Plateforme Incertitude Uranie
Fonctionnalités et Applications

F. Gaudier

CEA/DEN/DANS/DM2S/SFME/LGLS

fabrice.gaudier@cea.fr

Workshop HPC/GdR MASCOT NUM

Quantification d'Incertitude et Calcul Intensif

LJK - Grenoble

28 Mars 2013

Plan

- Présentation du *"framework"* **ROOT**
- Présentation de la plateforme Incertitude et Optimisation **Uranie**
- Applications:
  - Analyse de Sensibilité à partir de Réseaux de Neurones
  - Réseaux de Neurones sous GPU
  - Application HEMERA/CCRT

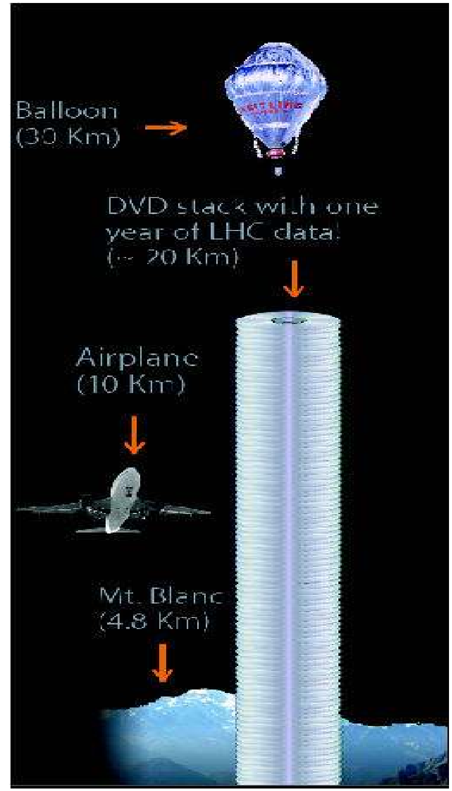
energie atomique • energies alternatives

CERN Large Hadron Collider (LHC)

- particle accelerator
- 27 km circumference tunnel in Geneva
- 4 experiments (ATLAS, CMS, ALICE, LHCb)

  Study the structure of matter

  - Search for the Higgs boson
  - Search for new physics
- Data quantity generated : 20 PetaBytes/year
- ROOT is the framework to store, treat and analyze this data

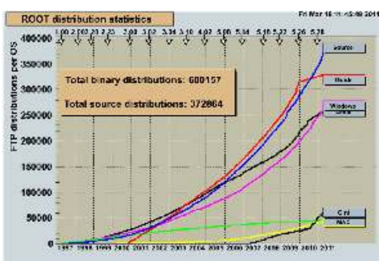


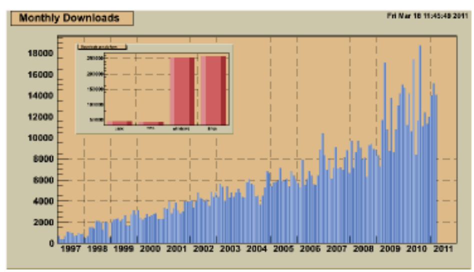
Fons Rademakers / CERN

ROOT (http://root.cern.ch)

ROOT is an object-oriented framework for large scale data analysis and data mining.

- 20 years of development (C++ with 3-4 releases/year)
- multi-plateform (Unix, Windows, Mac OS X)
- Offer :
  - A C++ interpreter, but also python (PyROOT), ruby
  - A hierarchical object-oriented database (machine independent, highly compressed, supporting schema evolution and object versioning)
  - Shared librairies (*automatic loading with "rootmap"*)
  - Advanced statistical analysis tools (subprojects *RooStats*, *RooFit*, *TMVA*)
  - Advanced visualization tools
- **LGPL License**

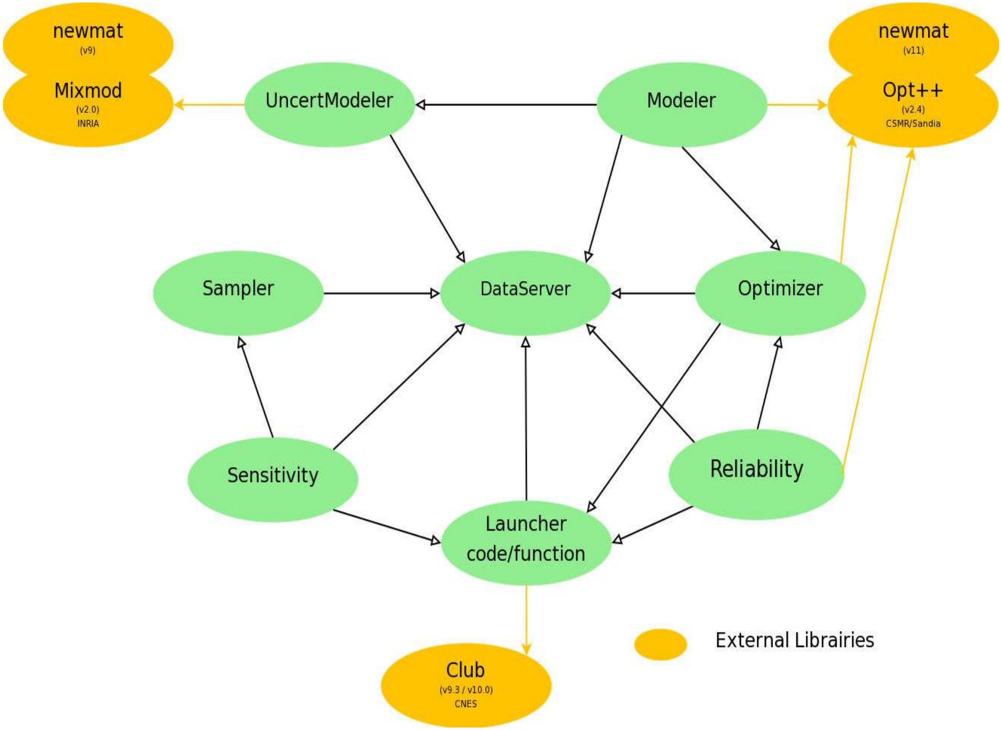


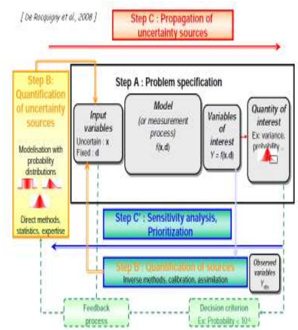


Uranie : CEA/DEN Uncertainty Platform

- Root (CERN), Mixmod (Gaussian Mixtures - INRIA), Opt++ (Optimization - Sandia), Club (Text parsing - CNES)
- Data access :
  - Flat file with header ("Salomé Table")
  - TTree (internal ROOT)
  - SQL Data base (MySQL, PostgreSQL, ...)
- Uncertainty/Sensitivity methods in URANIE
  - Design Of Experiments (SRS, LHS, ROA, qMC, MCMC, Copulas)
  - Clustering methods
  - Surrogate models (Polynomial, Artificial Neural Networks, Kriging, GLM, Splines)
  - Non Intrusive Spectrale Projection : Generalized Polynomial Chaos
  - Inverse Quantification of Uncertainty (Circe)
  - Sensitivity Analysis (Local, Morris, Regressions (*Pearson, Spearmann*), Sobol, FAST & RBD)
  - Optimization, Multi-Criteria (**Vizir** library : Genetic Algorithms)
  - Computing distribution

Uranie : Fonctional diagram

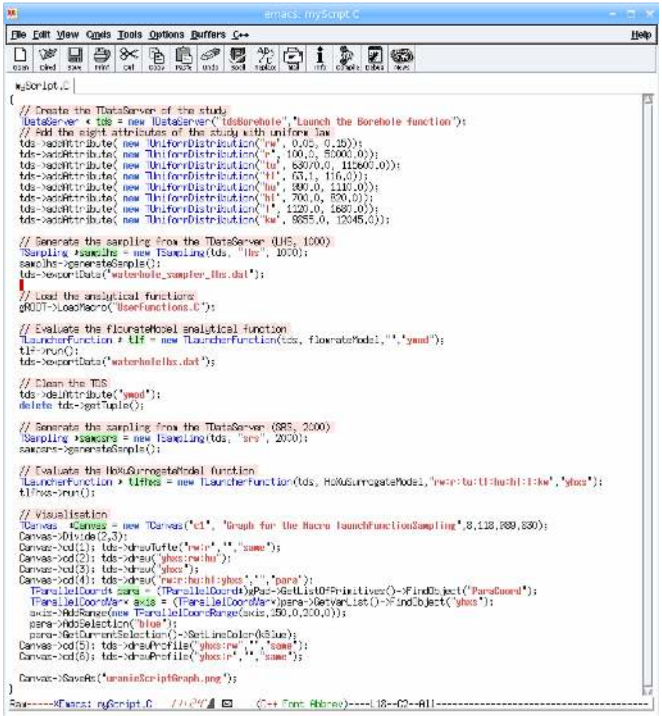


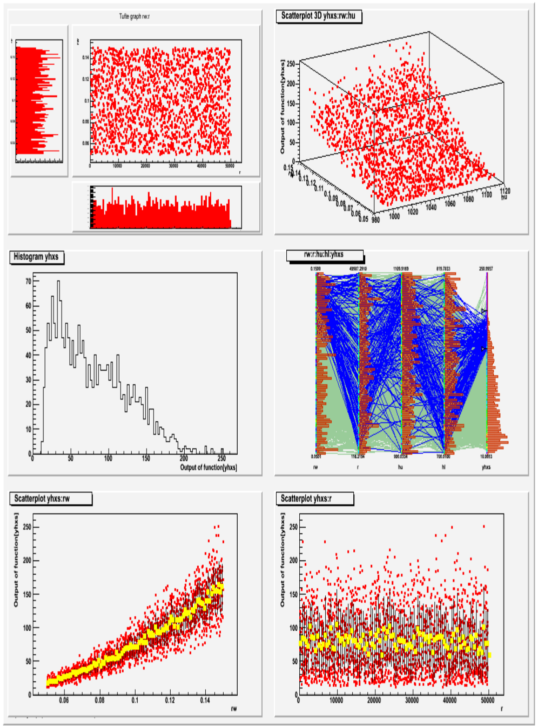


ROOT . . .
Uranie . . .
Launcher . . .
Applications

URANIE : Batch mode

> root myScript.C





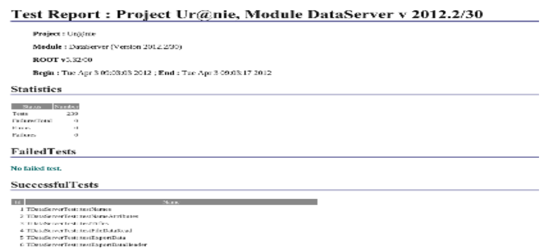
ROOT ...
Uranie ...
Launcher ...
Applications

