Cas d'études de calculs parallèles numériquement reproductibles

Philippe Langlois, Chemseddine Chohra, Rafife Nheili

Retour d'expéRiences sur la Recherche Reproductible
3-4 décembre 2015, Orléans

Cas d'études de calculs parallèles numériquement reproductibles

Philippe Langlois, Chemseddine Chohra, Rafife Nheili

DALI, Université de Perpignan Via Domitia
LIRMM, UMR 5506 CNRS - Université de Montpellier

DALI, Digits, Architectures
et Logiciels Informatiques

Acknowledgment

Christophe Denis, EDF R&D, Clamart and CMLA, Cachan
Jean-Michel Hervouet, LNE, EDR R&D, Chatou

DALI, Digits, Architectures et Logiciels Informatiques

Reproducibility failure of one industrial simulation code

- Simulation of free-surface flows in 1D-2D-3D hydrodynamic
- Integrated set of open source Fortran 90 modules, 300 000 loc.
- LNHE (EDF R&D) + international consortium, 20 years, 4000 reg. users

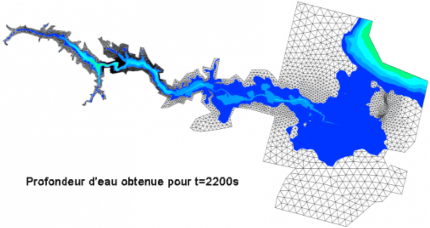
Telemac 2D [5]

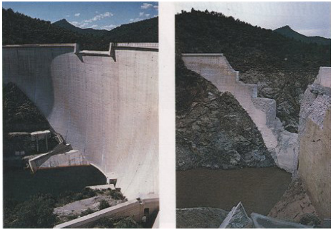
- 2D hydrodynamic: Saint Venant equations
- Finite element method, triangular element mesh, sub-domain decomposition for parallel resolution
- Mesh node unknowns: water depth (H) and velocity (U,V)

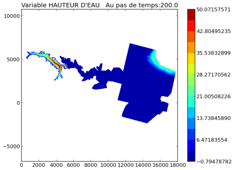
The Malpasset dam break: a reproducible simulation?

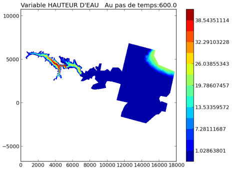
The Malpasset dam break (1959)

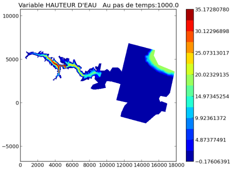
- A five year old dam break: 433 dead people and huge damage
- Simulation mesh: 26000 elements and 53000 nodes
- Simulation: 2200 seconds with a 2 sec. time step







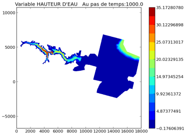




The Malpasset dam break: a reproducible simulation?

The Malpasset dam break (1959)

- A five year old dam break: 433 dead people and huge damage
- Simulation mesh: 26000 elements and 53000 nodes
- Simulation: 2200 seconds with a 2 sec. time step



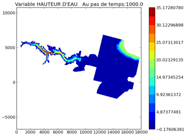
A reproducible simulation?

	velocity U	velocity V	depth H
The sequential run	0.4029747E-02	0.7570773E-02	0.3500122E-01

The Malpasset dam break: a reproducible simulation?

The Malpasset dam break (1959)

- A five year old dam break: 433 dead people and huge damage
- Simulation mesh: 26000 elements and 53000 nodes
- Simulation: 2200 seconds with a 2 sec. time step



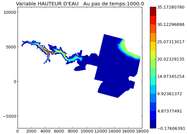
A reproducible simulation?

	velocity U	velocity V	depth H
The sequential run	0.4029747E-02	0.7570773E-02	0.3500122E-01
one 64 procs run	0.4935279E-02	0.3422730E-02	0.2748817E-01

The Malpasset dam break: a reproducible simulation?

The Malpasset dam break (1959)

- A five year old dam break: 433 dead people and huge damage
- Simulation mesh: 26000 elements and 53000 nodes
- Simulation: 2200 seconds with a 2 sec. time step



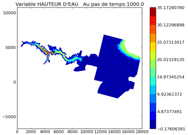
A reproducible simulation?

	velocity U	velocity V	depth H
The sequential run	0.4029747E-02	0.7570773E-02	0.3500122E-01
one 64 procs run	0.4935279E-02	0.3422730E-02	0.2748817E-01
one 128 procs run	0.4512116E-02	0.7545233E-02	0.1327634E-01

The Malpasset dam break: a reproducible simulation?

The Malpasset dam break (1959)

- A five year old dam break: 433 dead people and huge damage
- Simulation mesh: 26000 elements and 53000 nodes
- Simulation: 2200 seconds with a 2 sec. time step



A reproducible simulation?

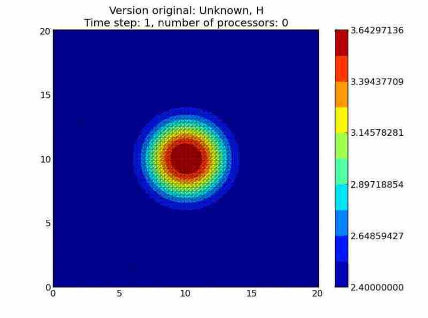
	velocity U	velocity V	depth H
The sequential run	0.4029747E-02	0.7570773E-02	0.3500122E-01
one 64 procs run	0.4935279E-02	0.3422730E-02	0.2748817E-01
one 128 procs run	0.4512116E-02	0.7545233E-02	0.1327634E-01

The privileged sequential run? uncertainty: up to $\times$ 2.5

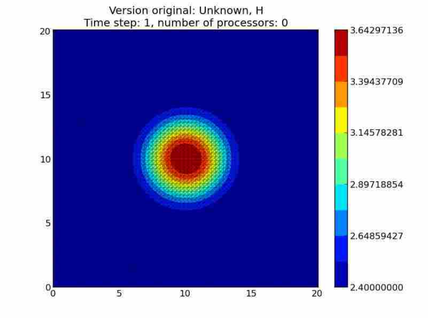
Telemac2D: the simplest gouttedo simulation

Expected numerical reproducibility

time step $= 1, 2, \ldots$



Sequential

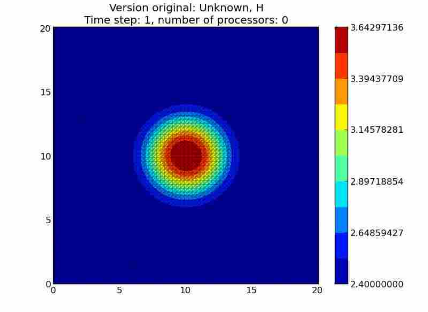


Parallel $p = 2$

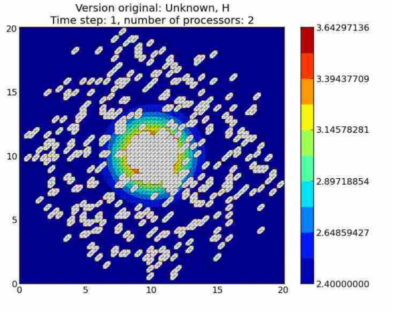
Telemac2D: gouttedo

Numerical reproducibility?

time step $= 1$



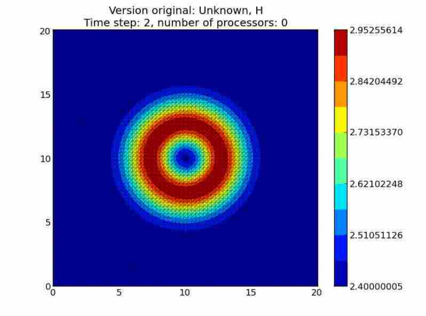
Sequential



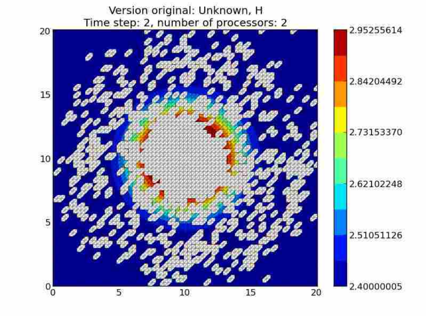
Parallel $p = 2$

Numerical reproducibility?

