



Verrou:

A Tool to Reproduce Floating-Point-Induced Non-Reproducibilities (and help fix them)

Numerical Reproducibility at Exascale
20/06/2019

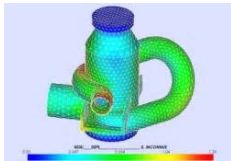
François Févotte*
Bruno Lathuilière

EDF R&D
Analysis and Numerical Modeling

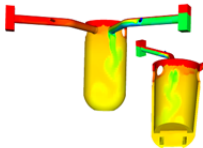


Industrial context – Numerical Verification

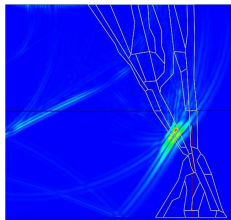
In-house development of Scientific Computing Codes



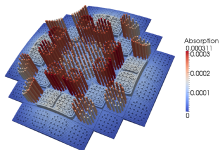
Structures



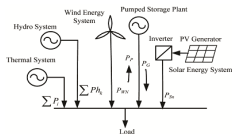
Fluid dynamics



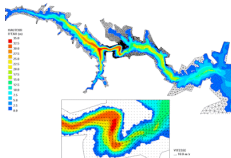
Wave propagation



Neutronics



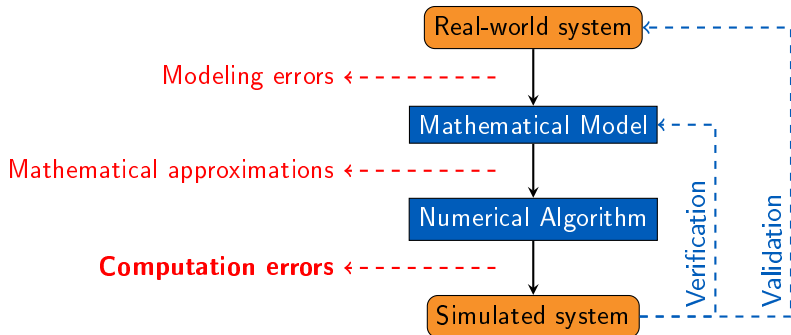
Power Systems



Free surface hydraulics

Industrial context – Numerical Verification

Verification & Validation

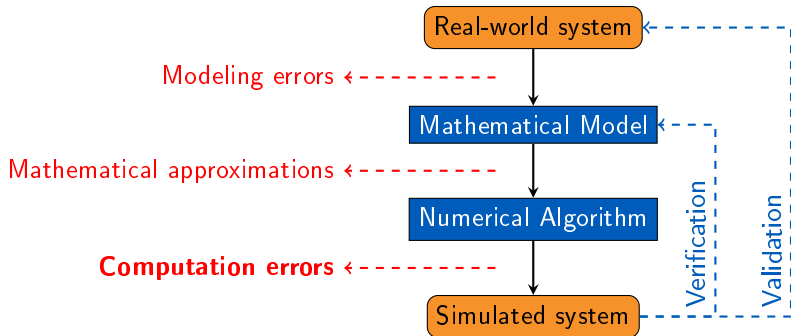


Floating-point arithmetic

- ▶ round-off errors: $a \oplus b \neq a + b$
- ▶ loss of associativity: $(a \oplus b) \oplus c \neq a \oplus (b \oplus c)$
- ▶ reproducibility issues with e.g. parallelization, vectorization, optimization...

Industrial context – Numerical Verification

Verification & Validation



Quantifying numerical errors: at stake

- ◆ quality of produced results (accuracy, reproducibility...)
- ◆ efficient use of resources (development & computing time)

Industrial context – Numerical Verification

Objectives / presentation outline

Diagnostics

- ◆ verify a code / show the presence of FP-related errors
- ◆ quantify the magnitude of issues

Debugging

- ◆ locate the origin of FP-related issues in the source code
 - ▶ unstable algorithms
 - ▶ unstable tests
- ◆ track the origin of issues during program execution
 - ▶ context of calls
 - ▶ temporal information (e.g. iteration number...)

