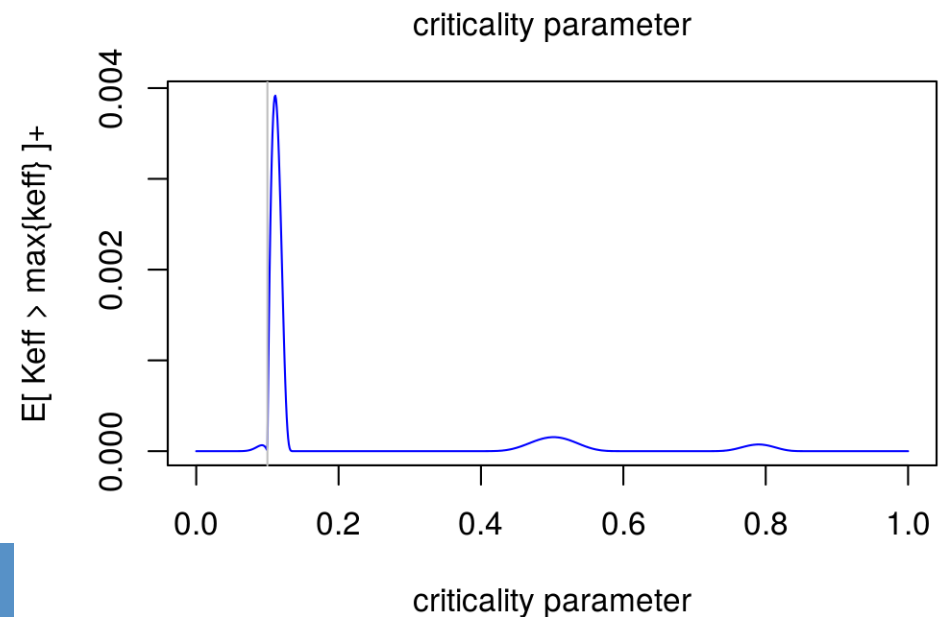
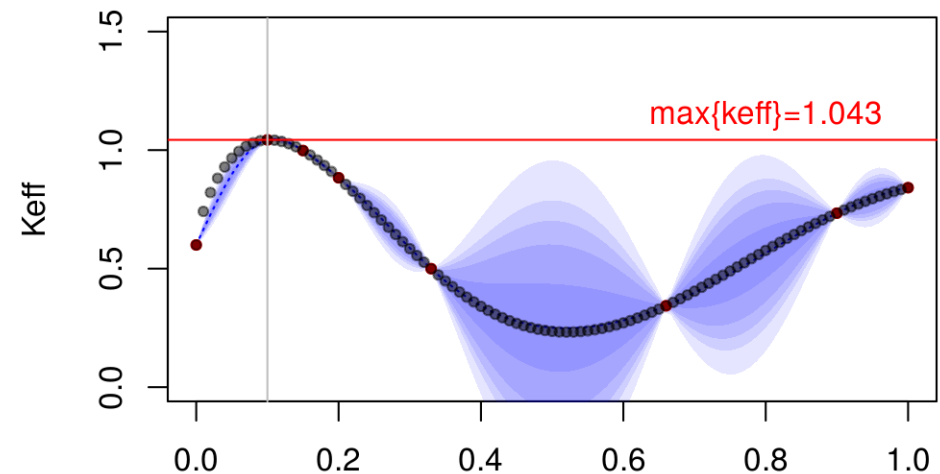


“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}

Where to add next points/calculations?

where is the highest $E [K_{\text{eff}} > \max\{k_{\text{eff}} \}]^+$



Exercise #2: “penalization”