Uranie : XML User Interface

```xml
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE Problem SYSTEM "/uranie.dtd">
<Problem>
  <Header name="boreholeXML" title="Launch the Borehole function in XML" debug="0">
    <Application name="uranie" version="1.0"/>
  </Header>
  <DataDictionary>
    <DataField name="rw" law="uniform" min="0.05" max="0.15"/>
    <DataField name="r"  law="uniform" min="100.0" max="50000.0"/>
    <DataField name="tu" law="uniform" min="63070.0" max="115600.0"/>
    <DataField name="tl" law="uniform" min="63.1" max="116.0"/>
    <DataField name="hu" law="uniform" min="990.0" max="1110.0"/>
    <DataField name="hl" law="uniform" min="700.0" max="820.0"/>
    <DataField name="l"  law="normal" mean="1120.0" std="12.25"/>
    <DataField name="kw" law="uniform" min="9855.0" max="12045.0"/>
  </DataDictionary>
  <Sampler method="LHS" N="1000" export="waterhole_sampler_lhs.dat"/>
  <Launcher macro="UserFunctions.C" function="flowrateModel" output="ymod" export="waterholelhs.dat"/>
  <Sampler method="SRS" N="2000"/>
  <Launcher function="HoXuSurrogateModel" input="rw:r:tu:tl:hu:hl:l:kw" output="yhxs"/>
</Problem>
```


```cpp
void evaluateXMLFile (TString xmlFile = "uranieproblem.xml")
{
TXMLProblem * xmlProblem = new TXMLProblem(xmlFile);
xmlProblem->submit();
}
```

- **93 200** lines & **226** classes
- Version of ROOT :
  v5.34 (2012 June)
  v5.32 (2011 Dec.)
- Compilation with **cmake**
  (Linux-Makefiles/Windows-Visual Project)
- Unitary tests with **CppUnit**
- Coverage with **gcov** in **CDash**
- **Exceptions**
  (Warning, Error, Deprecated)
- **User Manual** with **DocBook**
  Generate the HTML and PDF documents from the same `XML` files
- **Developper Manual** with **Doxygen**


Test Report : Project Ur@nie, Module DataServer v 2012.2/30

ROOT v5.32/00

Statistics

FailedTests

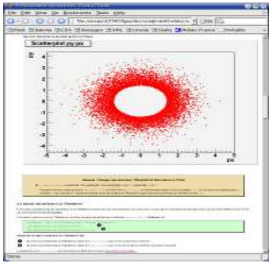
No failed test.

SuccessfulTests

XML Resume for tests

```
--- <WARNING> URANIE::MessageLogger : ==================== WARNING ========================
--- <WARNING> URANIE::MessageLogger : Deppreciated constructor since v0.3 to v0.5
--- <WARNING> URANIE::MessageLogger : Using the same constructor with a TDataServer object
--- <WARNING> URANIE::MessageLogger : ================= End Of Warning ===================
```

Deprecated message



HTML document

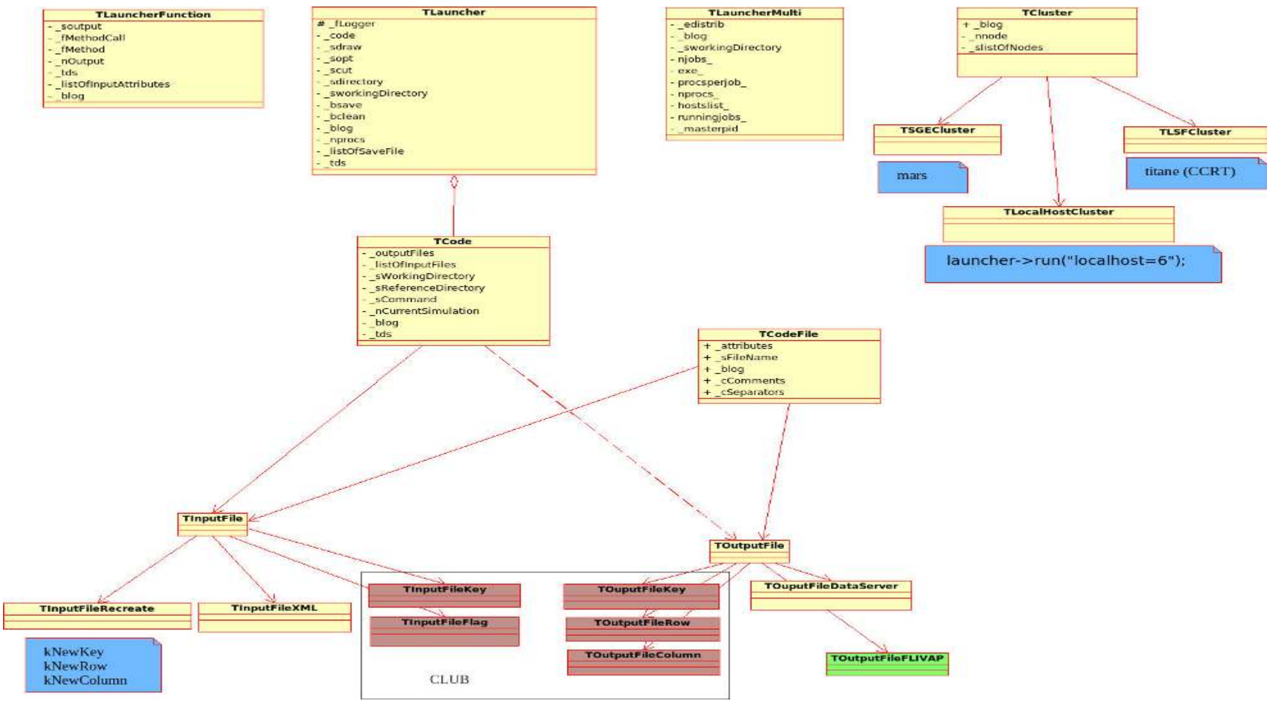
Projects using Uranie

- LEONAR tool for severe accidents in french nuclear reactor (**CEA-EDF**)
- PSI-Matador Methodology : Dosimetry computation in french nuclear reactor (**CEA-EDF**)
- EHPOC project : Meteor/Pleiades codes (**CEA DEC**)
- SIVIT project : (**CEA DTCD**)
  - ◇ Simulation de la vitrification
- Multi-criteria optimization (**CEA CESTA/CELIA/TRIAD**)
- Sensitivity Analysis for Cathare code (**Areva TA**)
- ALLIANCES platform (**CEA/ANDRA/EDF**)
  - ◇ is to provide a working environment for the simulation and analysis of phenomena to be taken into account for waste storage and disposal studies
- European project **NURESIM/NURISP**
  - ◇ The European Platform for NUclear REactor SIMulations, NURESIM, is a Common European Standard Software Platform for modeling, recording, and recovering computer data for nuclear reactors simulations

"Launcher" library

Distribute the model evaluations (sequential, cluster) for:

- Analytical function
- External code



energie atomique • energies alternatives

ROOT . . .
Uranie . . .
Launcher . . .
Applications

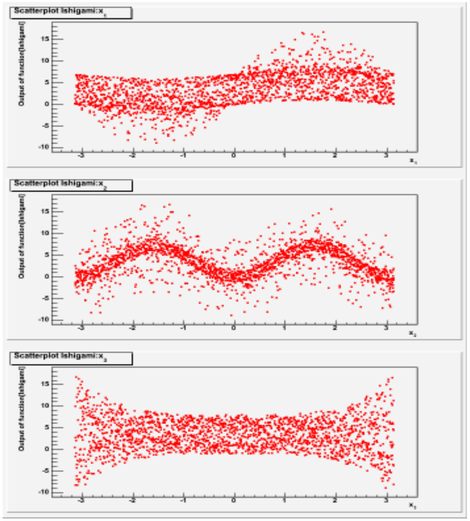
Analytical Function

void myFunction (double *x, double *y)

Ishigami benchmark : Analytical function of $R^3 \rightarrow R$ $(x_{i=1,2,3} \sim Unif[-\pi, \pi])$

$$y = \sin x_1 + A \sin^2 x_2 + B x_3^4 \sin x_1 \qquad A = 7, B = 0.1$$

```
#include "TMath.h"

void Ishigami(double *x, Double_t *y)
{
  Double_t A = 7.0, B = 0.1;

  Double_t x1 = x[0];
  Double_t x2 = x[1];
  Double_t x3 = x[2];

  y[0]  = Sin(x1) + A * Sin(x2) * Sin(x2) + B * Power(x3, 4.) * Sin(x1);
}

{
  TDataServer * tds = new TDataServer("tds", "TDS for Ishigami function");
  tds->addAttribute ( new TUniformDistribution("x_{1}",-1.* Pi(), Pi()));
  tds->addAttribute ( new TUniformDistribution("x_{2}",-1.* Pi(), Pi()));
  tds->addAttribute ( new TUniformDistribution("x_{3}",-1.* Pi(), Pi()));

  TSampling *sampling = new TSampling(tds, "lhs", 1000);
  sampling->generateSample();

  TLauncherFunction * tlf = new TLauncherFunction(tds, Ishigami);
  tlf->run();
}
```



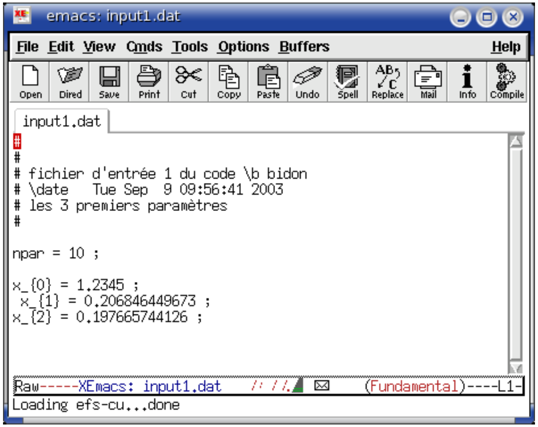
ROOT . . .

Uranie . . .

Launcher . . .

Applications

External code : Input Files with " **Key - Value**" 1/4




```
TAttribute *x1 = new TAttribute("x_{1}", 0.20, 0.40);
x1->setFileKey("input1.dat");

TAttribute *x2 = new TAttribute("x2", 0.15, 0.25);
x2->setFileKey("input1.dat", "x_{2}");
```

energie atomique • energies alternatives

ROOT . . .