time step = 2



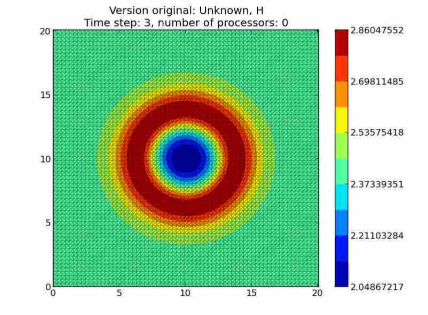
Sequential



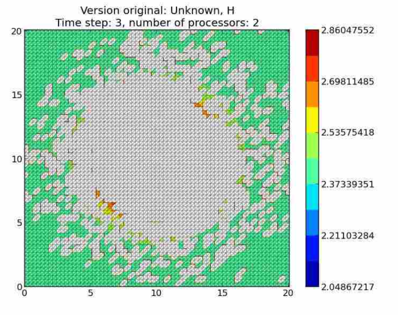
Parallel $p = 2$

Numerical reproducibility?

time step = 3



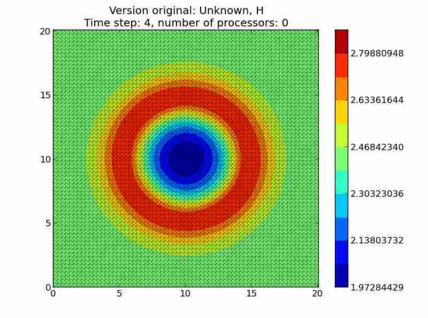
Sequential



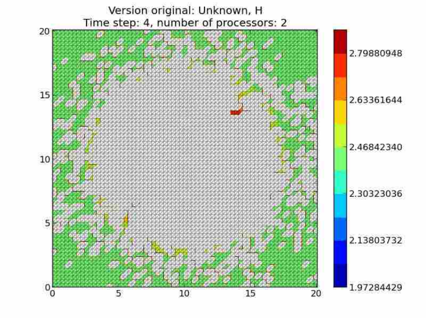
Parallel $p = 2$

Numerical reproducibility?

time step = 4



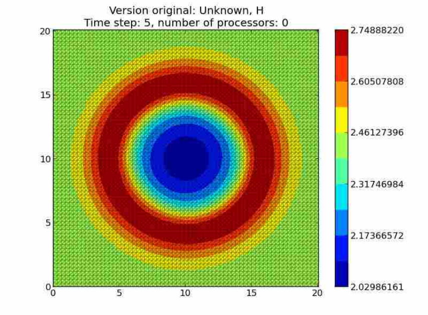
Sequential



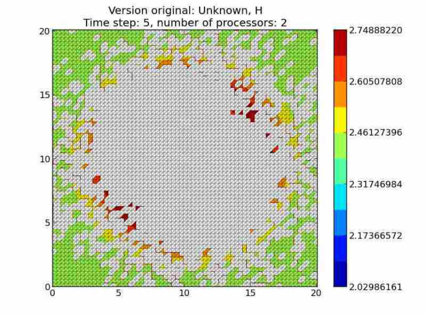
Parallel $p = 2$

Numerical reproducibility?

time step $= 5$



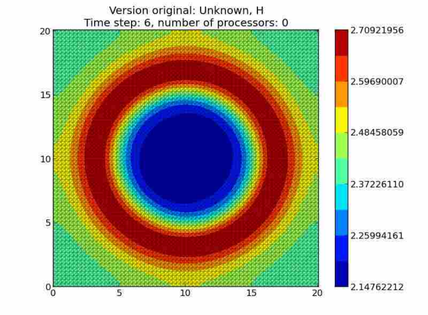
Sequential



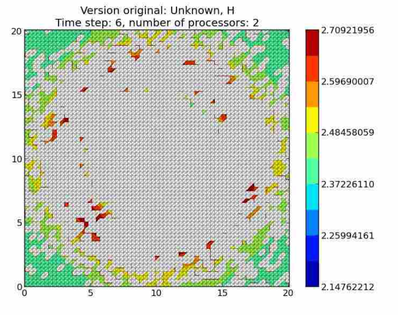
Parallel $p = 2$

Numerical reproducibility?

time step $= 6$



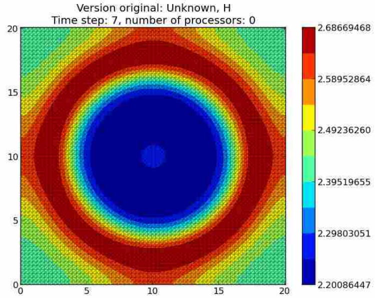
Sequential



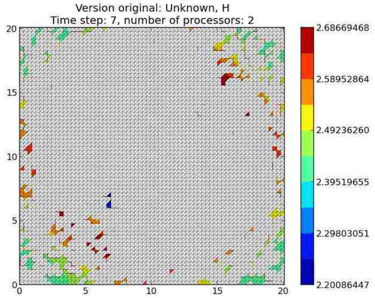
Parallel $p = 2$

Numerical reproducibility?

time step = 7



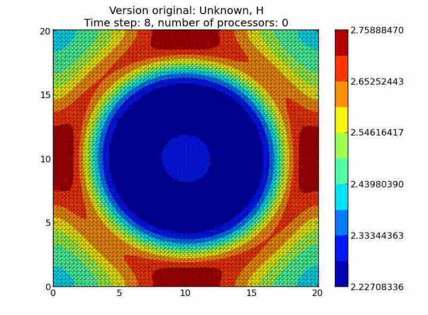
Sequential



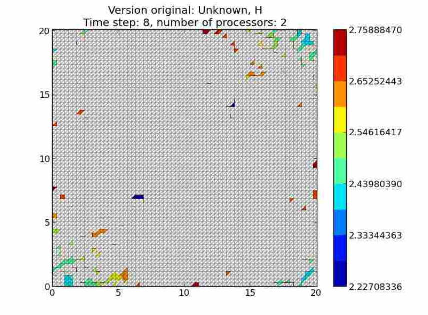
Parallel $p = 2$

Numerical reproducibility?

time step = 8



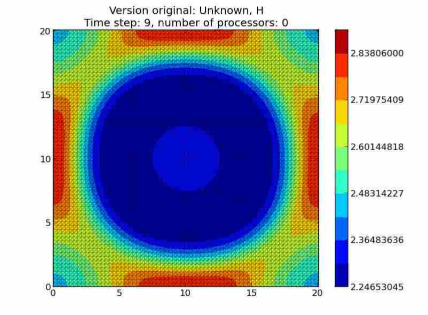
Sequential



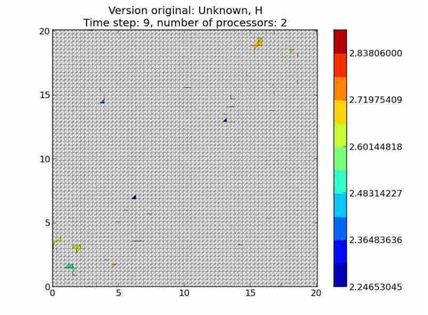
Parallel $p = 2$

Numerical reproducibility?

time step = 9



Sequential

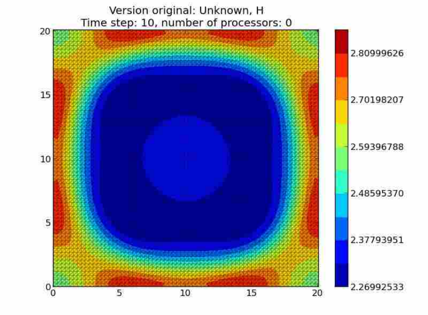


Parallel $p = 2$

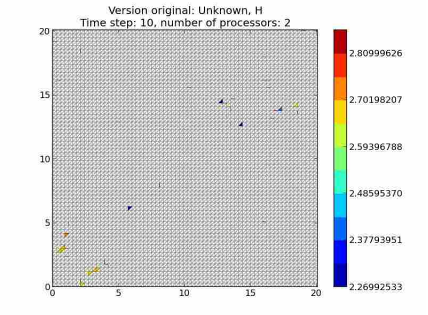
Telemac2D: gouttedo

Numerical reproducibility?

time step = 10



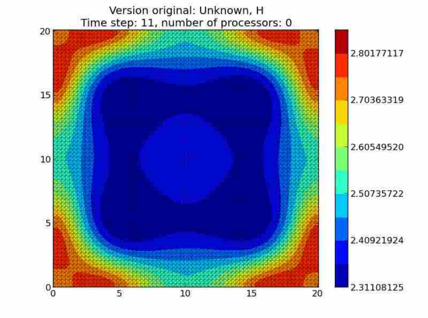
Sequential



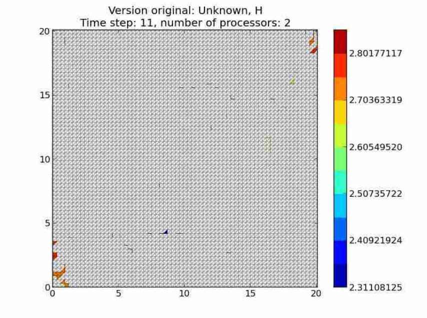
Parallel $p = 2$

Numerical reproducibility?