Optimization

- ◆ use mixed-precision implementations



Available on github (latest version: v2.1.0)

<http://github.com/edf-hpc/verrou>

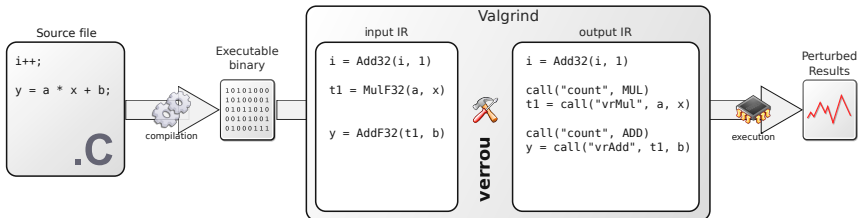
Documentation:

<http://edf-hpc.github.io/verrou/vr-manual.html>

Industrial context – Numerical Verification

Analyse dynamique du binaire avec Valgrind

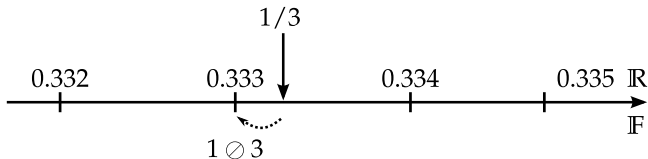
```
$ valgrind --tool=verrou [VERROU_ARGS] PROGRAM [ARGS...]
```



Diagnostics: detect and assess instabilities

Verrou back-end: random rounding

IEEE-754: nearest rounding mode

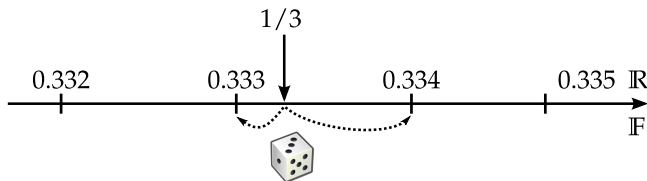


Diagnostics: detect and assess instabilities

Verrou back-end: random rounding

CESTAC/MCA: random rounding mode

--rounding-mode=random



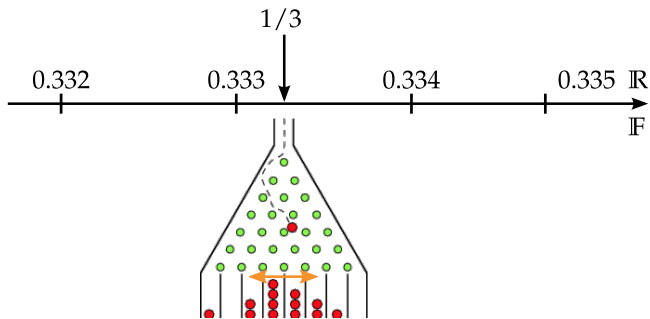
-
- [1] J. Vignes, "A stochastic arithmetic for reliable scientific computation," *Mathematics and Computers in Simulation*, vol. 35, no. 3, 1993.
 - [2] J.-L. Lamotte, J.-M. Chesneaux and F. Jézéquel, "CADNA_C: A version of CADNA for use with C or C++ programs", *Computer Physics Communications*, vol. 181, no. 11, 2010.
 - [3] D. Stott Parker, "Monte Carlo Arithmetic: exploiting randomness in floating-point arithmetic", *Univ. California*, 1997.

Diagnostics: detect and assess instabilities

Verrou back-end: random rounding

CESTAC/MCA: random rounding mode

--rounding-mode=random



- [1] J. Vignes, "A stochastic arithmetic for reliable scientific computation," *Mathematics and Computers in Simulation*, vol. 35, no. 3, 1993.
- [2] J.-L. Lamotte, J.-M. Chesneaux and F. Jézéquel, "CADNA_C: A version of CADNA for use with C or C++ programs", *Computer Physics Communications*, vol. 181, no. 11, 2010.
- [3] D. Stott Parker, "Monte Carlo Arithmetic: exploiting randomness in floating-point arithmetic", *Univ. California*, 1997.

Diagnostics: detect and assess instabilities

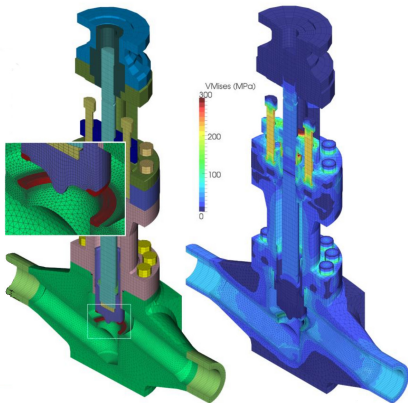
Code_aster

Mechanics

- ◆ Seismic
- ◆ Acoustic
- ◆ Thermo-mechanics

Code_Aster

- ◆ 1.2M code lines
- ◆ Fortran 90, C, Python
- ◆ thousands of test cases
- ◆ Large number of dependencies :
 - ▶ Linear solvers (MUMPS...)
 - ▶ Mesh generator and partitioning tools (Metis, Scotch...)