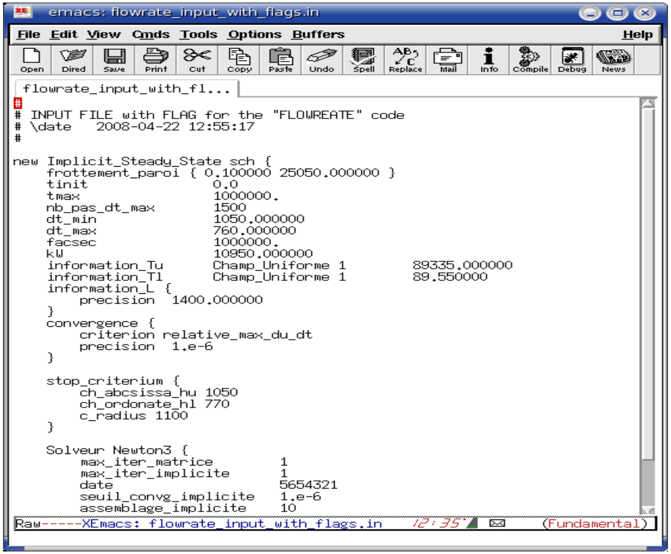
Uranie . . .

Launcher . . .

Applications

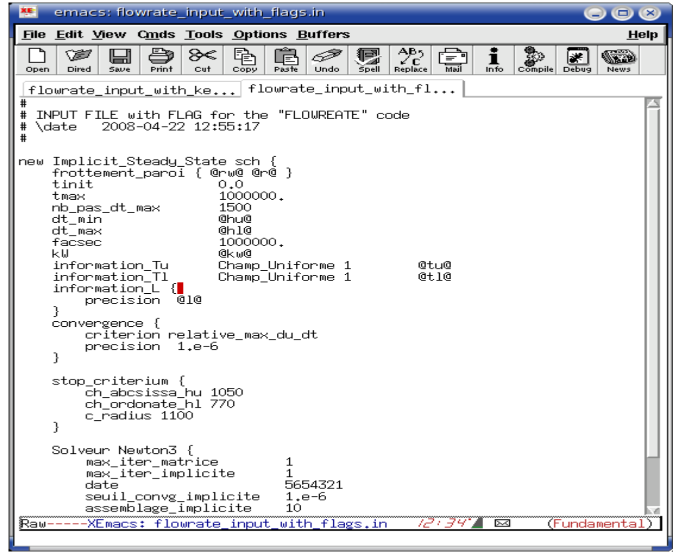
Hypothesis : unicity of the key

```
void TAttribute::setFileKey(
    TString sfile,
    TString skey="",
    TString sformatToSubstitute="%e",
    TAttributeFileKey::EFileType FileType=TAttributeFileKey::kKey);
```



```
emacs: flowrate_input_with_flags.in
File Edit View Cmds Tools Options Buffers Help
flowrate_input_with_fl...
#
# INPUT FILE with FLAG for the "FLOWREATE" code
# \date    2008-04-22 12:55:17
#

new Implicit_Steady_State sch {
    frottement_paroi { 0.100000 25050.000000 }
    tinit               0.0
    tmax                1000000.
    nb_pas_dt_max       1500
    dt_min              1050.000000
    dt_max              760.000000
    facsec              1000000.
    kW                  10950.000000
    information_Tu      Champ_Uniforme 1        89335.000000
    information_Tl      Champ_Uniforme 1        89.550000
    information_L  {
        precision   1400.000000
    }
    convergence {
        criterion relative_max_du_dt
        precision   1.e-6
    }

    stop_criterium {
        ch_abcsissa_hu 1050
        ch_ordonate_hl 770
        c_radius 1100
    }

    Solveur Newton3 {
        max_iter_matrice        1
        max_iter_implicite      1
        date                    5654321
        seuil_convg_implicite   1.e-6
        assemblage_implicite    10
Raw-----XEmacs: flowrate_input_with_flags.in    12:35    (Fundamental)
```

Original file



```
emacs: flowrate_input_with_flags.in
File Edit View Cmds Tools Options Buffers Help
flowrate_input_with_ke...  flowrate_input_with_fl...
#
# INPUT FILE with FLAG for the "FLOWREATE" code
# \date    2008-04-22 12:55:17
#

new Implicit_Steady_State sch {
    frottement_paroi { @rw@ @r@ }
    tinit               0.0
    tmax                1000000.
    nb_pas_dt_max       1500
    dt_min              @hu@
    dt_max              @hl@
    facsec              1000000.
    kW                  @kw@
    information_Tu      Champ_Uniforme 1        @tu@
    information_Tl      Champ_Uniforme 1        @tl@
    information_L  {
        precision   @l@
    }
    convergence {
        criterion relative_max_du_dt
        precision   1.e-6
    }

    stop_criterium {
        ch_abcsissa_hu 1050
        ch_ordonate_hl 770
        c_radius 1100
    }

    Solveur Newton3 {
        max_iter_matrice        1
        max_iter_implicite      1
        date                    5654321
        seuil_convg_implicite   1.e-6
        assemblage_implicite    10
Raw-----XEmacs: flowrate_input_with_flags.in    12:34    (Fundamental)
```

User Flag file

```
attrw->setFileFlag(”myfile.in”, ”@tu@”);
attrw->setFileKey(”myfile.in”, ”@tu@”, ”%f”,TAttributeFileKey::kFlag);
```

Hypothesis : unicity of the key not required but intervention of the user

energie atomique • energies alternatives

ROOT . . .
Uranie . . .
Launcher . . .
Applications

External code : Input Files in ”XML” 3/4

```
<?xml version="1.0"?>
<problem>
  <description name="flowrate" title="UseCase flowrate with XML input file" version="1.0" date="2011-07-22">
    <tool name="uranie" version="v2.3"/>
  </description>
 <steady_state name="sch">
   <wall_friction rw="0.141777" r="6829.9"/>
   <tinit>0.0</tinit>
   <tmax>1000000</tmax>
   <nb_pas_dt_max>1500</nb_pas_dt_max>
   <parameter>
      <tonode>mesh</tonode><toport>dt_hu</toport>
       <value><double>1088.66</double></value>
   </parameter>
   <parameter>
      <tonode>mesh</tonode><toport>dt_hl</toport>
       <value><double>786.653</double></value>
...
```

```
attrw->setFileXMLAttribute("input.xml", "wall_friction/@rw");
atthu->setFileXMLField("input.xml", "parameter[tonode='mesh and toport='dt_hu']/value/double");
```

Hypothesis : unicity of the key not required and no intervention of the user

- **TAttributeFileKey::kNewRow** 4^{th} argument in the **setFileKey** method

```
2.481733e+02 6.112975e-03 1.055352e-06 2.635758e-03 2.217372e+02 1.888999e+00 ...
```

- **TAttributeFileKey::kNewColumn** 4^{th} argument in the **setFileKey** method

```
2.481733e+02
6.112975e-03
...
```

- **TAttributeFileKey::kNewKey** 4^{th} argument in the **setFileKey** method

```
t = 2.481733e+02 ;
kl = 6.112975e-03 ;
kc = 1.055352e-06 ;
...
```

Hypothesis : The input files does not exist

External code : Output Files formats 1/2

- **TOutputFileRow** class

```
0.20E+05 0.5579978E-25 0.2789989E-25
0.30E+05 0.5121863E-20 0.2560931E-20
0.40E+05 0.8212720E-17 0.4106360E-17
0.50E+05 0.1432418E-14 0.7162090E-15
...
```

- **TOutputFileColumn** class

```
0.20E+05 0.30E+05 0.40E+05 0.50E+05 ....
0.5579978E-25 0.5121863E-20 0.8212720E-17 0.1432418E-14 ...
0.2789989E-25 0.2560931E-20 0.4106360E-17 0.7162090E-15 ...
...
```

External code : Output Files formats 2/2

- **TOutputFileKey** class

```
yhat = 3.591931e+01;
d = 2.415401e+03;
...
```

- **TOutputFileDataServer** class

```
#COLUMN_NAMES: yhat | d

3.591931e+01 2.415401e+03
...
```

Distribution

- PC multicores

```
launcher->run("localhost=5");
```

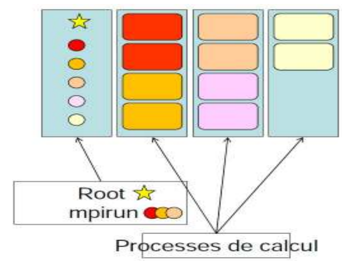
- Cluster (LSF, SGE)

```
#BSUB -n 10
#BSUB -J FlowrateSampling
#BSUB -o FlowrateSampling.out
#BSUB -e FlowrateSampling.err
source /home/cont002/uranie/uranie-titane.cshrc
rm -f FlowrateSampling.out FlowrateSampling.err
root -l -q lanceurFLOWRATE_SAMPLING.C
```

> bsub < BsubFile

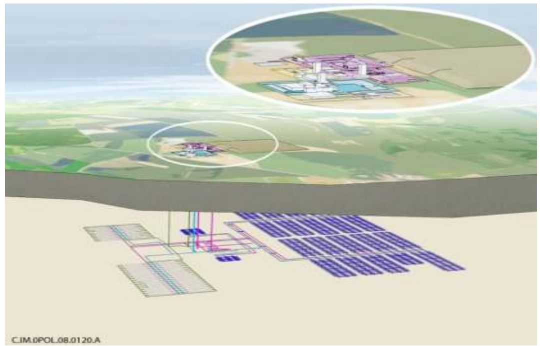
Utilisation d'Uranie au CCRT

- Le mécanisme de lancement des calculs dans URANIE est transparent pour l'utilisateur : le script URANIE est le même que l'on soit sur un poste de travail ou sur le CCRT;
- La séquence est la suivante :
  - Le plan d'expérience est généré (en fonction de la méthode et des incertitudes sur les paramètres d'entrée)
  - URANIE analyse la machine locale à l'aide de variables d'environnement et déduit le nombre de processeurs disponibles
  - Un pool de processeurs est ensuite géré pour répartir les calculs au fur et à mesure sur les processeurs disponibles
- Travail réalisé par les équipes du Support Applicatif CCRT;
- Difficulté liée au fait qu'il est impossible de lancer *mpirun* depuis *mpirun*;
- Méthode choisie :
  - Le noeud maître gère la distribution des calculs au fur et à mesure;
  - Lorsqu'un groupe de processeurs est disponible, le process maître est forké et lance un *mpirun*;
  - La fin de l'exécution du cas est détectée en analysant l'état du process fils.