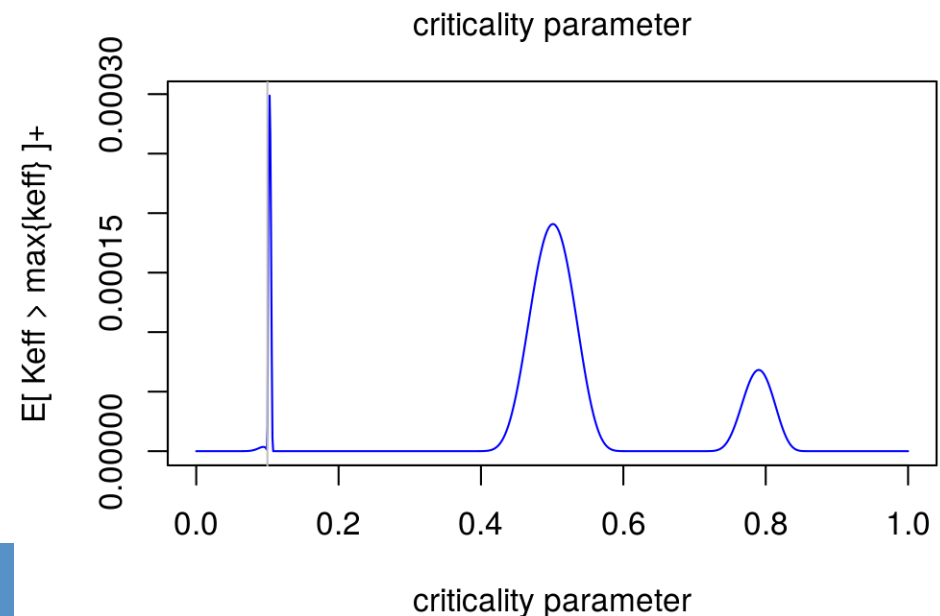
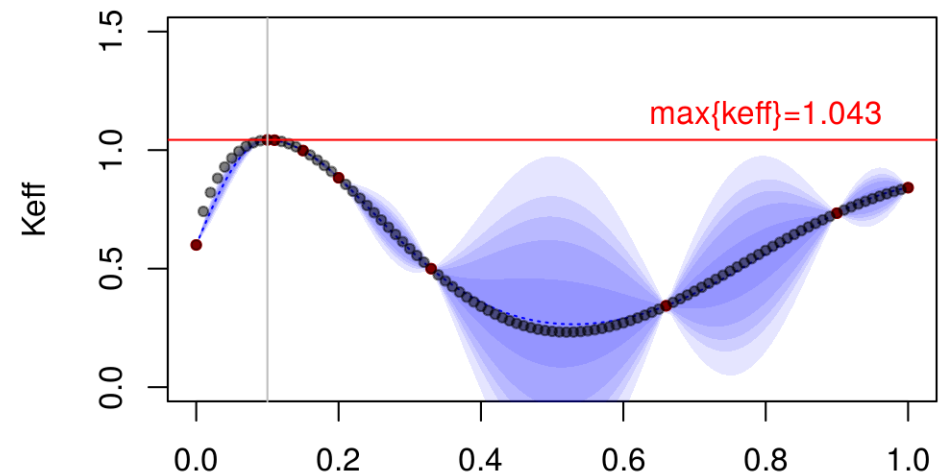
“valuable” point (d_{PuO_2} , water)

We aim to reach highest k_{eff}

Where to add next points/calculations?

where is the highest $E [K_{\text{eff}} > \max\{k_{\text{eff}} \}]^+$

**In the end,
we reached the highest k_{eff}**



Exercise #3: “generalization”



- **Exercise 3:** Since interaction conditions strongly affects the results, what are the true safe conditions for the storage?

What are the $\{\text{pitch}, m_{\text{Pu}}\}$ where $\max(k_{\text{eff}}) < 0.95$ for any $\{d_{\text{PuO}_2}, \text{water}\}$?

- Resort to your own Design of Experiments or to an Algorithm?
- Which kind of algorithm may solve such problems? SUR? EGO?

none of them, nor a combination of the two



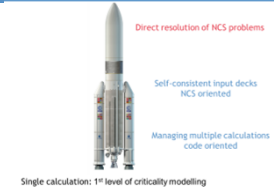
Exercise #3: “generalization”

■ What are the $\{\text{pitch}, m_{\text{Pu}}\}$ where $\max(k_{\text{eff}}) < 0.95$ for any $\{d_{\text{PuO}_2}, \text{water}\}$?