time step = 11



Sequential

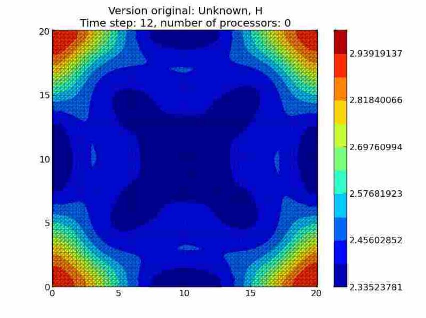


Parallel $p = 2$

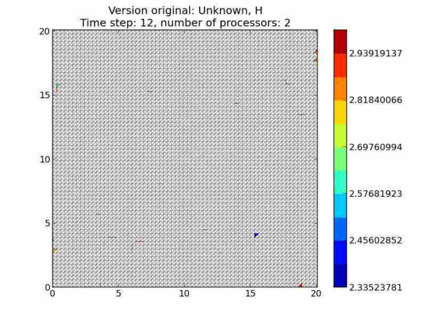
Telemac2D: gouttedo

Numerical reproducibility?

time step $= 12$



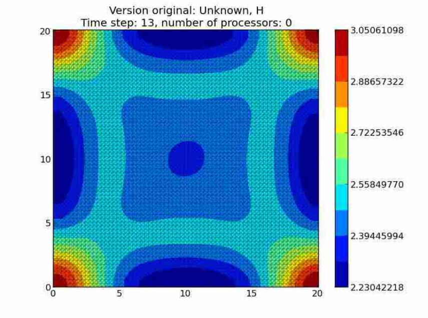
Sequential



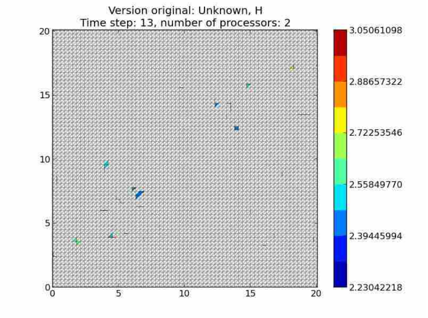
Parallel $p = 2$

Numerical reproducibility?

time step $= 13$



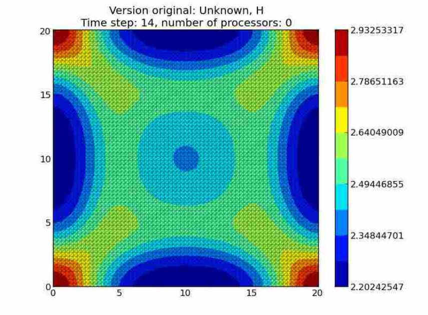
Sequential



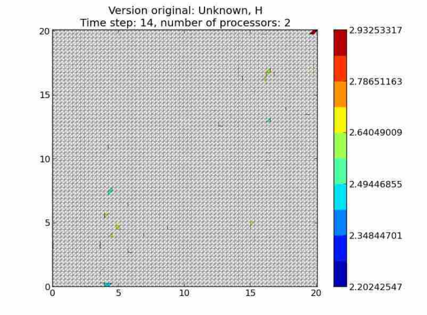
Parallel $p = 2$

Numerical reproducibility?

time step = 14



Sequential

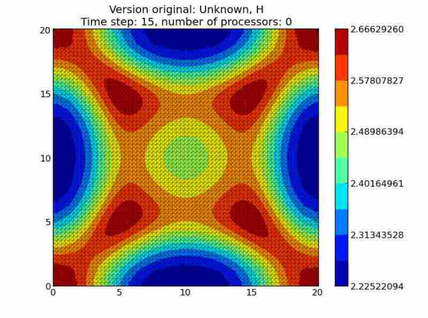


Parallel $p = 2$

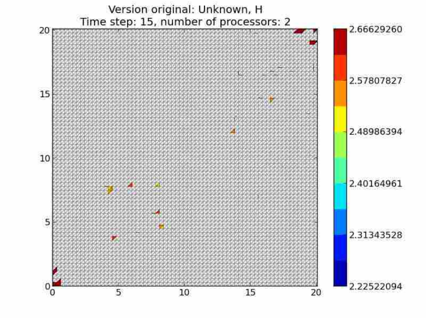
Telemac2D: gouttedo

NO numerical reproducibility!

time step = 15

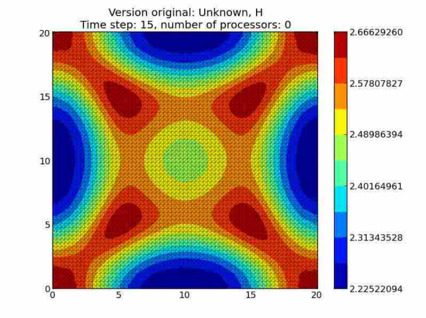


Sequential

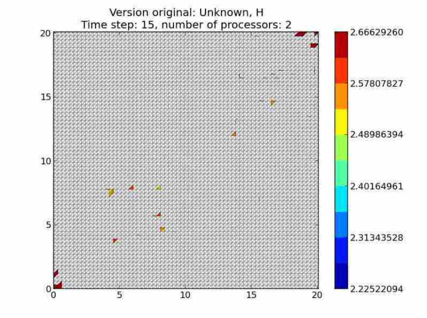


Parallel $p = 2$

NO numerical reproducibility!

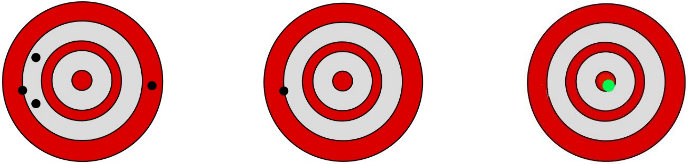


Sequential



Parallel $p = 2$

Numerical Reproducibility



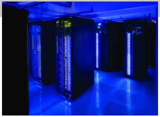
1. Motivations
2. Basic ingredients
3. Recovering reproducibility in a finite element resolution
4. Efficient and reproducible BLAS 1
5. Conclusion and work in progress

Motivations

Exascale HPC and numerical simulation

- Moore's rule $\rightarrow 10^{\mathbf{18}}$ flop/sec in 2020
- Massive and heterogeneous parallelism : 1 million of computing units
- Numerical simulation of complex and sensitive physical phenomena

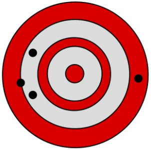
Motivations



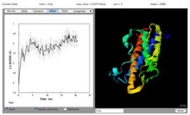
Exascale HPC and numerical simulation

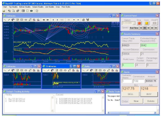
- Moore's rule $\rightarrow 10^{18}$ flop/sec in 2020
- Massive and heterogeneous parallelism : 1 million of computing units
- Numerical simulation of complex and sensitive physical phenomena

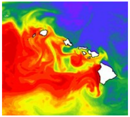
Numerical reproductibility failure of finite precision computations



- Non associative floating-point addition
- Computed value depends on the operation order
- Reproducibility failures reported in numerical simulations for energy [12], dynamical weather science [4], dynamical molecular [11], dynamical fluid [8]







Reproducibility failure (1)

When? why?

- Operation order uncertainty for consecutive executions of a given binary file
- Appears both in parallel and "sequential+SIMD" environments

Reproducibility failure (1)

When? why?

- Operation order uncertainty for consecutive executions of a given binary file
- Appears both in parallel and "sequential+SIMD" environments

Parallel reduction: SIMD, openMP, MPI, GPU

- Let p be the number of computing units
- When p varies, the partial computed values before the reduction also vary
- For a given $p > 2$, the computed reduced value depends on the dynamic scheduling of the reduction: omp, mpi, gpu

Reproducibility failure (2)

Reproducibility $\neq$ Portability

- Portability : one source $\rightarrow$ different binaries
- Parameters: compilers and their options, librairies, OS, comput. units
- Reproducibility may fail for a given set of portability parameters

Reproducibility failure (2)

Reproducibility $\neq$ Portability

- Portability : one source $\rightarrow$ different binaries
- Parameters: compilers and their options, librairies, OS, comput. units
- Reproducibility may fail for a given set of portability parameters

Reproducibility $\neq$ Accuracy

- Reproducibility: bitwise identical results for every p-parallel run, $p \geq 1$
- Full accuracy = unit roundoff accuracy = bitwise exact result
- Improving accuracy up to correct rounding $\Rightarrow$ reproducibility

Reproducibility failure (2)

Reproducibility $\neq$ Portability

- Portability : one source $\rightarrow$ different binaries
- Parameters: compilers and their options, librairies, OS, comput. units
- Reproducibility may fail for a given set of portability parameters

Reproducibility $\neq$ Accuracy

- Reproducibility: bitwise identical results for every p-parallel run, $p \geq 1$
- Full accuracy = unit roundoff accuracy = bitwise exact result
- Improving accuracy up to correct rounding $\Rightarrow$ reproducibility

Reproducibility

- Pros: numerical debug, validation, legal agreements
- Cons: numerical debug, stochastic arithmetic

Today's issues

Feasibility

- Do existing techniques easily provide reproducibility to large scale (industrial) scientific software?