Applications (1/3)

- Analyse de Sensibilité à partir de Réseaux de Neurones
  - G. Pepin (ANDRA)



- Réseaux de Neurones sous GPU
  - C. Canonne
- Application HEMERA/CCRT
  - V. Bergeaud, N. Crouzet, J.C. Le Pallec, C. Delavaud

energie atomique • energies alternatives

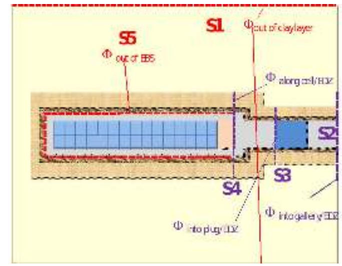
Uranie Application with ANDRA : GdR MoMaS

- Context

  Make Sensitivity Analysis with Surrogate Model : Artificial Neurals Networks ("ANN")

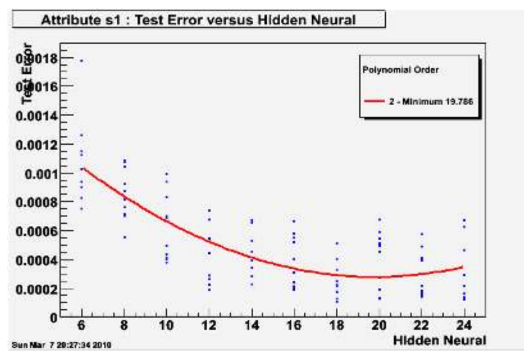
- CPU time single calculation : $1 < t < 5$ hours (Cluster)

- Original DataSet
  - nX = 17 input variables
  - nY = 5 output variables
  - Build the 5 ANN : nS = 1500 patterns



Guillaume Pépin (ANDRA)

- Neural networks
  - different architectures (MLP)
  - Input with a logarithm PDF $x := \log(x)$
  - Output $y := \frac{1}{1+log(y)}$
  - cross validation
  - Validate the 5 ANN on another dataset with nS = 1000 patterns



Computation Sensitivity Indexes (SI)

First Order : $$S_i = \frac{Var[\,\mathbb{E}[\,Y \mid x_i\,]\,]}{Var[Y]}$$

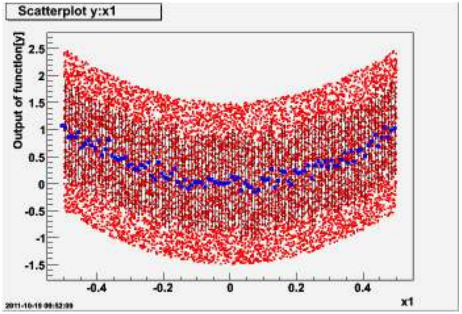
1. **Regression Analysis** $$\hat{y} = b_0 + \sum b_i\, x_i$$

   – on the Values : "SRC" ("Standardised Regression Coefficient")
   – on the Ranks : "SRRC" ("Standardised Rank Regression Coefficient")

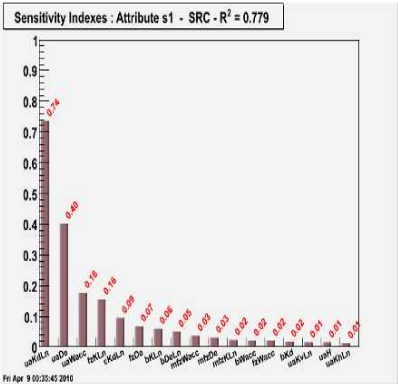
   The regression is valid when $R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$ close to 1.0 (≥ 0.7)

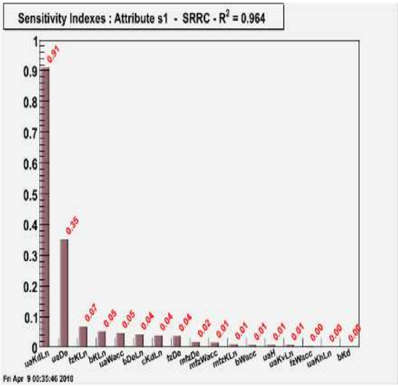
2. **Brute-Force Method**
   Cost : nX * nCV * nPts

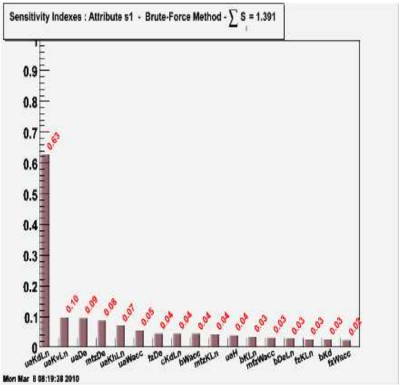
   – nX = number of inputs parameters
   – nCV = n. of conditionnal values
   – nPts = n. of pts over cond. values

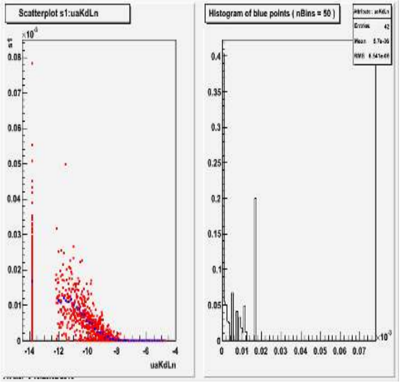


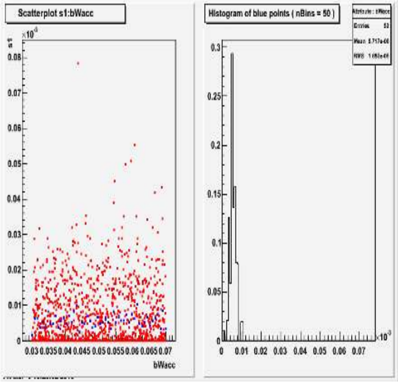
Original Database (nS = 1500) & ”s1”

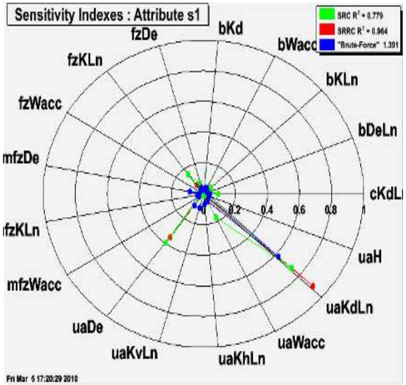












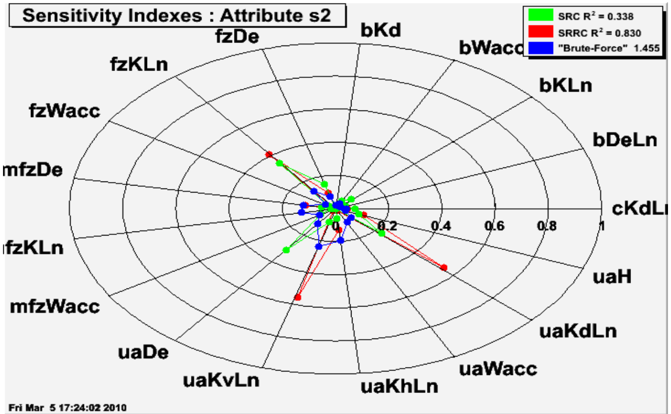
ROOT . . .

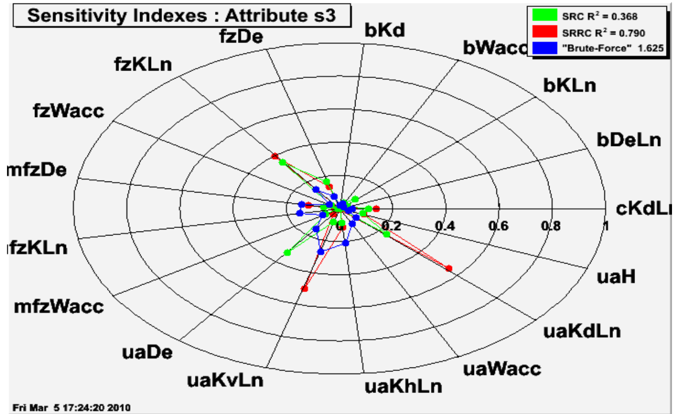
Uranie . . .

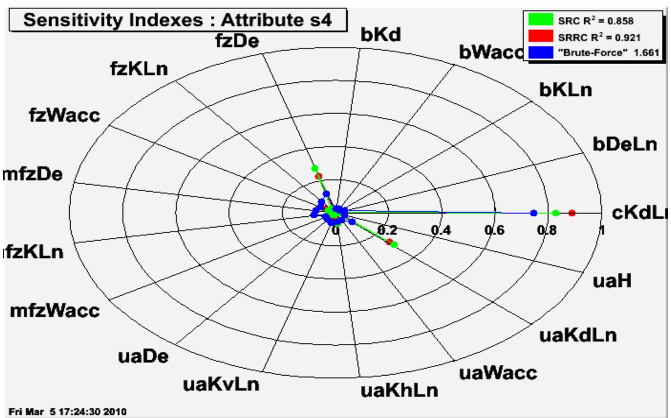
Launcher . . .

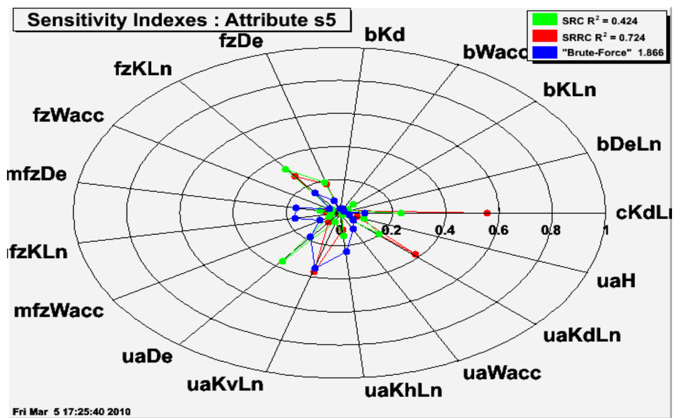
Applications

Original Database (nS = 1500) & other outputs









energie atomique • energies alternatives

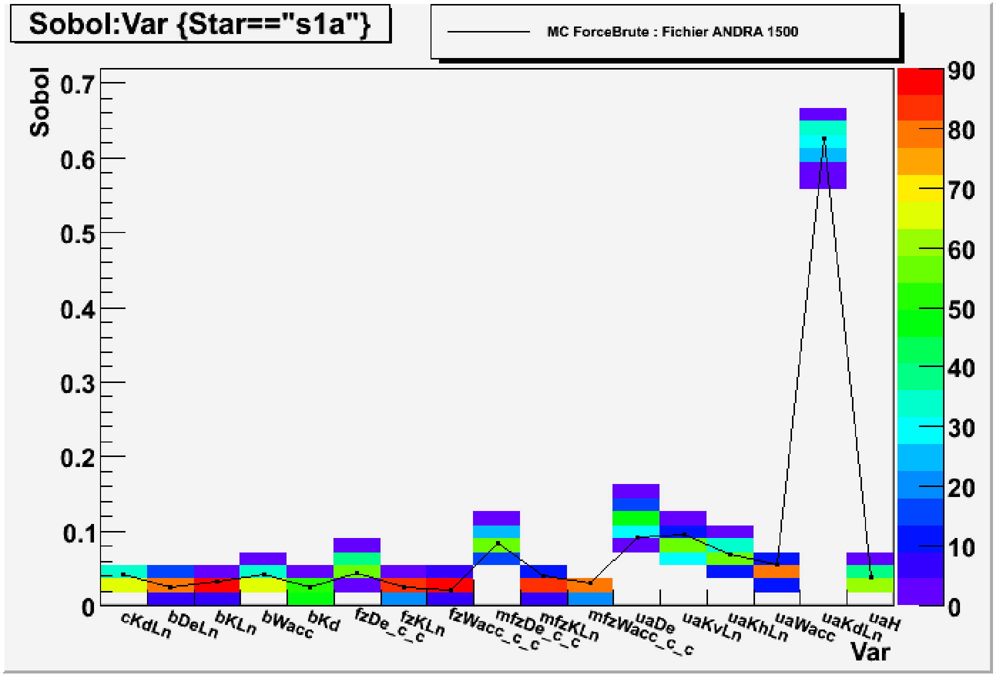
ROOT . . .
Uranie . . .
Launcher . . .
Applications