Goals

- ◆ understand the non-reproducibility between test computers

Diagnostics: detect and assess instabilities

Using Verrou and Random Rounding

Test case	nearest
ssls108i	OK
ssls108j	OK
ssls108k	OK
ssls108l	OK
sdnl112a	OK
ssnp130a	OK
ssnp130b	OK
ssnp130c	OK
ssnp130d	OK

Diagnostics: detect and assess instabilities

Using Verrou and Random Rounding

Test case	nearest	rnd ₁
ssls108i	OK	OK
ssls108j	OK	OK
ssls108k	OK	OK
ssls108l	OK	OK
sdnl112a	OK	KO
ssnp130a	OK	OK
ssnp130b	OK	OK
ssnp130c	OK	OK
ssnp130d	OK	OK

Diagnostics: detect and assess instabilities

Using Verrou and Random Rounding

Test case	nearest	Status		
		rnd ₁	rnd ₂	rnd ₃
ssls108i	OK	OK	OK	OK
ssls108j	OK	OK	OK	OK
ssls108k	OK	OK	OK	OK
ssls108l	OK	OK	OK	OK
sdnl112a	OK	KO	KO	KO
ssnp130a	OK	OK	OK	OK
ssnp130b	OK	OK	OK	OK
ssnp130c	OK	OK	OK	OK
ssnp130d	OK	OK	OK	OK

10 minutes

20 minutes each

(72 test cases)

Diagnostics: detect and assess instabilities

Using Verrou and Random Rounding

Test case	nearest	Status			# common decimal digits $C(\text{rnd}_1, \text{rnd}_2, \text{rnd}_3)$
		rnd_1	rnd_2	rnd_3	
ssls108i	OK	OK	OK	OK	10
ssls108j	OK	OK	OK	OK	10
ssls108k	OK	OK	OK	OK	10
ssls108l	OK	OK	OK	OK	9
sdnl112a	OK	KO	KO	KO	3
ssnp130a	OK	OK	OK	OK	9
ssnp130b	OK	OK	OK	OK	9
ssnp130c	OK	OK	OK	OK	9
ssnp130d	OK	OK	OK	OK	9

10 minutes

(72 test cases)

20 minutes each

$$C(x) = \log_{10} \left| \frac{\mu(x)}{\sigma(x)} \right|$$

Debugging: locate issues in the source code

Using code coverage comparison: code_aster (test-ca:

IEEE-754 coverage

```
120:subroutine fun1(area, a1, a2, n)
-:   implicit none
-:   integer :: n
-:   real(kind=8) :: area, a1, a2
120:   if (a1 .eq. a2) then
13:     area = a1
-:   else
107:     if (n .lt. 2) then
107:       area = (a2-a1) / (log(a2)-log(a1))
###:     else if (n .eq.2) then
###:       area = sqrt (a1*a2)
-:     else
###:       ! ...
-:     endif
-:   endif
120:end subroutine
```


Debugging: locate issues in the source code

Using code coverage comparison: code_aster (test-case sdn1112a)

IEEE-754 coverage

```
120:subroutine fun1(area, a1, a2, n)
-:   implicit none
-:   integer :: n
-:   real(kind=8) :: area, a1, a2
120:   if (a1 .eq. a2) then
13:       area = a1
-:   else
107:       if (n .lt. 2) then
107:           area = (a2-a1) / (log(a2)-log(a1))
###:       else if (n .eq.2) then
###:           area = sqrt (a1*a2)
-:       else
###:           ! ...
-:       endif
-:   endif
120:end subroutine
```

Stochastic Arithmetic coverage

```
120:subroutine fun1(area, a1,...
-:   implicit none
-:   integer :: n
-:   real(kind=8) :: area,...
120:   if (a1 .eq. a2) then
4:       area = a1
-:   else
116:       if (n .lt. 2) then
116:           area = (a2-a1)...
###:       else if (n .eq.2)...
###:           area = sqrt (a...
-:       else
###:           ! ...
-:       endif
-:   endif
120:end subroutine
```

Debugging: locate issues in the source code

Delta-debugging

```
log.L                .../aster.release
volum2_             .../aster.release
bilpla_             .../aster.release
ecrval_             .../aster.release
print_plath_       .../aster.release
classer_groupes_   .../aster.release
etupla_             .../aster.release
couhyd_pi_         .../aster.release
ecrplr_            .../aster.release
imovi_             .../aster.release
resopt_            .../aster.release
getgrp_marginal_   .../aster.release
ecrpla_            .../aster.release
fin_exec_main_     .../aster.release
decopt_pi_         .../aster.release
paraend_           .../aster.release
resopt_cnt_zones_ .../aster.release
apstop_            .../aster.release
ihyd_              .../aster.release
impression_info_   .../aster.release
coupla_            .../aster.release
gere_print_plath_ .../aster.release
log                .../aster.release
thepla_            .../aster.release
coutot_            .../aster.release
iprit_             .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou



Debugging: locate issues in the source code

Delta-debugging