Efficiency

- Do correctly rounded summation algorithms provide efficient implementations of reproducible parallel BLAS routines?

Today's issues

Feasibility

- Do existing techniques easily provide reproducibility to large scale (industrial) scientific software?
  - openTelemac-Mascaret: Tomawak, Telemac2D
  - finite element assembly, domain decomposition, linear system solving

Efficiency

- Do correctly rounded summation algorithms provide efficient implementations of reproducible parallel BLAS routines?

Today's issues

Feasibility

- Do existing techniques easily provide reproducibility to large scale (industrial) scientific software?
  - openTelemac-Mascaret: Tomawak, Telemac2D
  - finite element assembly, domain decomposition, linear system solving

Efficiency

- Do correctly rounded summation algorithms provide efficient implementations of reproducible parallel BLAS routines?
  - BLAS 1: asum, dot, nrm2
  - overcost vs. Intel MKL

Today's issues

Feasibility

- Do existing techniques easily provide reproducibility to large scale (industrial) scientific software?
  - openTelemac-Mascaret: Tomawak, Telemac2D
  - finite element assembly, domain decomposition, linear system solving
- Compensation yields reproducibility here!

Efficiency

- Do correctly rounded summation algorithms provide efficient implementations of reproducible parallel BLAS routines?
  - BLAS 1: asum, dot, nrm2
  - overcost vs. Intel MKL
- Convincing rtn-BLAS 1 but ...

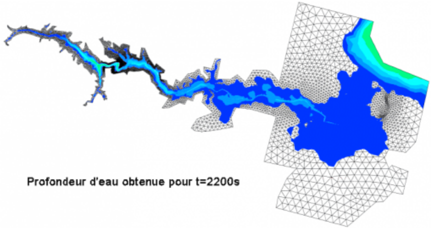
Basic ingredients

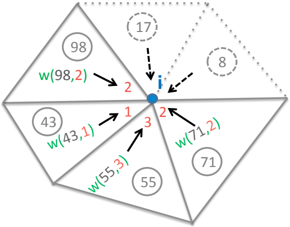
Basic ingredients

Sequential finite element assembly

Assembly step principle: $V(i) = \sum_{elements} W_e(i)$

- compute the inner node values
- accumulate W_e for every *ielem* that contains *i*





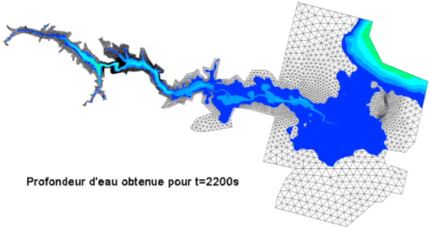
The assembly loop

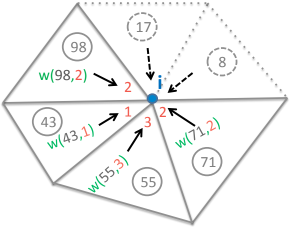
```
for idp = 1, ndp         //idp: triangular local numbering (ndp=3)
  for ielem = 1, nelem
    i = IKLE(ielem, idp)
    V(i) = V(i) + W(ielem, idp)      //i: domain global numbering
```

Sequential finite element assembly

Assembly step principle: $V(i) = \sum_{elements} W_e(i)$

- compute the inner node values
- accumulate W_e for every *ielem* that contains i



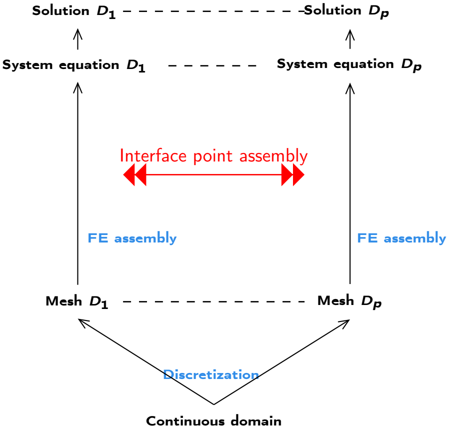


The assembly loop

```
for idp = 1, ndp          //idp: triangular local numbering (ndp=3)
  for ielem = 1, nelem
    i = IKLE(ielem, idp)   <-- LOOP INDEX INDIRECTION
    V(i) = V(i) + W(ielem, idp)      //i: domain global numbering
```

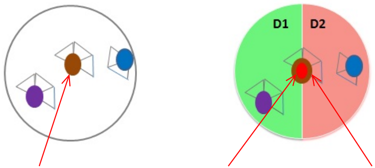
Parallel FE assembly

Parallel FE: subdomain decomposition



Exact arithmetic

sequential | parallel



$V(i) = a$ | $V(i_1) = b$ | $V(i_2) = c$

Interface point assembly

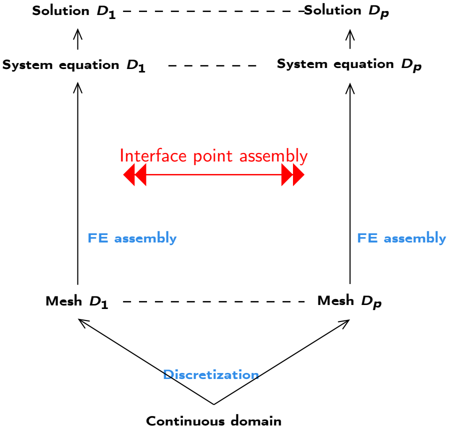
$V(i) = b + c = a$

IP assembly = communications and reductions

$V(i) = \sum_{D_k} V(i)$
subdomains D_k, $k = 1...p$

Parallel FE assembly

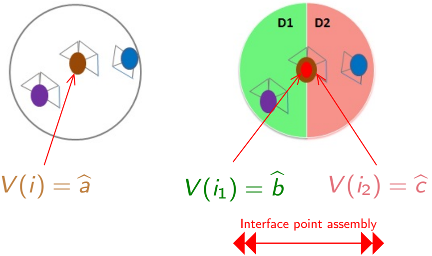
Parallel FE: subdomain decomposition



IP assembly =
communications and reductions

Floating point arithmetic

sequential parallel



$$V(i) = \widehat{b} \oplus \widehat{c} \neq \widehat{a}$$

$$V(i) = \sum_{D_k} V(i)$$

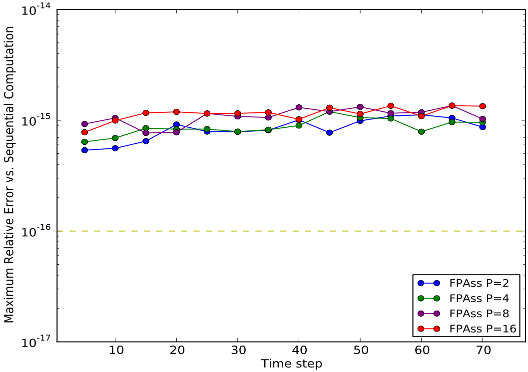
subdomains $D_k, k = 1 \ldots p$

Assembly step: example of reproducibility failure

Sequential vs. p-parallel results differ for $p = 2, 4, 8, 16$

- Assembly with the classical floating-point accumulation
- sequential *FPAss* vs. p-parallel $FPAss_p$

$$\max |FPAss_p - FPAss| / |FPAss|$$



Mean frequency wave, Nice test case, Tomawac

Basic ingredients

1. Motivations
2. Basic ingredients
   - Finite element assembly: sequential and parallel cases
   - Sources of non reproducibility in Telemac2D
   - Compensation
3. Recovering reproducibility in a finite element resolution
   - Reproducible parallel FE assembly
   - Reproducible conjugate gradient
4. Efficient and reproducible BLAS 1
5. Conclusion and work in progress

Sources of non reproducibility in Telemac2D

Culprits: theory

- Building step: interface point assembly
- Resolution with conjugate gradient: matrix-vector and dot products

Culprits: practice = optimizations

- Element-by-element storage of FE matrix and second member
- Wave equation and associated algebraic transformations
- Interface point assembly and system solving are merged

Sources of non reproducibility in Telemac2D

Telemac2D finite element method (FE)