Estimation of SI with ANN and nS = 1500

Design nL = 100 databases with size nS = 1500 points

Simulate these databases using ANN surrogate models

Compute the first order SI by the Brute-Force method



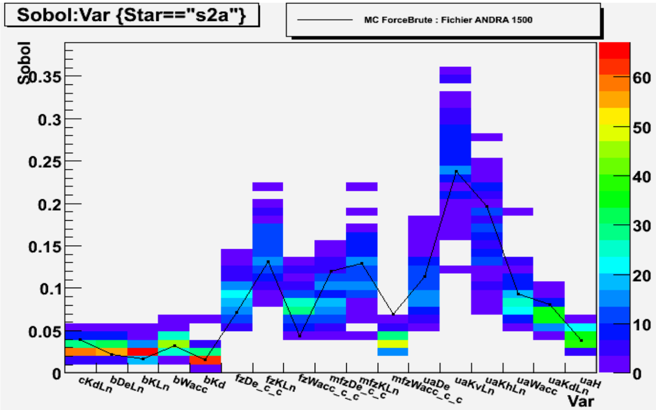
ROOT ...

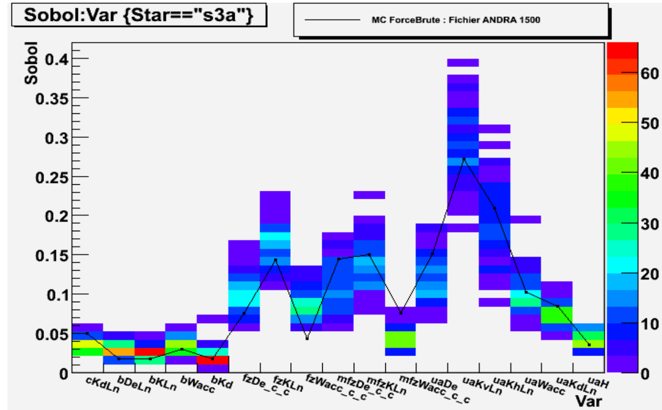
Uranie ...

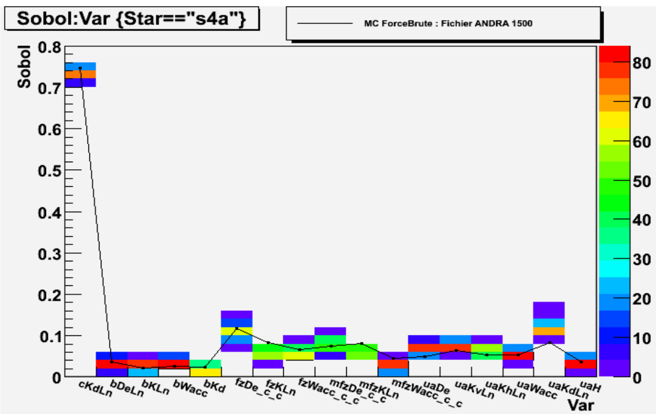
Launcher ...

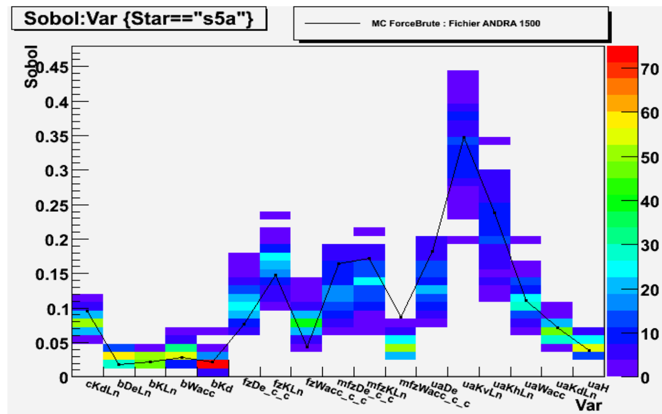
Applications

Estimation of SI with ANN and nS = 1500



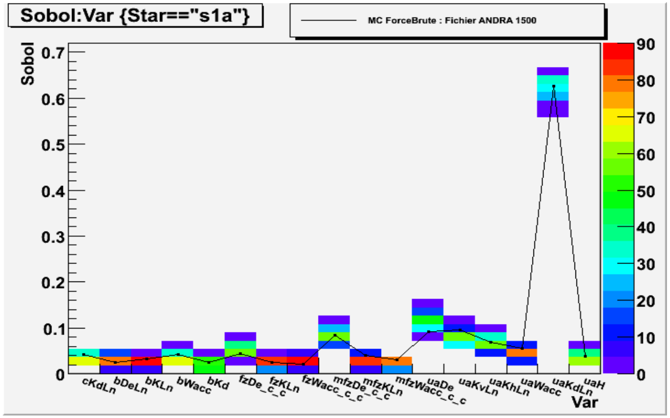


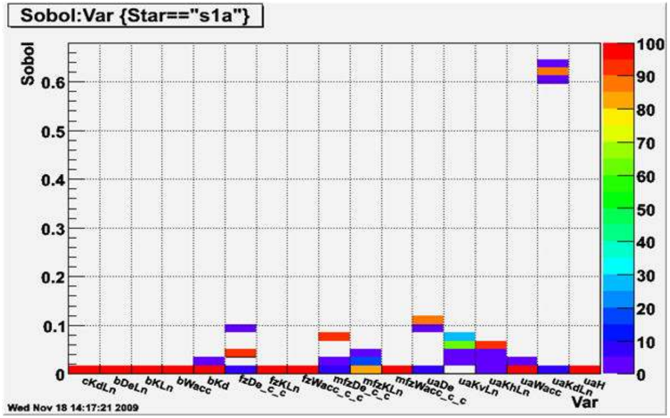




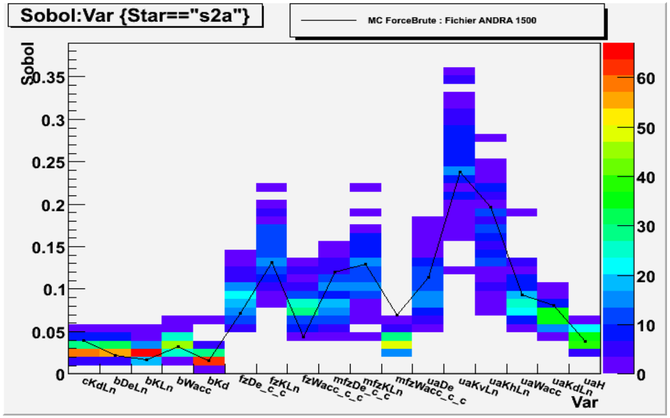
ROOT ...
Uranie ...
Launcher ...
Applications

Same with nS = 20000 (« s1 » and « s2 »)