```
# log.L                .../aster.release
# volum2_              .../aster.release
# bilpla_              .../aster.release
# ecrval_              .../aster.release
# print_plath_         .../aster.release
# classer_groupes_    .../aster.release
# etupla_              .../aster.release
# couhyd_pi_           .../aster.release
# ecrplr_              .../aster.release
# imovi_               .../aster.release
# resopt_              .../aster.release
# getgrp_marginal_    .../aster.release
# ecrpla_              .../aster.release
# fin_exec_main_      .../aster.release
# decopt_pi_          .../aster.release
# paraend_             .../aster.release
# resopt_cnt_zones_   .../aster.release
# apstop_              .../aster.release
# ihyd_                .../aster.release
# impression_info_    .../aster.release
# coupla_              .../aster.release
# gere_print_plath_   .../aster.release
# log                  .../aster.release
# thepla_              .../aster.release
# coutot_              .../aster.release
# iprit_               .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou



Debugging: locate issues in the source code

Delta-debugging

```
# log.L                .../aster.release
# volum2_             .../aster.release
# bilpla_             .../aster.release
# ecrval_             .../aster.release
# print_plath_       .../aster.release
# classer_groupes_   .../aster.release
# etupla_            .../aster.release
# couhyd_pi_         .../aster.release
# ecrplr_            .../aster.release
# imovi_             .../aster.release
# resopt_            .../aster.release
# getgrp_marginal_   .../aster.release
# ecrpla_            .../aster.release
fin_exec_main_       .../aster.release
decopt_pi_          .../aster.release
paraend_            .../aster.release
resopt_cnt_zones_   .../aster.release
apstop_            .../aster.release
ihyd_              .../aster.release
impression_info_    .../aster.release
coupla_            .../aster.release
gere_print_plath_   .../aster.release
log                 .../aster.release
thepla_            .../aster.release
coutot_            .../aster.release
iprit_             .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou



Debugging: locate issues in the source code

Delta-debugging

```
# log.L                .../aster.release
# volum2_             .../aster.release
# bilpla_             .../aster.release
# ecrval_             .../aster.release
# print_plath_       .../aster.release
# classer_groupes_   .../aster.release
# etupla_            .../aster.release
couhyd_pi_           .../aster.release
ecrplr_              .../aster.release
imovi_              .../aster.release
resopt_             .../aster.release
getgrp_marginal_    .../aster.release
ecrpla_             .../aster.release
fin_exec_main_      .../aster.release
decopt_pi_          .../aster.release
paraend_            .../aster.release
resopt_cnt_zones_   .../aster.release
apstop_             .../aster.release
ihyd_               .../aster.release
impression_info_    .../aster.release
coupla_             .../aster.release
gere_print_plath_   .../aster.release
log                 .../aster.release
thepla_             .../aster.release
coutot_             .../aster.release
iprit_              .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou



Debugging: locate issues in the source code

Delta-debugging

```
log.L                .../aster.release
volum2_              .../aster.release
bilpla_              .../aster.release
ecrval_              .../aster.release
print_plath_         .../aster.release
classer_groupes_    .../aster.release
etupla_              .../aster.release
# couhyd_pi_         .../aster.release
# ecrplr_            .../aster.release
# imovi_             .../aster.release
# resopt_            .../aster.release
# getgrp_marginal_   .../aster.release
# ecrpla_            .../aster.release
fin_exec_main_       .../aster.release
decopt_pi_           .../aster.release
paraend_             .../aster.release
resopt_cnt_zones_   .../aster.release
apstop_              .../aster.release
ihyd_                .../aster.release
impression_info_     .../aster.release
coupla_              .../aster.release
gere_print_plath_    .../aster.release
log                  .../aster.release
thepla_              .../aster.release
coutot_              .../aster.release
iprit_               .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou



Debugging: locate issues in the source code

Delta-debugging