Robust Stepwise Uncertainty Reduction (RSUR) algorithm

1. calculate k_{eff} for first few points ($\text{pitch}, m_{\text{Pu}}, d_{\text{PuO}_2}, \text{water}$), random + bounds
 2. generate a surrogate function K_{eff} :
interpolating previous $\{k_{\text{eff}}(\text{pitch}, m_{\text{Pu}}, d_{\text{PuO}_2}, \text{water})\}$ calculations
 3. search the next most “valuable” points ($\text{pitch}, m_{\text{Pu}}, d_{\text{PuO}_2}, \text{water}$)
 4. perform these k_{eff} calculations, stack with previous $\{k_{\text{eff}}\}$
- repeat 2-3-4 as needed to get a reliable definition of the safe area $\{\text{pitch}, m_{\text{Pu}}\}$
- Key details:
 - surrogate function: estimate $\text{Prob}[K_{\text{eff}}(\text{pitch}, m_{\text{Pu}}, [d_{\text{PuO}_2}], [\text{water}]) > 0.95]$
 - “valuable” point: helps to predict where $k_{\text{eff}}([d_{\text{PuO}_2}], [\text{water}]) > 0.95$

Exercise #3: “generalization”



■ RSUR results

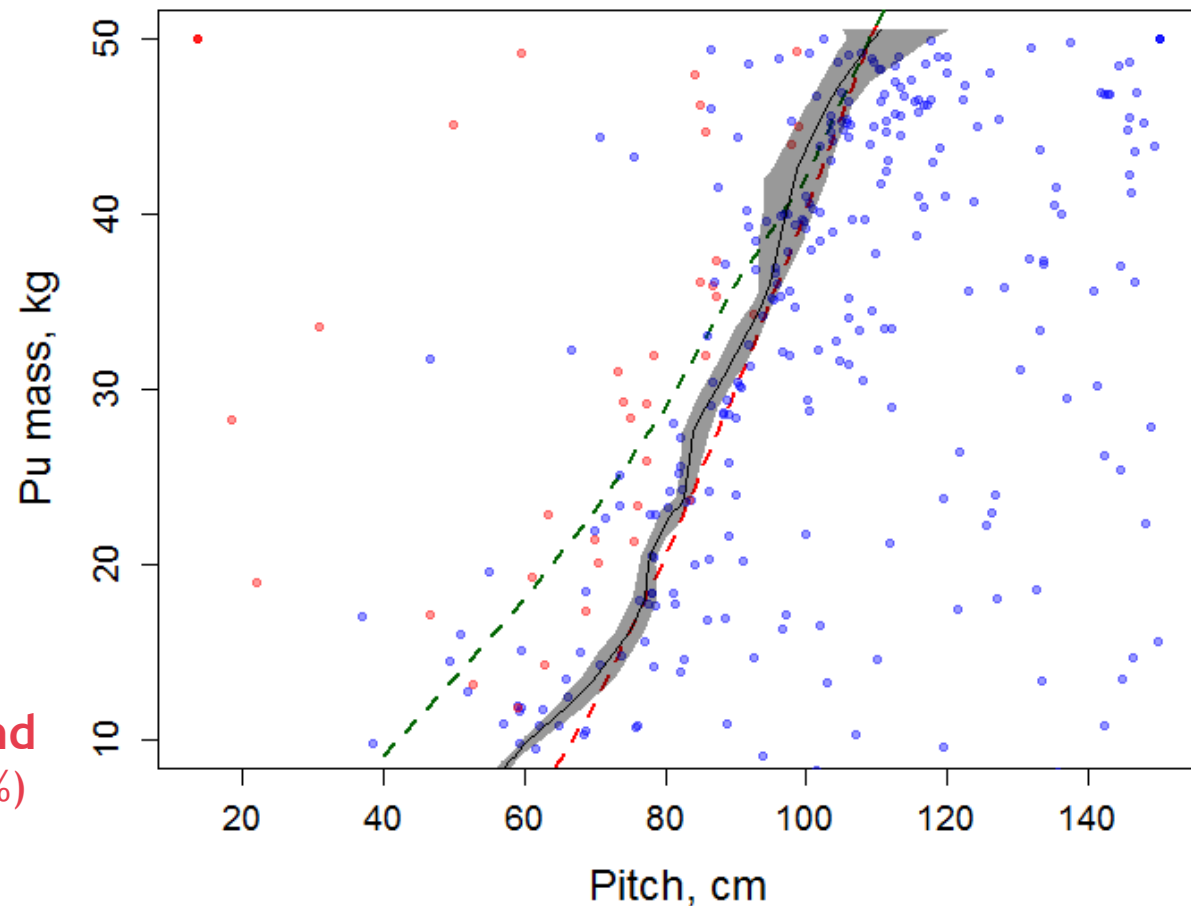
Target k_{eff} : 0.95

green: non-penalized
boundary (Exercise #1)

red: penalized boundary
(‘brute force’ results)