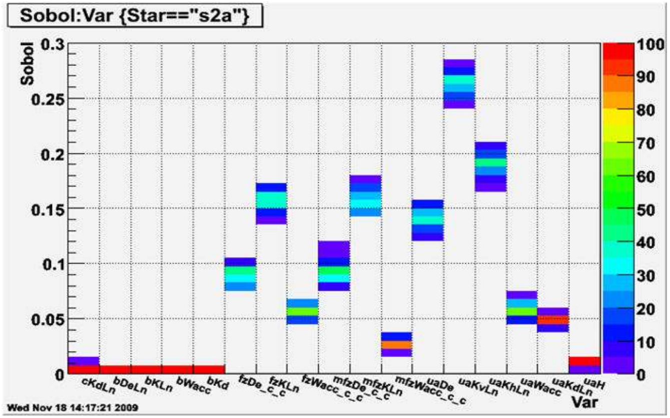




ROOT . . .
Uranie . . .
Launcher . . .
Applications

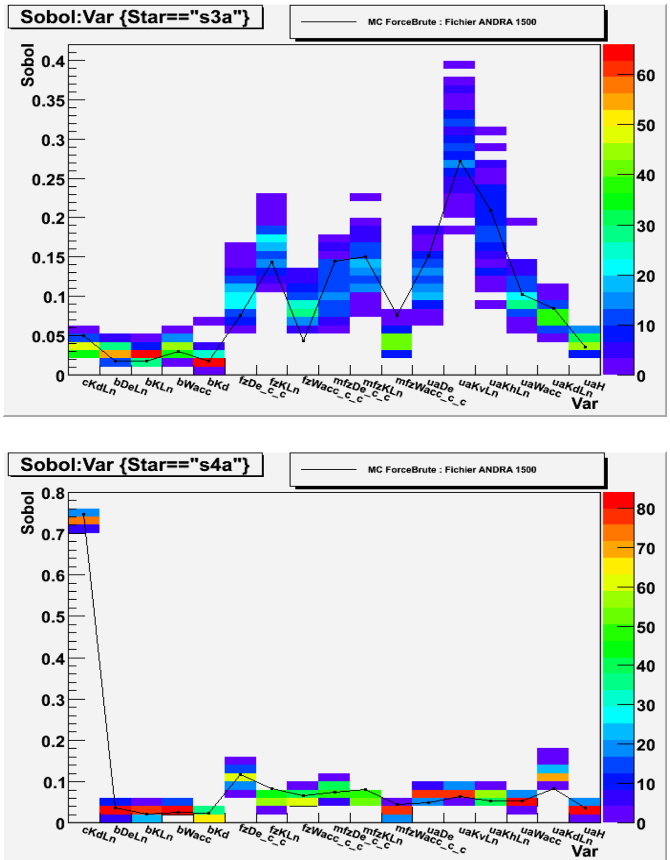


1 500 points

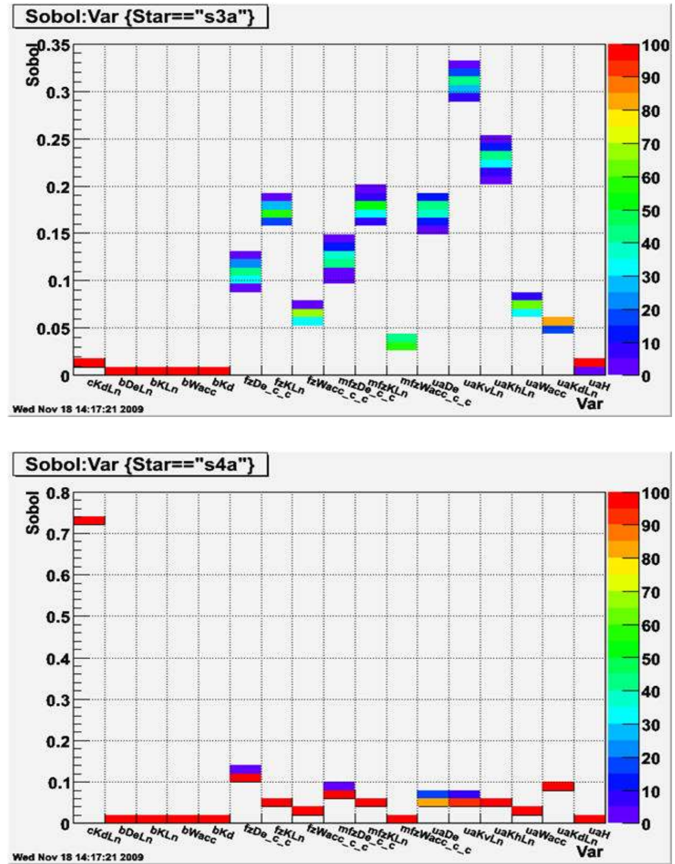


20 000 points

Same with nS = 20000 (« s3 » and « s4 »)



1 500 points



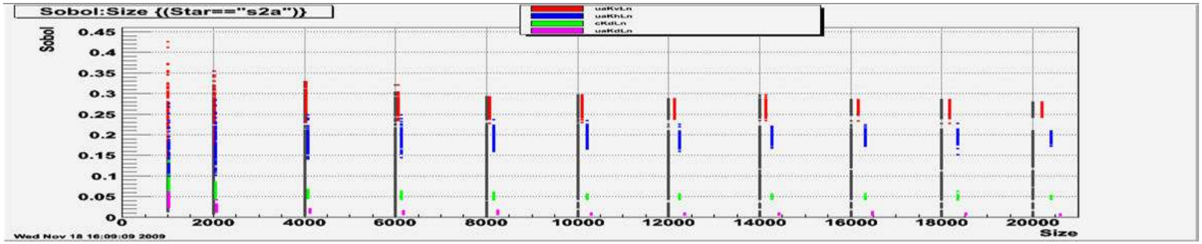
20 000 points

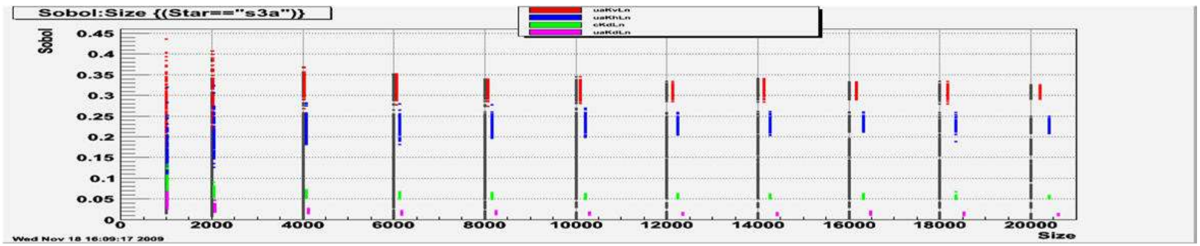
energie atomique • energies alternatives

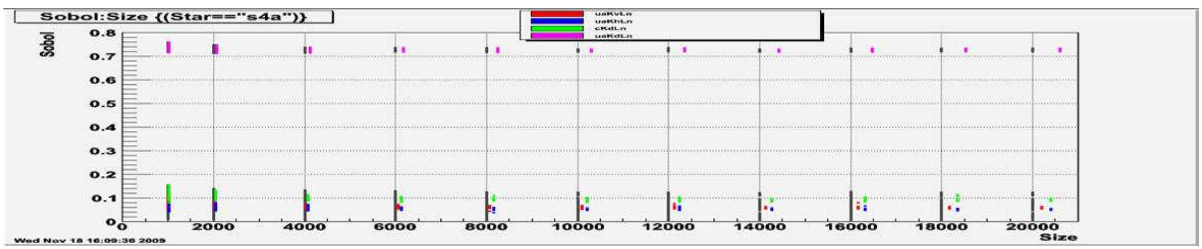
ROOT . . .
Uranie . . .
Launcher . . .
Applications

Uranie Application : All simulations (1/2)

Same loop (nL = 100 DoE) but for nS=2000 to 20000 by step 2000 :

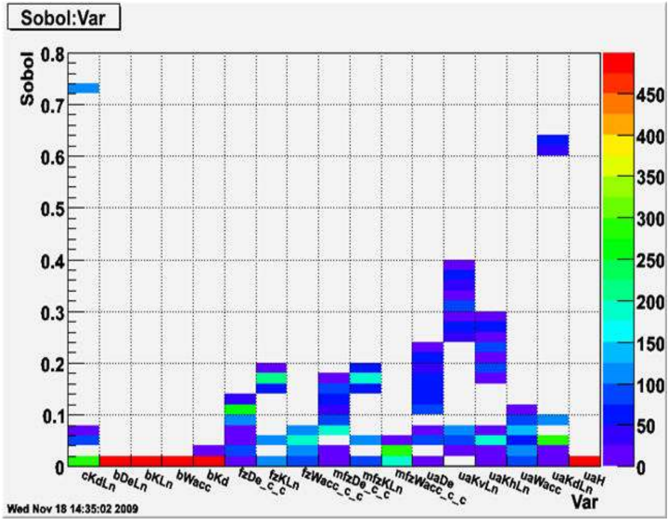




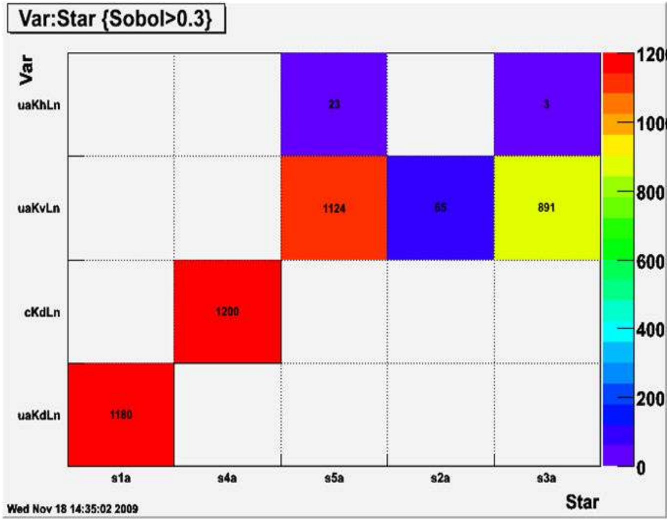


Uranie Application : All simulations (2/2)

1. 5 inputs parameters are not influence (first order) on all the 5 outputs
2. Only 4 inputs parameters are sensitives (for all 5 outputs)



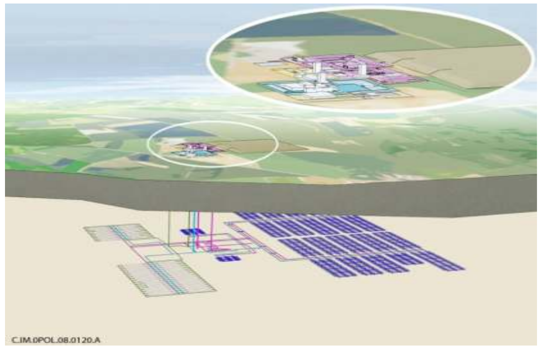
ttre->Draw("Sobol:Var","","BOXCOL2Z")



ttre->Draw("Var:Star","Sobol>0.3","BOXCOL2Ztext")

Applications (2/3)

- Analyse de Sensibilité à partir de Réseaux de Neurones
  - G. Pepin (ANDRA)



- Réseaux de Neurones sous GPU
  - C. Canonne
- Application HEMERA/CCRT
  - V. Bergeaud, N. Crouzet, J.C. Le Pallec, C. Delavaud