Solution U,V

wave equation

Solution H

conjugate gradient

System equation $Ay = b$

$$\begin{pmatrix} AM1 & 0 & 0 \\ 0 & AM2 & 0 \\ 0 & 0 & AM3 \end{pmatrix} \begin{pmatrix} H \\ U \\ V \end{pmatrix} = \begin{pmatrix} CV1 \\ CV2 \\ CV3 \end{pmatrix}$$

algebraic transformation

$$\begin{pmatrix} A_{hh} & A_{hu} & A_{hv} \\ A_{uh} & A_{uu} & 0 \\ A_{vh} & 0 & A_{vv} \end{pmatrix} \begin{pmatrix} H \\ U \\ V \end{pmatrix} = \begin{pmatrix} B_h \\ B_u \\ B_v \end{pmatrix}$$

FE assembly +

algebraic computation

Mesh (elements, nodes)

Discretization

$CV2 = B_u - A_{uh}H,$

$CV3 = B_v - A_{vh}H.$

Interface node assembly:

$CV2$, $CV3$

H

AM2,AM3

Interface node assembly:

$AM1$ in each iteration

AM1,CV1

$AM2, AM3$: diagonal matrices,

$AM1 = A_{hh} - A_{hu}AM2^{-1}A_{uh} - A_{hv}AM3^{-1}A_{vh},$

$CV1 = B_h - A_{hu}AM2^{-1}B_u - A_{hv}AM3^{-1}B_v,$

Interface node assembly:

$AM2, AM3, CV1$

Sources of non reproducibility in Telemac2D

Telemac2D finite element method (FE)

Solution U,V

wave equation

$CV2 = B_u - A_{uh}H,$

$CV3 = B_v - A_{vh}H.$

Interface node assembly:

$CV2, CV3$

H

AM2,AM3

Solution H

conjugate gradient

Interface node assembly:

$AM1$ in each iteration

AM1,CV1

System equation $Ay = b$

$$\begin{pmatrix} AM1 & 0 & 0 \\ 0 & AM2 & 0 \\ 0 & 0 & AM3 \end{pmatrix} \begin{pmatrix} H \\ U \\ V \end{pmatrix} = \begin{pmatrix} CV1 \\ CV2 \\ CV3 \end{pmatrix}$$

algebraic transformation

$AM2, AM3$: diagonal matrices,

$AM1 = A_{hh} - A_{hu}AM2^{-1}A_{uh} - A_{hv}AM3^{-1}A_{vh},$

$CV1 = B_h - A_{hu}AM2^{-1}B_u - A_{hv}AM3^{-1}B_v,$

Interface node assembly:

$AM2, AM3, CV1$

$$\begin{pmatrix} A_{hh} & A_{hu} & A_{hv} \\ A_{uh} & A_{uu} & 0 \\ A_{vh} & 0 & A_{vv} \end{pmatrix} \begin{pmatrix} H \\ U \\ V \end{pmatrix} = \begin{pmatrix} B_h \\ B_u \\ B_v \end{pmatrix}$$

FE assembly +

algebraic computation

Mesh (elements, nodes)

Discretization

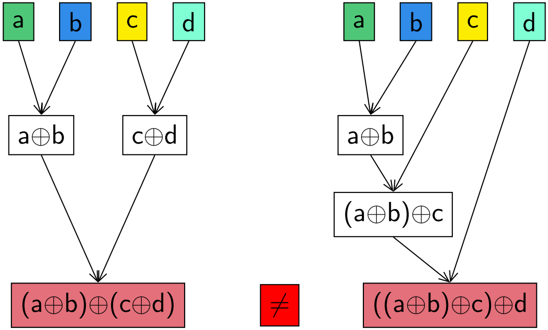
Objective

Correct sources of non-reproducibility
to compute reproducible system and solutions

Basic ingredients

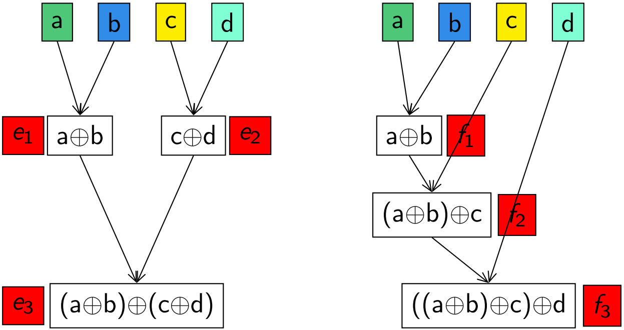
Parallel reduction and compensation techniques

Reproducibility failure of the parallel reduction



Parallel reduction and compensation techniques

Compensation principle

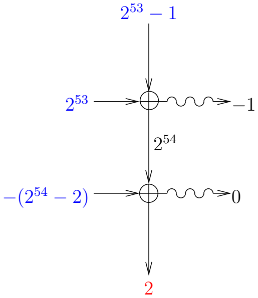


$$((a \oplus b) \oplus (c \oplus d)) \oplus ((e_1 \oplus e_2) \oplus e_3) = (((a \oplus b) \oplus c) \oplus d) \oplus ((f_1 \oplus f_2) \oplus f_3)$$

Compensated summation: one example

IEEE binary64 (double): $x_1 = 2^{53} - 1$, $x_2 = 2^{53}$ and $x_3 = -(2^{54} - 2)$.
Exact sum: $x_1 + x_2 + x_3 = 1$.

Classic summation

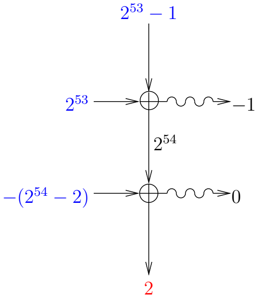


Relative error = 1

Compensated summation: one example

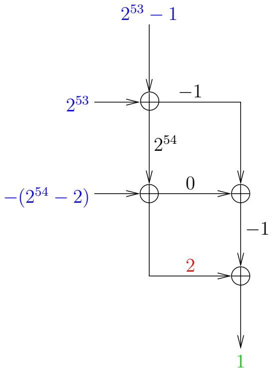
IEEE binary64 (double): $x_1 = 2^{53} - 1$, $x_2 = 2^{53}$ and $x_3 = -(2^{54} - 2)$.
Exact sum: $x_1 + x_2 + x_3 = 1$.

Classic summation



Relative error = 1

Compensation of the rounding errors

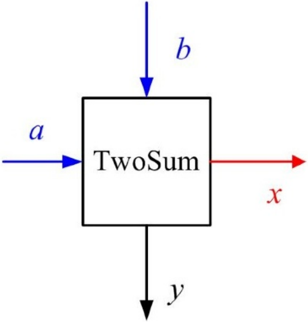


The exact result is computed

Rounding errors are computed with EFT

2Sum (Knuth, 65), Fast2Sum (Dekker, 71) for base ≤ 2 and RTN.

$$a + b = x + y, \text{ with } a, b, x, y \in \mathbb{F} \text{ and } x = a \oplus b.$$



Algorithm (Knuth)

function [x,y] = 2Sum(a,b)

$x = a \oplus b$

$z = x \ominus a$

$y = (a \ominus (x \ominus z)) \oplus (b \ominus z)$

Algorithm ($|a| > |b|$, Dekker)

function [x,y] = Fast2Sum(a,b)

$x = a \oplus b$

$z = x \ominus a$

$y = b \ominus z$

Other existing techniques

Existing techniques to recover numerical reproducibility in summation

- Accurate compensated summation [6]
- Demmel-Nguyen's reproducible sums [3]
- Integer convertion [7]

Recovering reproducibility in a finite element resolution

Recovering reproducibility in Telemac2D

Culprits

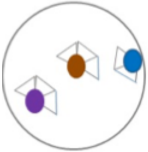
- Building step: interface point assembly
- Resolution: matrix-vector and dot products
- Element-by-element storage of FE matrix and second member
- Wave equation (decoupling) and associated algebraic transformations
- Interface point (IP) assembly and system solving are merged

Reproducible resolution: principles

- Compensate FE assembly of inner nodes
- Propagate rounding errors and compensate while assembling the IP
- Compensate the EBE matrix-vector products
- Compensate the MPI parallel dot products
- vector $V \to [V, E_V] \to V + E_V$

Accurate compensated assembly: the sequential case

for idp= 1, ndp
  for ielem= 1, nelem
    i=IKLE(ielem, idp)
    X(i)=X(i)+W(ielem,idp)



D1

FE assembly

Contribution $W_i(e_1)$ for every node mesh i belonging to element $e_1 \ldots e_n$

Computation and accumulation of rounding errors Δ

$$X(i) = W_i(e_1) + W_i(e_2) + \ldots + W_i(e_n)$$

Δ_1 Δ_2 Δ_n

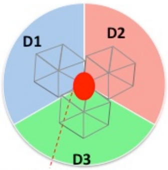
$$E(i) = \Delta_1 + \Delta_2 + \ldots + \Delta_n$$

Compensation of the FE assembly for every mesh node I

$$X(i)=X(i)+E(i)$$

The parallel case is easy to derive

```
for idp=1, ndp
  for ielem=1, nelem
    i=IKLE (ielem, idp)
    X(i)=X(i)+ W(ielem, idp)
```



i : Interface Point

Interface Point assembly

D1 D2 D3

Subdomain FE assembly.