After 300 calculations,
penalized boundary ~ found
(uncertainty due to $\sigma_{\text{calc.}} = 0.3\%$)

Batch#60 - 36 initial calculations + 300 new calculations (RSUR)



Conclusions

■ “2nd level of criticality modelling”

- **Stage 1: Basic Parametrization** to manage multiple calculations
 - **Stage 2: Advanced Parametrization** to have self-consistent input decks oriented from a NCS point of view (rather than a code-centric view)
 - Requires more time to build the decks
 - Requires to think about dependence of varying parameters
 - **Stage 3: Use of statistical learning algorithm** to solve directly NCS problems
 - Should not replace experts but may assist them (*a priori* and *a posteriori*)
 - Requires to interpret and challenge algorithm results (as well as k_{eff} results today: validation, convergence,...)
- ➔ **Algorithms can become your best friends, but NEVER take them on trust**

Use of statistical learning algorithm

Advanced parametrization

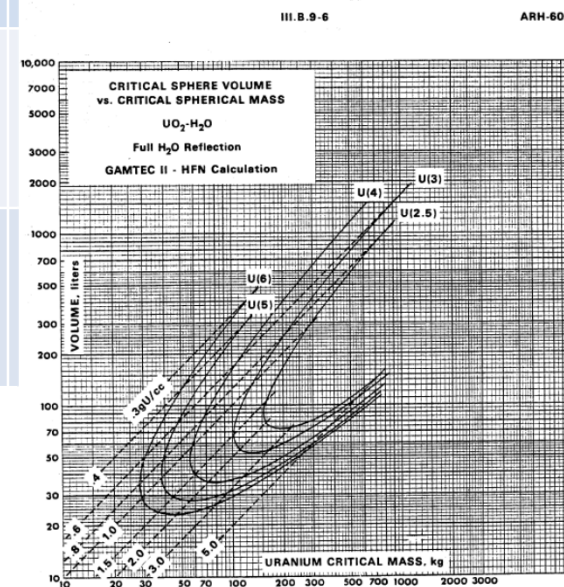
Basic parametrization



Conclusions

■ “2nd level of criticality modelling”

Type of parameters in the NCS problem	Class of Problem (Algorithm)
Controlled parameters (e.g. Multi-Parameter sub-critical limits, Design of a process,...)	Inversion (SUR)



Conclusions

I “2nd level of criticality modelling”

Type of parameters in the NCS problem	Class of Problem (Algorithm)
Controlled parameters (e.g. Multi-Parameter sub-critical limits, Design of a process,...)	Inversion (SUR)
Nuisance parameters (e.g. peer-review of an evaluation,...)	Optimization (EGO)
Controlled & Nuisance parameters (general case: sub-critical limits, design,...)	Robust Inversion (RSUR)

→ (Almost) all NCS problems can be covered by one of these
3 classes of problems

More details?

■ <http://promethee.irsn.org/>

The screenshot shows the Promethee website interface. The header includes the IRSN logo (Institut de Radioprotection et de Sûreté Nucléaire) and the tagline "Enhancing safety." A navigation menu on the left lists: Home, User documentation, Admin documentation, Development resources, Download, and Private area. A search bar is at the bottom of the menu. A central banner features a 3D sphere graphic and a text box stating: "The final version of Promethee 1.3 is available." followed by a list of improvements: improved "Rat Race" Queueing system, add user control to refresh results (plots, tables), add minimum resources filter on calculator (CPU, mem, disk), and add output file filter to reduce disk space needs. Below this, a detailed flowchart illustrates the system architecture. It starts with "Project start" leading to "Input file(s)", then "Parameterize", and "Input model". These lead to "Parse & eval model variables" and "Instantiate cases", which then feed into the "Math engine". The "Math engine" outputs to "Parse & eval output values", which leads to "Algorithm iteration" and "Analyse & display results". This results in a 3D plot. The process then moves to "Iterate project hypothesis" and finally "Project end". A "Promethee user interface" is shown as a central hub connecting the input/output stages to the core processing and iteration loops.

■ **Contacts:**
yann.richet@irsn.fr
gregory.caplin@irsn.fr
matthieu.duluc@irsn.fr

IRSN

INSTITUT
DE RADIOPROTECTION
ET DE SÛRETÉ NUCLÉAIRE

Enhancing nuclear safety

Thank you for your attention



Enhancing nuclear safety