GPU - Description of the test

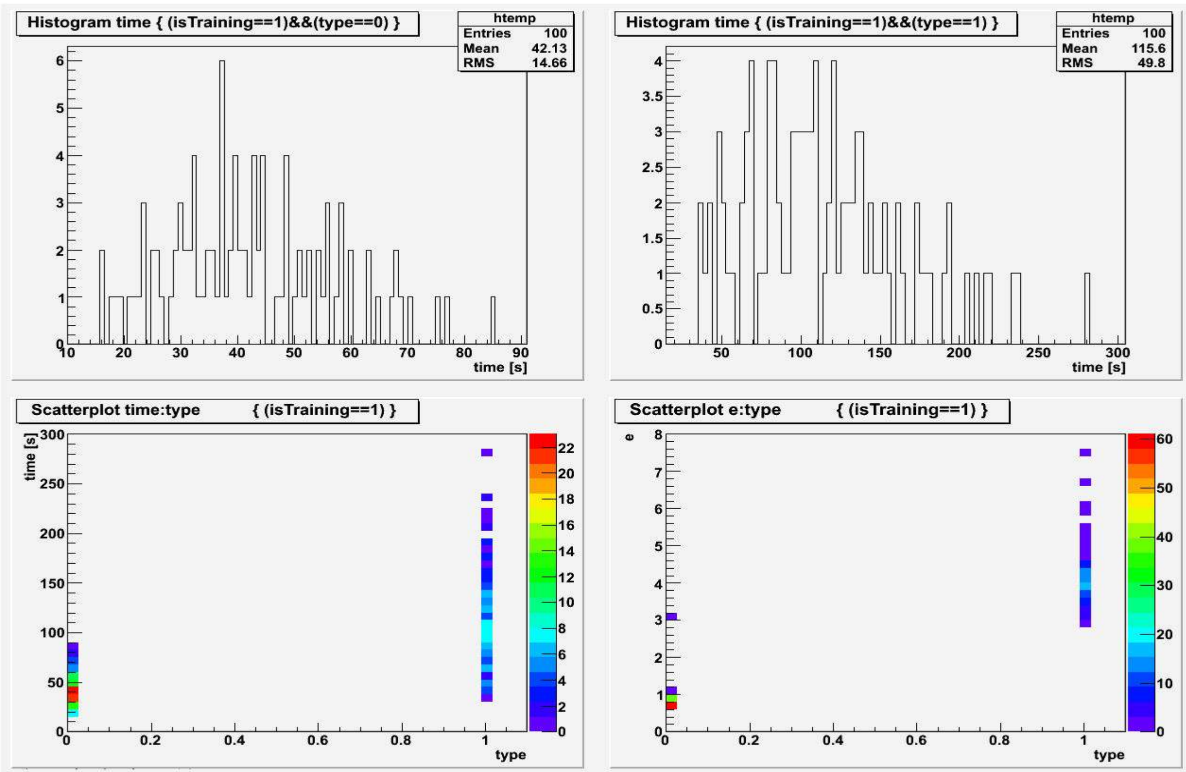
- Output : **st4** with $nH = 16$ neurons in the hidden layer ($\mathbb{R}^{17} \rightarrow \mathbb{R}^{16} \rightarrow \mathbb{R}$)
- Learning :
  - 1 database with 1 400 patterns
  - 100 learning
- Running :
  - 6 databases with 1 500, 5 000, 10 000, 20 000 and 50 000 patterns
  - 10 running for each databases
- criteria for comparison : time (t) and error (e)

1. CPU : 2 quadri cores Xeon 5500 (Boost.uBLAS)
2. GPU : 8 Fermi Tesla C2050 (CUDA/cuBLAS)

energie atomique • energies alternatives

GPU - Learning Performance

Mean : GPU 42 s versus CPU 115 s



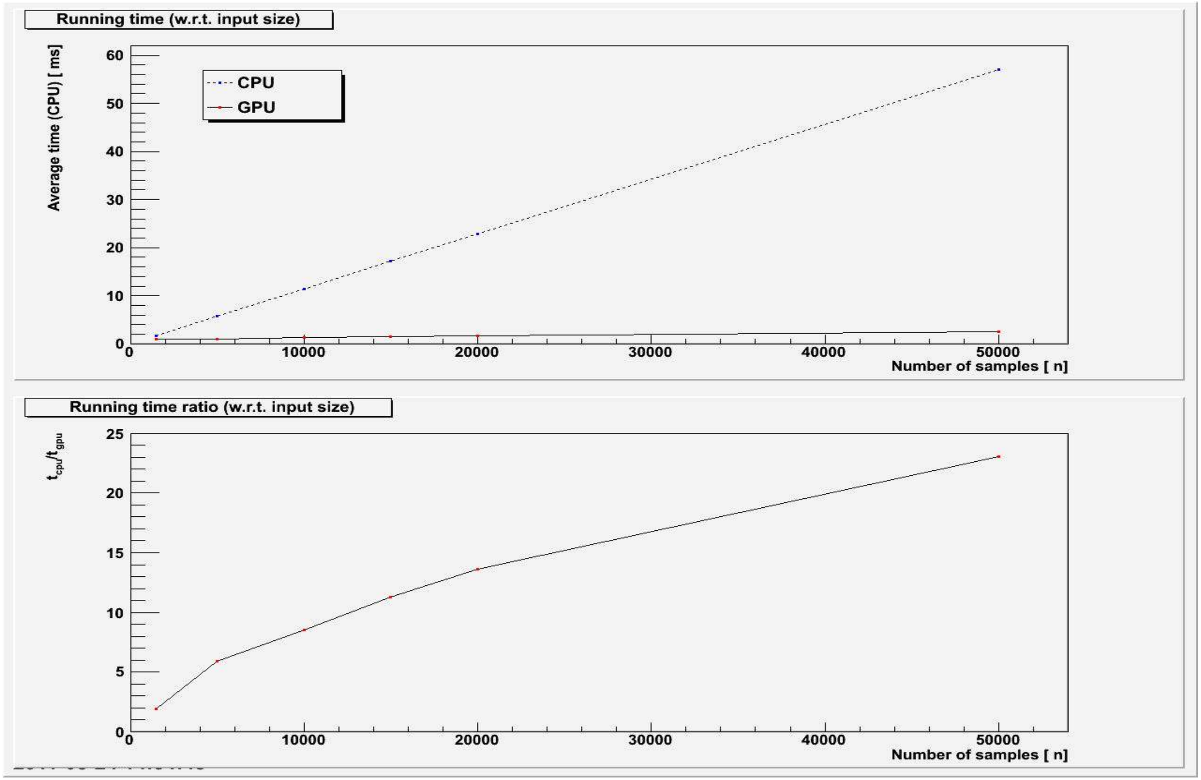
ROOT . . .

Uranie . . .

Launcher . . .

Applications

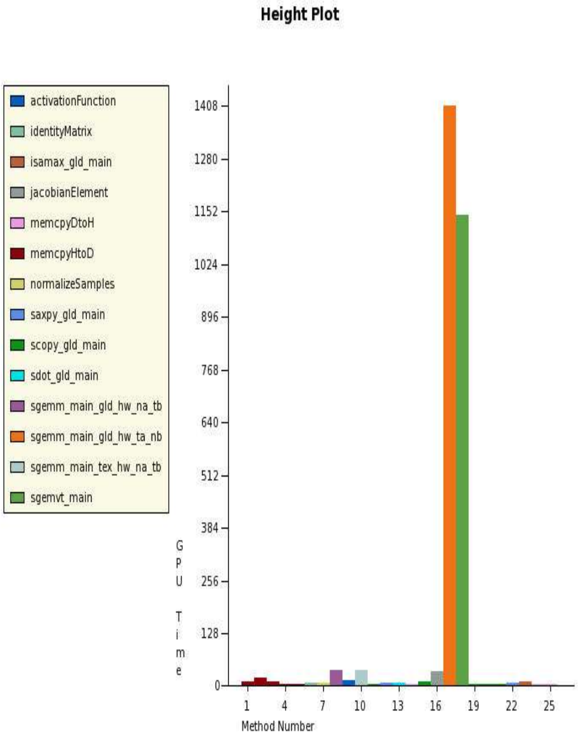
GPU - Evaluation Performance

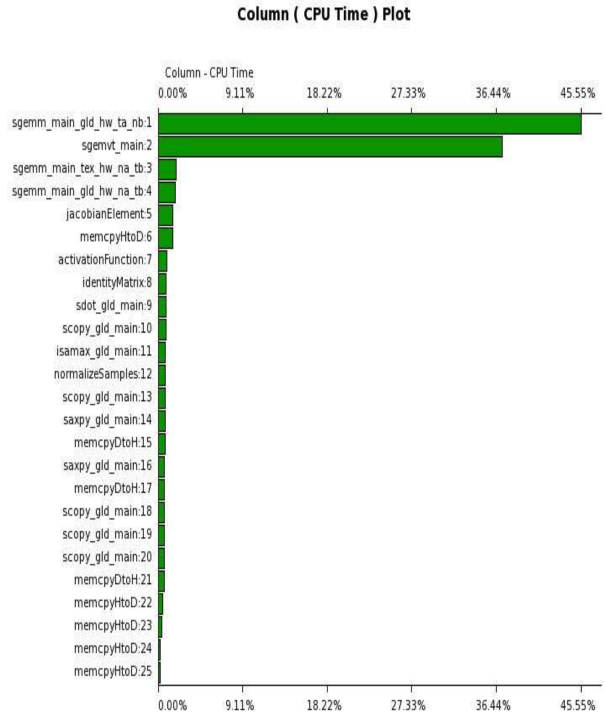


energie atomique • energies alternatives

ROOT ...
Uranie ...
Launcher ...
Applications

GPU - Profiling





energie atomique • energies alternatives

ROOT . . .

Uranie . . .

Launcher . . .

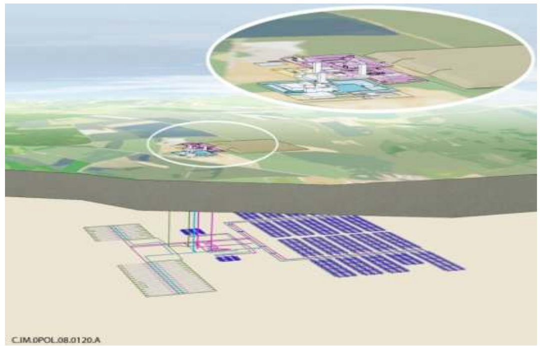
Applications

GPU - Conclusions

1. GPU is 2 $\longrightarrow$ 23 faster than CPU for evaluation ANN
2. GPU is only 3 faster than CPU for learning ANN for small database
   same speed-up for large database ?
3. main time consumer : matrix computing

Applications (3/3)

- Analyse de Sensibilité à partir de Réseaux de Neurones
  - G. Pepin (ANDRA)



- Réseaux de Neurones sous GPU
  - C. Canonne
- Application HEMERA/CCRT
  - V. Bergeaud, N. Crouzet, J.C. Le Pallec, C. Delavaud

energie atomique • energies alternatives

Cas d'étude

- Analyse de sensibilité sur un scénario accidentel
- Séquence d'analyse
  - Mise en place d'un modèle (code CRONOS)
  - Définition des paramètres incertains
  - Evaluation des incertitudes
  - Apprentissage d'un méta-modèles "Réseau de neurones"
  - Propagation des incertitudes sur le méta-modèles
  - Calcul des indices de sensibilité sur les variables de sorties

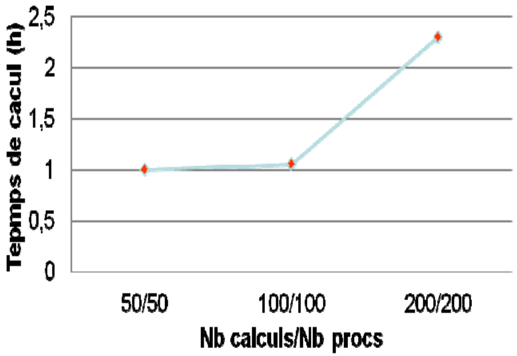
energie atomique • energies alternatives