```
log.L                .../aster.release
volum2_             .../aster.release
bilpla_            .../aster.release
ecrval_            .../aster.release
print_plath_       .../aster.release
classer_groupes_  .../aster.release
etupla_            .../aster.release
# couhyd_pi_       .../aster.release
ecrplr_            .../aster.release
imovi_             .../aster.release
resopt_            .../aster.release
getgrp_marginal_  .../aster.release
ecrpla_            .../aster.release
fin_exec_main_    .../aster.release
# decopt_pi_       .../aster.release
paraend_           .../aster.release
resopt_cnt_zones_ .../aster.release
apstop_           .../aster.release
# ihyd_            .../aster.release
impression_info_  .../aster.release
coupla_           .../aster.release
gere_print_plath_ .../aster.release
log                .../aster.release
thepla_           .../aster.release
# coutot_         .../aster.release
# iprit_          .../aster.release
```

- ◆ Delta-Debugging [A. Zeller, 1999]
- ◆ relies on ability to restrict the scope of instrumentation/perturbations induced by Verrou
- ◆ Inputs :
 - ▶ run script
 - ▶ comparison script
- ◆ Output:
 - ▶ “unstable” code parts
- ◆ Also works at the source line granularity:
 - ▶ if the code was compiled with `-g`

Optimization for mixed precision

Verrou back-end: precision reduction

- ◆ All high-precision values are cast to single precision before performing arithmetic operations
 - ▶ emulates the use of low-precision arithmetic
 - ▶ ... but not the performance boost.
- ◆ Currently implemented only for double→single (`--rounding-mode=float`)
 - ▶ can be generalized to all precision reductions (e.g single → half)
 - ▶ can not emulate higher precision (memory problem)
- ◆ Can be combined with Delta-Debugging
 - ▶ find which code parts can be downgraded to lower precision

[1] A. Dawson and P. D. Düben, “rpe v5: an emulator for reduced floating-point precision in large numerical simulations”, *Geoscientific Model Development*, 2017.

Wrap-up

Verrou as a tool to help with FP issues

Diagnostics

- ☀ show the presence of FP-related errors (random-rounding back-end)
- 🌱 quantify the magnitude of issues (post-processing)

Debugging

- 🌱 locate the origin of FP-related issues in the source code
 - ☀ unstable algorithms (Delta-Debugging)
 - 🌱 unstable tests (code coverage analysis)
- ☁ track the origin of issues during program execution
 - ▶ context of calls
 - ▶ temporal information (e.g. iteration number...)

Optimization

- ☁ emulate mixed-precision implementations (reduced precision back-end)
- ▶ re-use debugging features (Delta-Debugging)

Outlooks

Interflop

- ◆ Verrou is no silver bullet
 - ▶ multiply techniques & tools

Interflop (toolbox)

Common interface for Verificarlo & Verrou (and anyone interested!)

- ◆ share Stochastic Arithmetic back-ends
- ◆ share accompanying tools (Delta-Debugging...)
- ◆ improve performance of instrumentation front-ends

Interflop (consortium)

- ◆ Explore different analysis methods and the links between them
 - ▶ Stochastic Arithmetic
 - ▶ Interval Arithmetic & Affine Forms







Thank you !
Questions ?





Get Verrou on github:
<http://github.com/edf-hpc/verrou>

Documentation:
<http://edf-hpc.github.io/verrou/vr-manual.html>






Relevant references I

-  Jean-Marie Chesneaux and Jean Vignes, *On the robustness of the cestac method*, C. R. Acad.Sci. Paris **1** (1988), 855–860.
-  A. Dawson and P. D. Düben, *rpe v5: an emulator for reduced floating-point precision in large numerical simulations*, Geoscientific Model Development **10** (2017), no. 6, 2221–2230.
-  Christophe Denis, Pablo de Oliveira Castro, and Eric Petit, *Verificarlo: checking floating point accuracy through Monte Carlo Arithmetic*, 23rd IEEE Internatinal Symposium on Computer Arithmetic (ARITH'23), 2016.
-  François Févotte and Bruno Lathuilière, *VERROU: Assessing Floating-Point Accuracy Without Recompiling*, <https://hal.archives-ouvertes.fr/hal-01383417>, October 2016.

Relevant references II

-  François Févotte and Bruno Lathuilière, *Studying the numerical quality of an industrial computing code: A case study on code_aster*, 10th International Workshop on Numerical Software Verification (NSV) (Heidelberg, Germany), July 2017, pp. 61–80.
-  Stef Graillat, Fabienne Jézéquel, and Romain Picot, *Numerical validation of compensated algorithms with stochastic arithmetic*, Applied Mathematics and Computation **329** (2018), 339 – 363.
-  Fabienne Jézéquel, Jean-Marie Chesneaux, and Jean-Luc Lamotte, *A new version of the CADNA library for estimating round-off error propagation in Fortran programs*, Computer Physics Communications **181** (2010), no. 11, 1927–1928.
-  William Kahan, *How futile are mindless assessments of roundoff in floating-point computations?*, <https://people.eecs.berkeley.edu/~wkahan/Mindless.pdf>, 2006.

Relevant references III

-  Jean-Luc Lamotte, Jean-Marie Chesneaux, and Fabienne Jézéquel, *CADNA_C: A version of CADNA for use with C or C++ programs*, Computer Physics Communications **181** (2010), no. 11, 1925–1926.
-  Devan Sohier, Pablo De Oliveira Castro, François Févotte, Bruno Lathuilière, Eric Petit, and Olivier Jamond, *Confidence Intervals for Stochastic Arithmetic*, preprint, <https://hal.archives-ouvertes.fr/hal-01827319>.
-  Douglas Stott Parker, *Monte Carlo arithmetic: exploiting randomness in floating-point arithmetic*, Tech. Report CSD-970002, University of California, Los Angeles, 1997.
-  Jean Vignes, *A stochastic arithmetic for reliable scientific computation*, Mathematics and Computers in Simulation **35** (1993), 233–261.
-  Andreas Zeller, *Yesterday, My Program Worked. Today, It Does Not. Why?*, SIGSOFT Softw. Eng. Notes **24** (1999), no. 6, 253–267.

Annexes



Additional slides



Case study on ACTS: emulation of reduced precision + NaN detection