Contribution $W_i(e_k)$ for every node mesh i belonging to elem $e_1,\ldots, e_n$

Computation and accumulation of rounding errors Δ

similar

$X(i) = W_i(e_1) + W_i(e_2) + \ldots + W_i(e_n)$

$\Delta_1 \quad \Delta_2 \quad \Delta_n$

$E(i) = \Delta_1 + \Delta_2 + \ldots + \Delta_n$

similar

Interface point assembly

i : interface point between D1,D2,D3

Accumulation of errors Δ and ε

X(i), E(i)

$X(i) = X(i) + X(i) + X(i)$

$\varepsilon_1 \quad \varepsilon_2$

$E(i) = ((E(i) + E(i) + \varepsilon_1) + E(i) + \varepsilon_2)$

X(i), E(i)

Compensation of the FE assembly for every mesh node i

similar

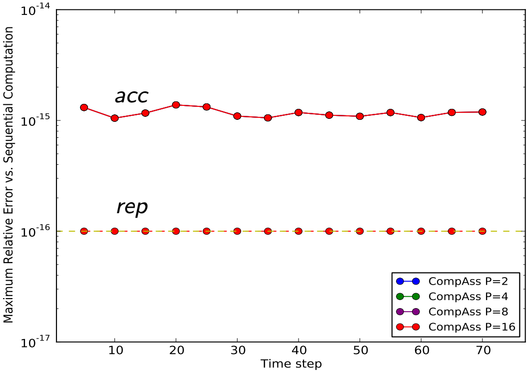
$X(i)=X(i)+E(i)$

similar

Accurate compensated assembly gives reproducibility

Reproducibility in Tomawac

- Reproducibility and accuracy



A^s: sequential, A^p: p-parallel

$\max_{rel}(A_1, A_2) = |A_1 - A_2|/|A_2|$

Accuracy (of FPSum):

$acc = \max_{rel}(CompAss^p, FPAss^s)$

Reproducibility:

$rep = \max_{rel}(CompAss^p, CompAss^s)$

Mean frequency wave, Nice test case, Tomawac

Recovering reproducibility in a finite element resolution

Towards a reproducible conjugate gradient

A=[$AM1$, E_{AM1}]

B=$CV1$

Initialization:

$$r^0 = AX^0 - B;\ \text{a given } d^0$$

$$\rho^0 = \frac{(r^0, d^0)}{(Ad^0, d^0)};\ X^1 = X^0 - \rho^0 d^0$$

Iterations:

$$r^m = r^{m-1} - \rho^{m-1} Ad^{m-1}$$

$$d^m = r^m + \frac{(r^m, r^m)}{(r^{m-1}, r^{m-1})} d^{m-1}$$

$$\rho^m = \frac{(r^m, d^m)}{(d^m, Ad^m)}$$

$$X^{m+1} = X^m - \rho^m d^m$$

$\rightarrow$ X=H

Sources of non-reproducibility :

Matrix-vector product

Dot product

Last steps to compensate

- Conjugate gradient
- Matrix-vector product
- Dot product
  - ponderated dot product
  - MPI reduced dot product

Reproducible conjugate gradient

A=[*AM1*, E_{AM1}]

B=*CV1*

Initialization:

$$r^0 = AX^0 - B; \text{ a given } d^0$$

$$\rho^0 = \frac{(r^0, d^0)}{(Ad^0, d^0)}; \; X^1 = X^0 - \rho^0 d^0$$

Iterations:

$$r^m = r^{m-1} - \rho^{m-1} Ad^{m-1}$$

$$d^m = r^m + \frac{(r^m, r^m)}{(r^{m-1}, r^{m-1})} d^{m-1}$$

$$\rho^m = \frac{(r^m, d^m)}{(d^m, Ad^m)}$$

$$X^{m+1} = X^m - \rho^m d^m$$

$\longrightarrow$ **X=*H***

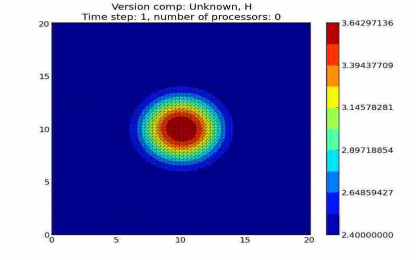
Not necessarily more accurate but reproducible

Same errors in compensated values for both sequential and parallel executions

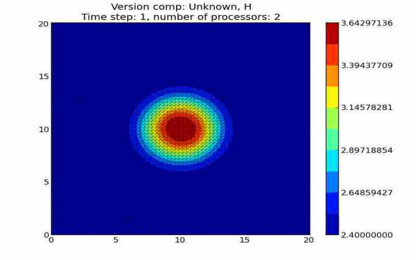
Reproducible Telemac2D!

Time step 1

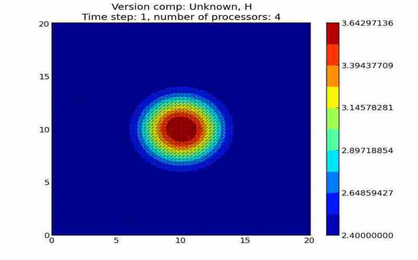
p=1



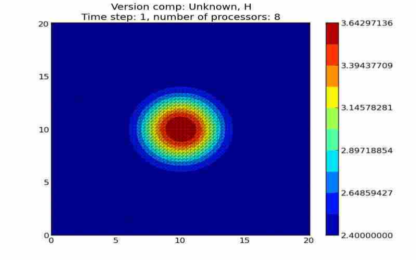
p=2



p=4

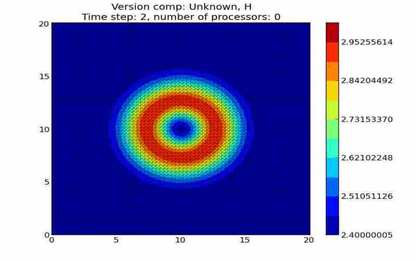


p=8

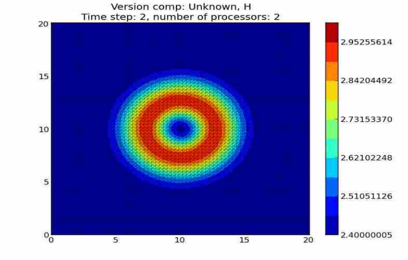


Time step 2

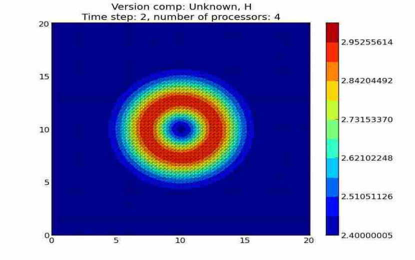
p=1



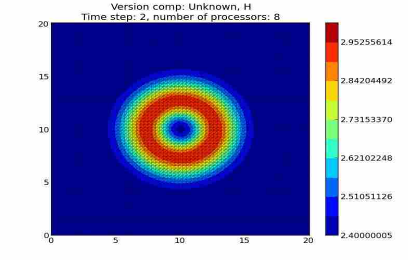
p=2



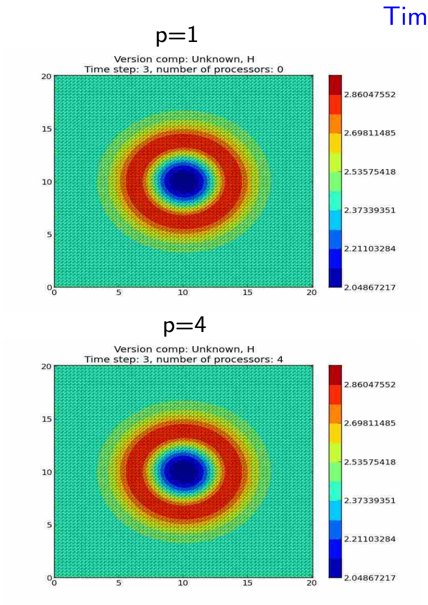
p=4



p=8

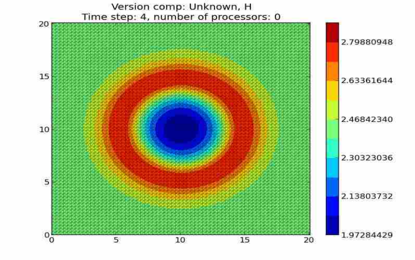


Reproducible Telemac2D!