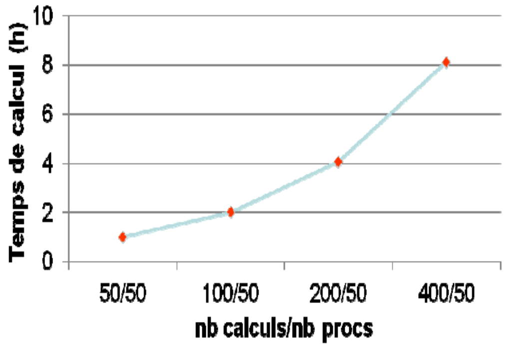
Paramètres d'entrée

	Nom du paramètre	Désignation	Unité	Domaine de variation
Grandeurs neutroniques	Poids de la grappe éjectée	ρ_{grappe}	pcm	ρ_0+/-10%
	Fraction de neutrons retardés	β	pcm	β_0+/-5%
	Coefficient Doppler	α_D	pcm/K	α_{D0}+/-20%
	Coefficient modérateur	α_m	pcm/g/cm^3	α_{m0}+/-30%
Grandeurs thermiques	Capacité calorifique du combustible	C_{comb}	J/kg/K	C_{comb}+/-10%
	Capacité calorifique de la gaine	C_{gaine}	J/kg/K	C_{gaine}+/-10%
	Conductivité thermique du combustible	λ_{comb}	W/m/K	λ_{comb}+/-10%
	Conductivité thermique de la gaine	λ_{gaine}	W/m/K	λ_{gaine}+/-10%

Passage du Plan d'Expérience sur le CCRT

- Tests de diffétentes configurations pour pass-er les calculs sur Titane
- Temps de calcul anormalement élevé
  - Augmentation suspecte au-delà de 100 calculs



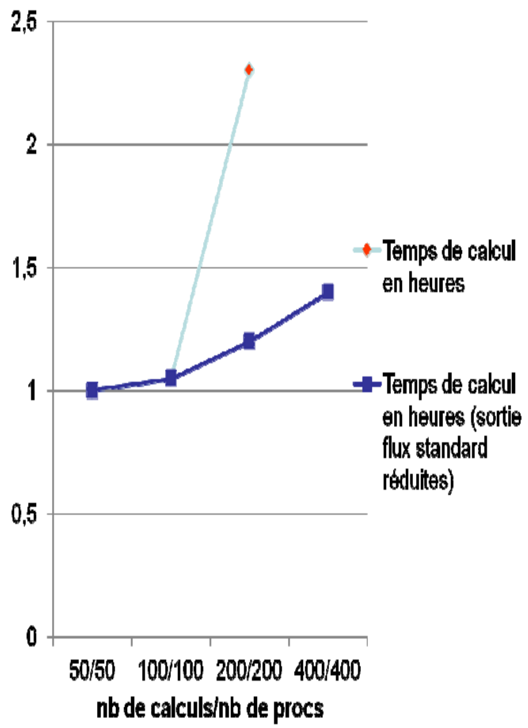


energie atomique • energies alternatives

ROOT ...
Uranie ...
Launcher ...
Applications

Passage de cas sur le CCRT (en réfléchissant un peu)

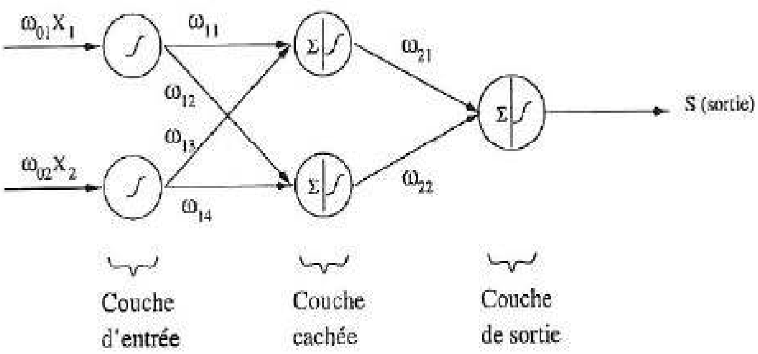
- En diminuant les sorties sur le flux standard, on retrouve un comportement acceptable
- Néanmoins, on constate toujours une légère dégradation des temps CPU au fur et à mesure de l'augmentation du nombre de processus
  - Existence de nombreux accès fichiers simultané pour des écritures de faible taille

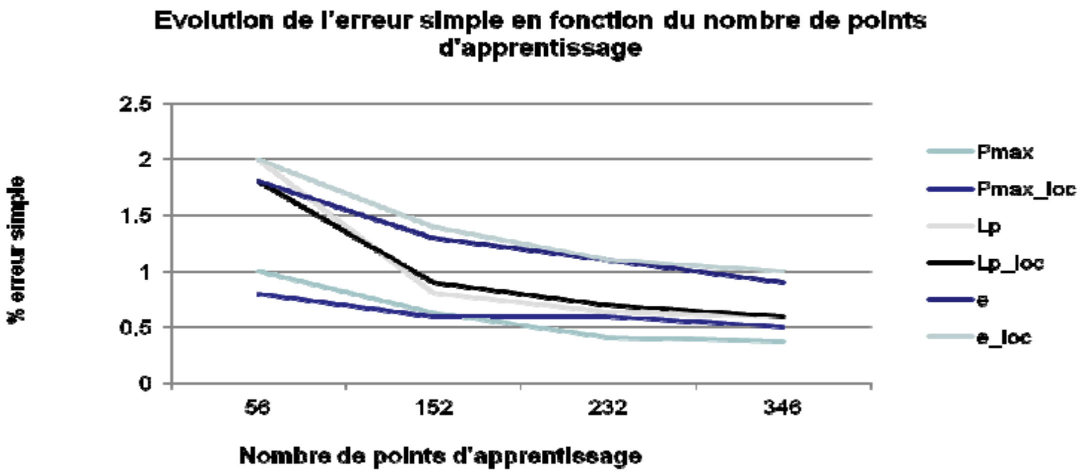


energie atomique • energies alternatives

ROOT . . .
Uranie . . .
Launcher . . .
Applications

Apprentissage et Validation d'un méta-modèle

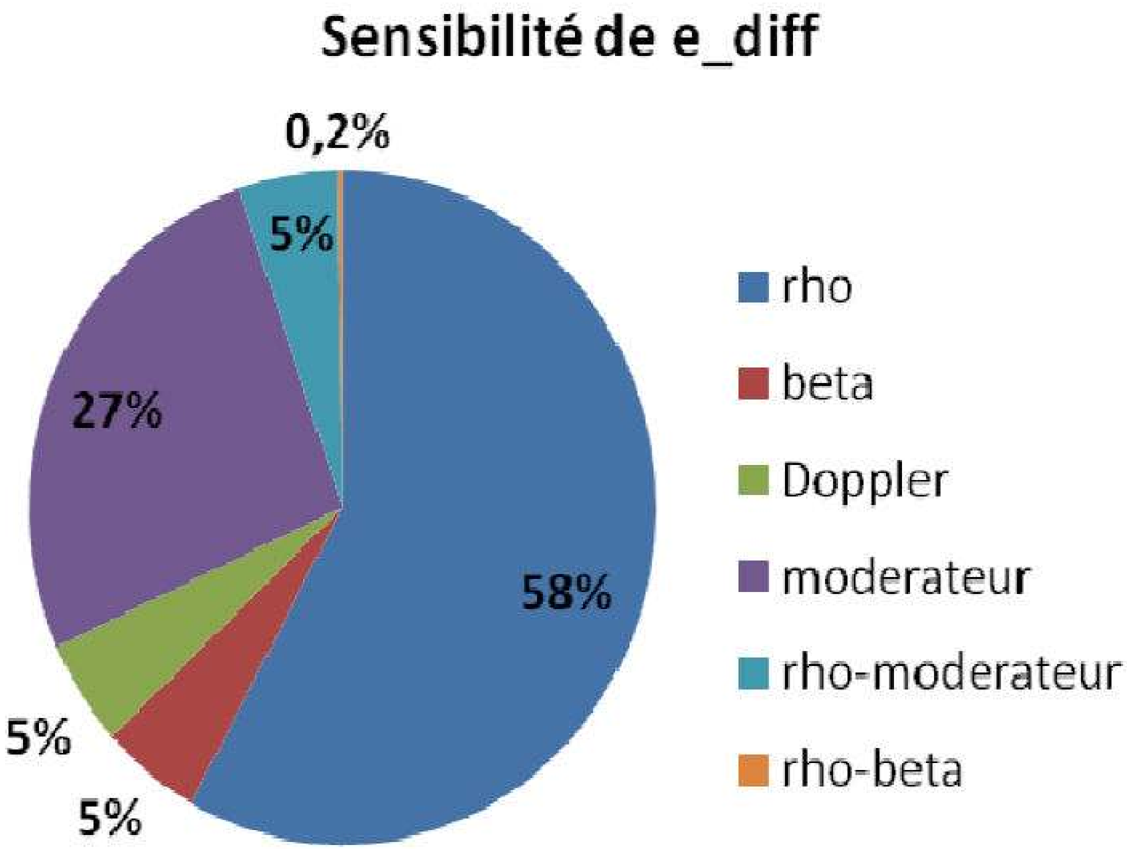




energie atomique • energies alternatives

ROOT ...
Uranie ...
Launcher ...
Applications

Analyse de Sensibilité



Perspectives (sur le cas test)

- Définition de plans d'expérience avec des calculs multiphysiques complets
  - Couplage neutronique/thermohydraulique sur des scénarios RTV (*Rupture Tuyauterie Vapeur*) dans le cadre de la plate-forme SALOME
- Besoins de calculs accrus
  - Calcul unitaire de 48 heures sur 2 processeurs
  - Phénomène moins linéaires → nécessité d'avoir plus de points pour l'apprentissage du méta-modèle

energie atomique • energies alternatives

Conclusions

- Introduction au *"framework"* **ROOT**
- Présentation de la plateforme Incertitude et Optimisation **Uranie**
- Applications d'Uranie :
  - Analyse de Sensibilité à partir de Réseaux de Neurones
  - Réseaux de Neurones sous GPU
- La mise en place de plans d'expérience numérique est un outil d'étude rendu accessible par les puissances de calcul
  - Etudes de sensibilité;
  - Quantification d'incertitudes;
  - Etudes de sureté;
  - Optimisation;
- Son usage au CEA/DEN s'étend:
  - Physique des réacteurs (mécanique, neutronique, thermohydraulique, matériaux);
  - Aval du cycle;
  - Usage chez les parteniares
- Bien que de nature *"embarrasingly parallel"*, ce type de calculs peut réserver des surprises du fait de patterns d'entrées/sorties particuliers.

energie atomique • energies alternatives