```
$ valgrind --tool=verrou --rounding-mode=float --exclude=excludes.ex --demangle=no --trace-children=yes --num-callers=50 \  
> ctest -V -R "BFieldMapUtils"
```

[...]

==1863== NaN:

```
==1863== at 0x5847E7F: _ZN5Eigen8internal4pmulIDv2_dEET_RKS3_S5_(emmintrin.h:271)  
==1863== by 0x585B570: _ZNK5Eigen8internal17scalar_product_opIddE8packetOpIDv2_dEET_RS6_S7_(BinaryFuncctors.h:89)  
==1863== by 0x59A31C9: _ZNK5Eigen8internal16binary_evaluatorINS_13CwiseBinaryOpINSO_17scalar_product_opIddE8KNS_14CwiseNullaryOpINSO_18scalar_constant_opIddE8KNS_6MatrixIdLi2ELi1ELi0ELi2ELi1EEEEESA_EENS0_10IndexBasedESE_ddE6packetILi16EDv2_dEET0_11(CoreEvaluators.h:727)
```

[...]

```
==1863== by 0x58DE2D4: _ZN5Eigen6MatrixIdLi2ELi1ELi0ELi2ELi1EEaSINS_13CwiseBinaryOpINS_8internal13scalar_sum_opIddE8KNS3_INS4_17scalar_product_opIddE8KNS_14CwiseNullaryOpINS4_18scalar_constant_opIddE8KNS1_EESC_EESG_EEERS1_RKNS_9DenseBaseIT_EE(Matrix.h:225)  
==1863== by 0x58DAD82: _ZN4Acts6detail16interpolate_implIN5Eigen6MatrixIdLi2ELi1ELi0ELi2ELi1EEEE4_St5arrayIdLm2EES6_Lm1ELm4EE3runERKS4_RKS6_SB_RKS5_IS4_Lm4EE(interpolation_impl.hpp:133)  
==1863== by 0x58D59E8: _ZN4Acts11interpolateIN5Eigen6MatrixIdLi2ELi1ELi0ELi2ELi1EEELm4ES3_St5arrayIdLm2EES5_vEET_RKT1_RKT2_RKT3_RKS4_IS6_XTO_EE(Interpolation.hpp:95)
```

[...]

```
==1863== by 0x4F6D54: _ZNK4Acts21interpolatedBFieldMap1FieldMapperILj2ELj2EE8getFieldERKN5Eigen6MatrixIdLi3ELi1ELi0ELi3ELi1EEEE(InterpolatedBFieldMap.hpp:166)  
==1863== by 0x477030: _ZN4Acts4Test15bfield_creation1itest_methodEv(BFieldMapUtilsTests.cpp:88)  
==1863== by 0x474767: _ZN4Acts4TestL23bfield_creation_invokerEv(BFieldMapUtilsTests.cpp:25)  
==1863== by 0x549B9B: _ZN5boost6detail8function22void_function_invokerOIPFvvEvE6invokeERNs1_15function_bufferE(function_template.hpp:118)
```

[...]