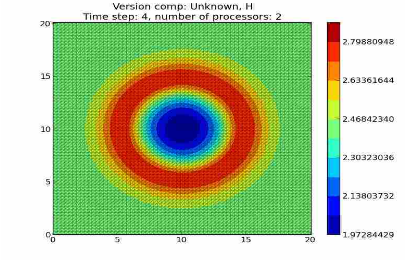


Time step 4

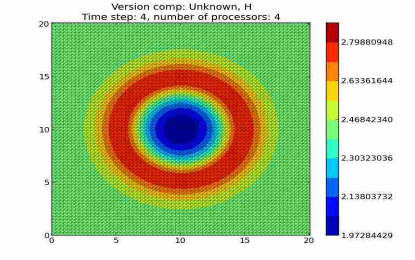
p=1



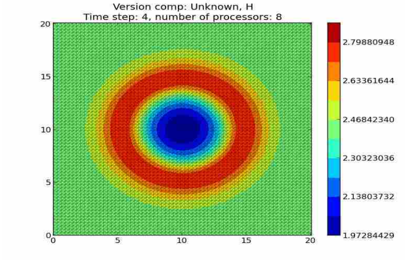
p=2



p=4

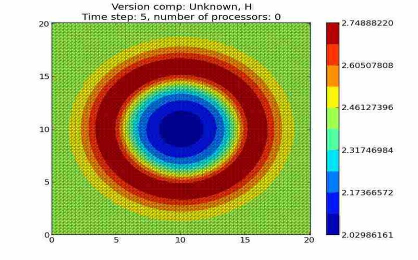


p=8

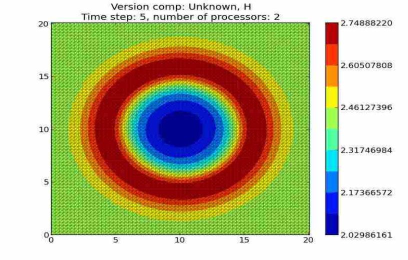


Time step 5

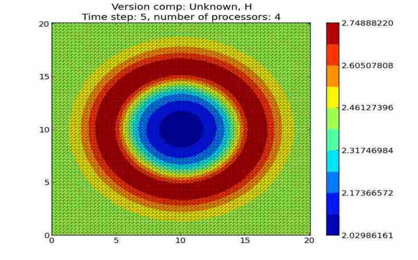
p=1



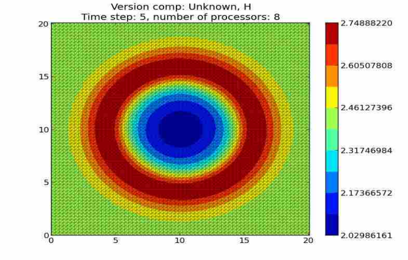
p=2



p=4

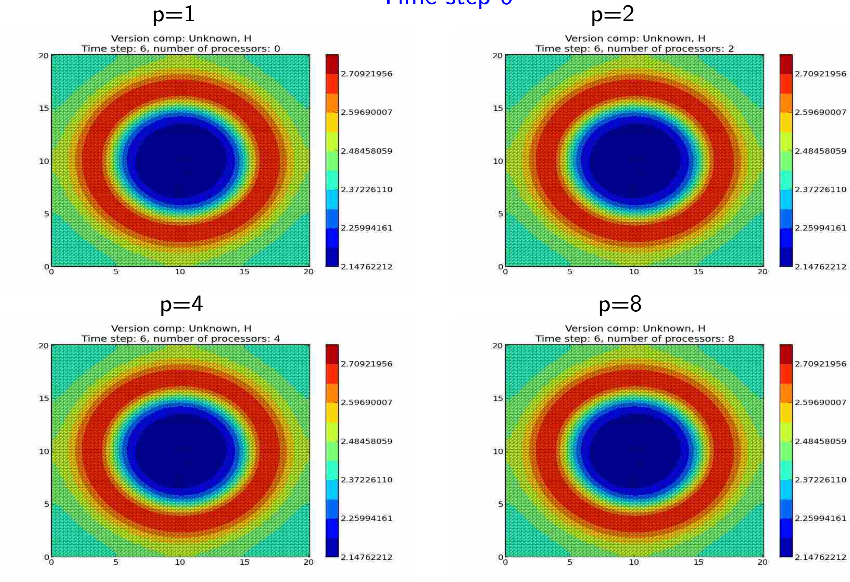


p=8



Reproducible Telemac2D!

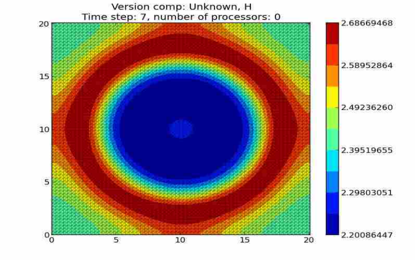
Time step 6



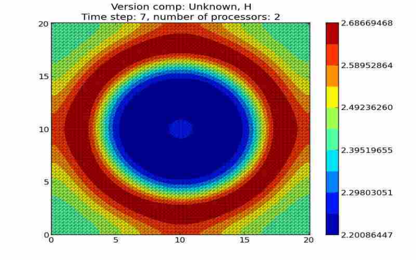
Reproducible Telemac2D!

Time step 7

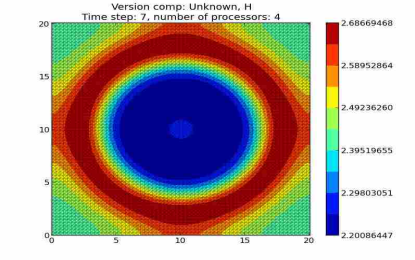
p=1



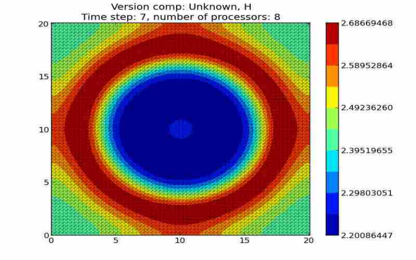
p=2



p=4



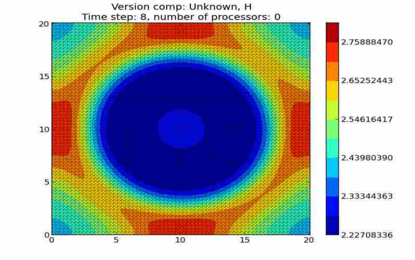
p=8



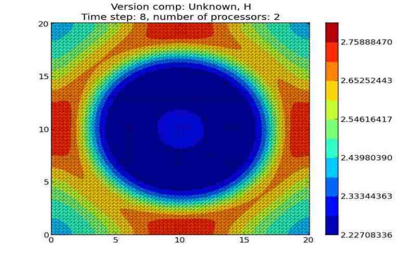
Reproducible Telemac2D!

Time step 8

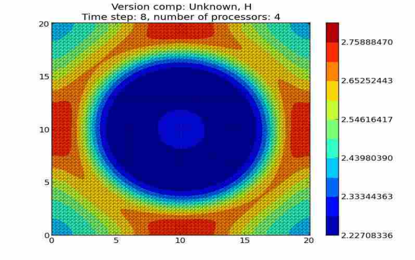
p=1



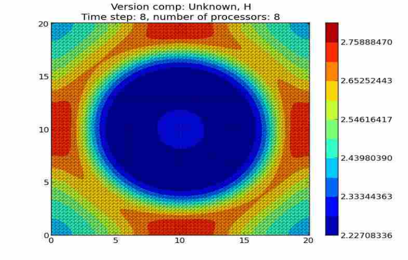
p=2



p=4



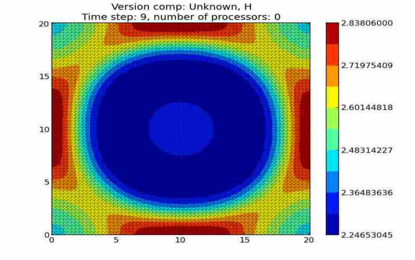
p=8



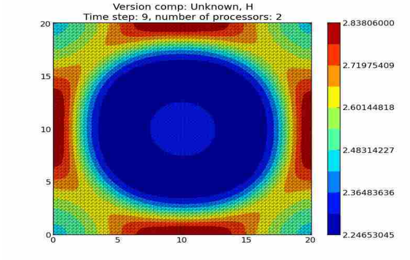
Reproducible Telemac2D!