```
==1863== by 0x43F2FE: _ZN5boost9unit_test9framework3runEmb(framework.hpp:1629)  
==1863== by 0x463F10: _ZN5boost9unit_test14unit_test_mainEPPNSO_10test_suiteEiPPCeiS4_(unit_test_main.hpp:247)  
==1863== by 0x4648CB: main(unit_test_main.hpp:303)
```

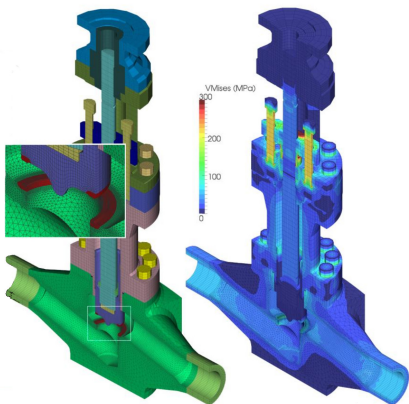
[...]

Running 12 test cases...

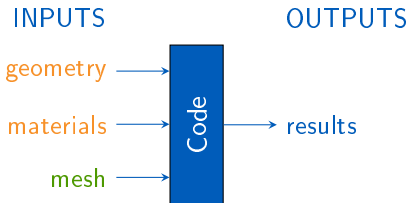
```
acts-core/Tests/Utilities/BFieldMapUtilsTests.cpp(105): error: in "bfield_creation": difference{-nan} between  
value2_rz.perp(){-nan} and bfield2_rz.perp(){} exceeds 1e-09%
```


Additional slides

V&V process: ad-hoc numerical instability detection methods



Physical input: affects the result
Simulation parameter: should be neutral



▶ Idea: measure the sensitivity of the results w.r.t “neutral” parameters



easy to do



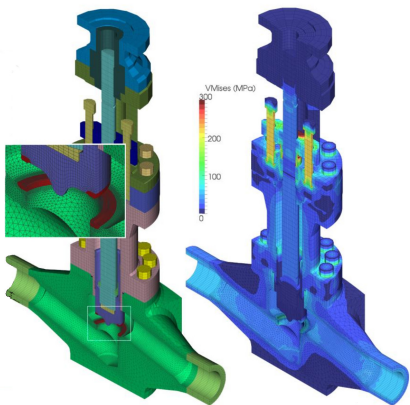
ad hoc, no localization

Additional slides

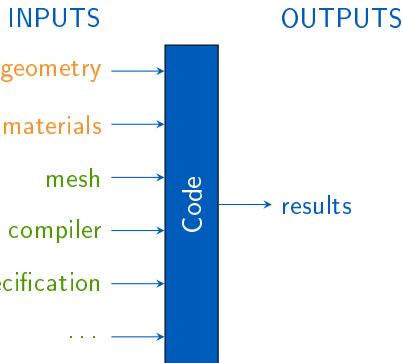


V&V process: ad-hoc numerical instability detection methods

Physical input: affects the result
Simulation parameter: should be neutral

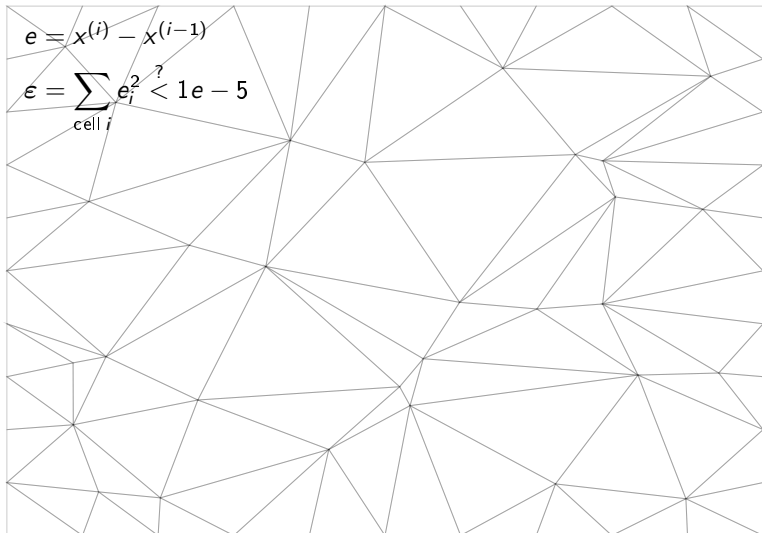


hardware specification



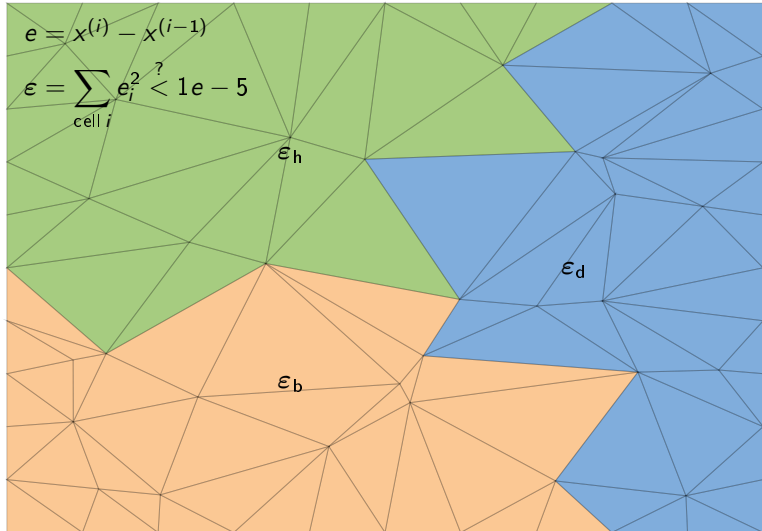
Additional slides

Example 1: failing computation in free surface hydraulics (Telemac 2D)



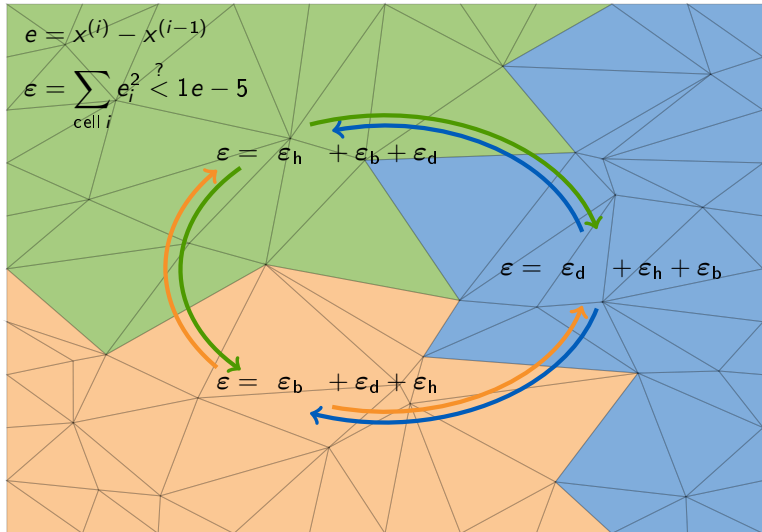
Additional slides

Example 1: failing computation in free surface hydraulics (Telemac 2D)



Additional slides

Example 1: failing computation in free surface hydraulics (Telemac 2D)



Additional slides

Example 2: performances of an optimization problem



Initial problem formulation

$$\max_{p,v} \sum_{t \in T} \sum_{i \in I} \lambda_t p_t^i + \sum_{r \in R} \omega_r \left(v_t^r - V_0^r - \sum_{t \in T} \Gamma_t^r \right)$$

Objective functional computation

$$\sum_{t \in T} \sum_{i \in I} \lambda_t p_t^i + \sum_{r \in R} \omega_r v_t^r \quad 1534019.677371745$$

$$- \sum_{r \in R} \omega_r \left(V_0^r + \sum_{t \in T} \Gamma_t^r \right) \quad -1534019.677282780$$

0.00008896500000000000

11 digits

Additional slides

Example 2: performances of an optimization problem



Proposed re-formulation

$$\left[\max_{p,v} \sum_{t \in T} \sum_{i \in I} \lambda_t p_t^i + \sum_{r \in R} \omega_r v_t^r \right] - \sum_{r \in R} \omega_r \left(v_0^r + \sum_{t \in T} \Gamma_t^r \right)$$

Objective functional computation

$$\sum_{t \in T} \sum_{i \in I} \lambda_t p_t^i + \sum_{r \in R} \omega_r v_t^r \quad 1534019.677371745$$

$$- \sum_{r \in R} \omega_r \left(v_0^r + \sum_{t \in T} \Gamma_t^r \right) \quad -1534019.677282780$$

0.00008896500000000000

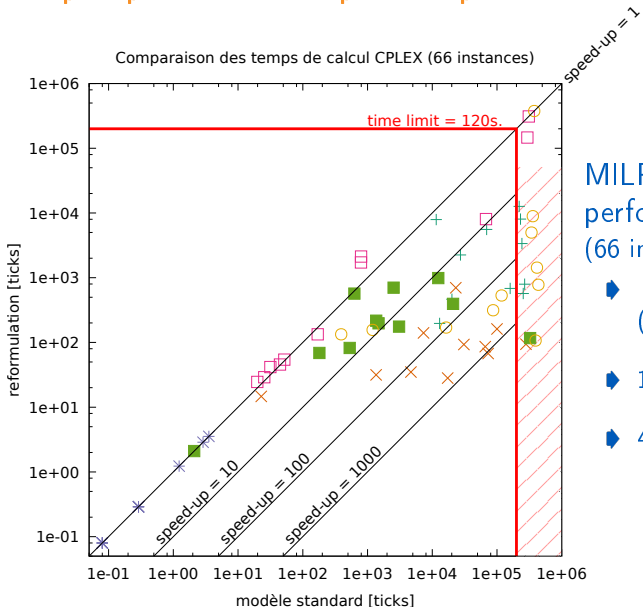
11 digits

Additional slides

Example 2: performances of an optimization problem



Comparaison des temps de calcul CPLEX (66 instances)



MILP resolution performances (66 instances)

- ◆ 2 (3%) slowed down ($\times 2$)
- ◆ 15 (23%) unchanged
- ◆ 49 (74%) sped up ($\times 2-1100$)