Time step 9

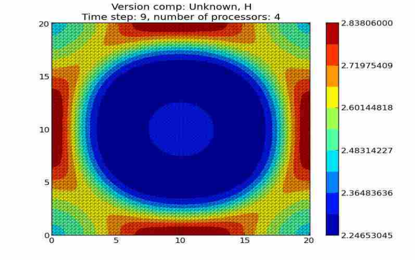
p=1



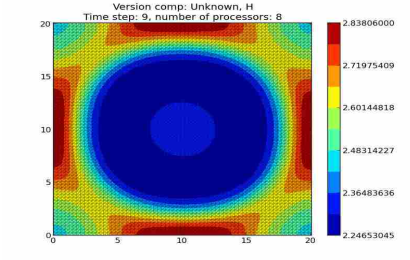
p=2



p=4



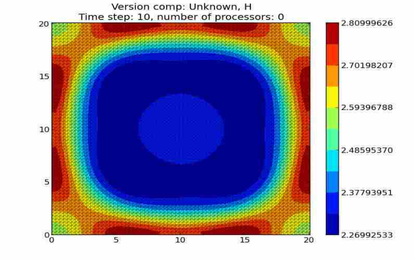
p=8



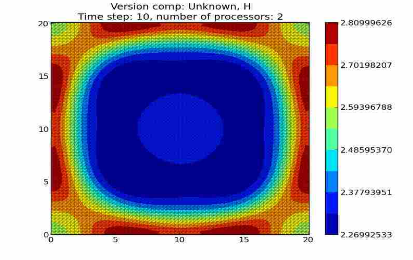
Reproducible Telemac2D!

Time step 10

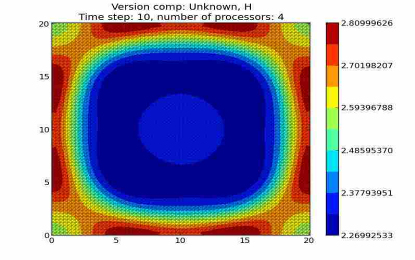
p=1



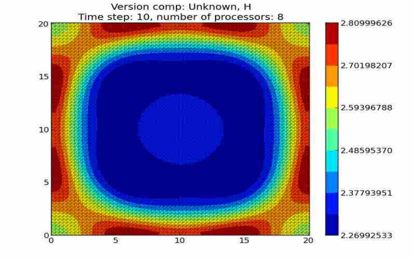
p=2



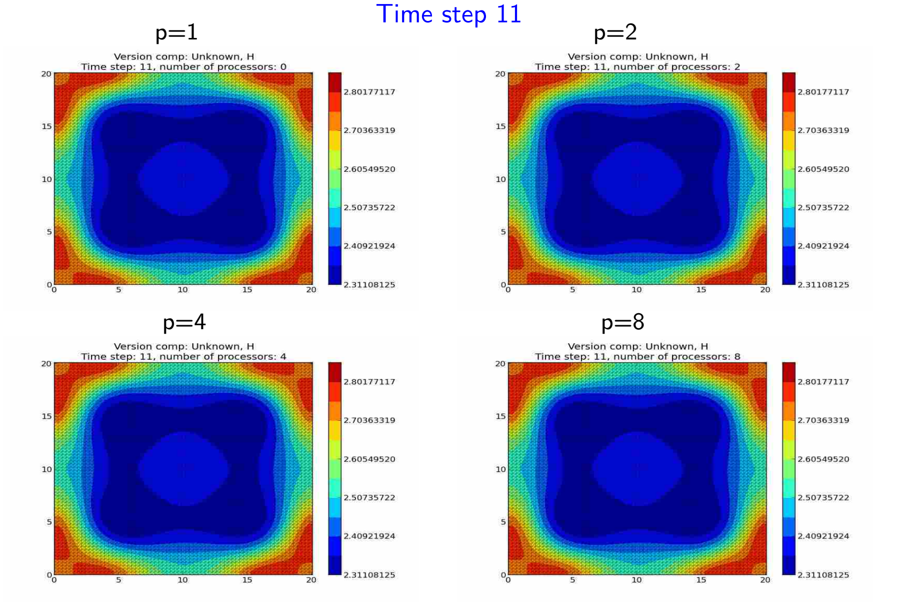
p=4



p=8

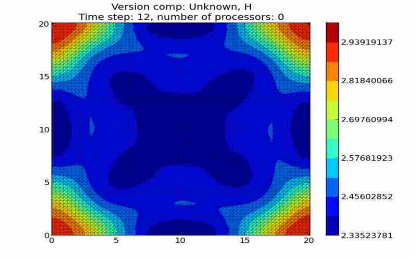


Reproducible Telemac2D!

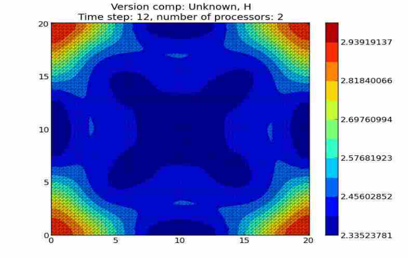


Time step 12

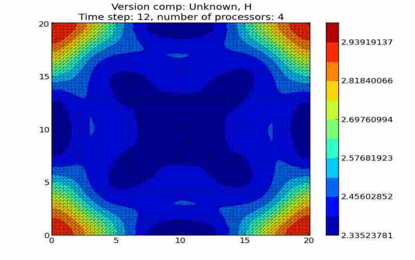
p=1



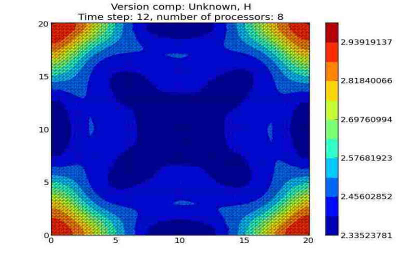
p=2



p=4



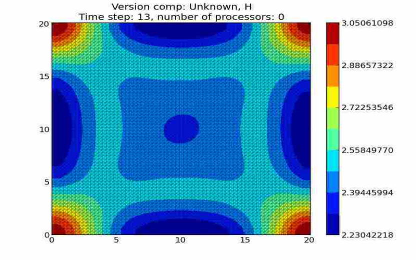
p=8



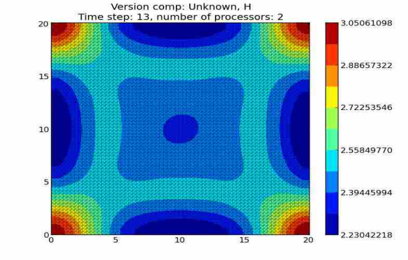
Reproducible Telemac2D!

Time step 13

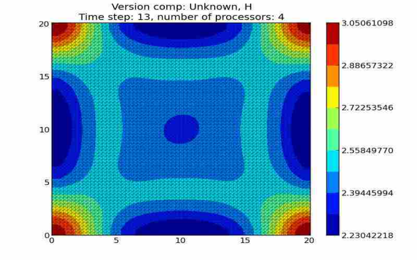
p=1



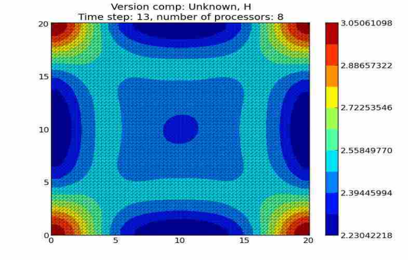
p=2



p=4



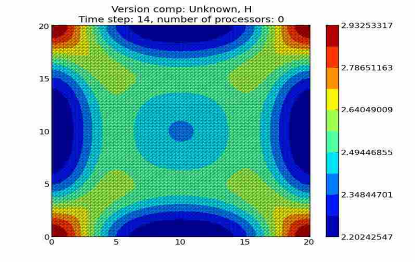
p=8



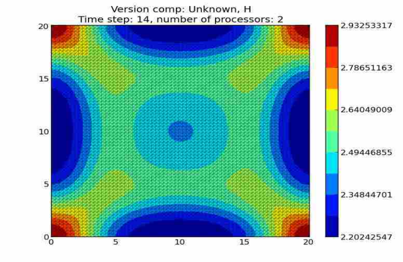
Reproducible Telemac2D!

Time step 14

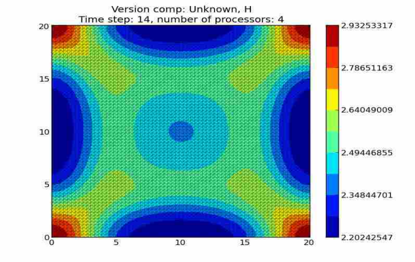
p=1



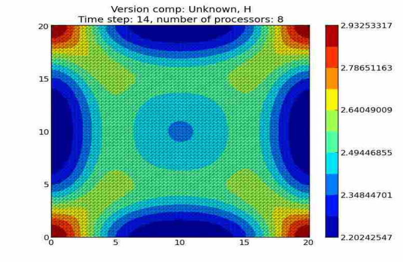
p=2



p=4



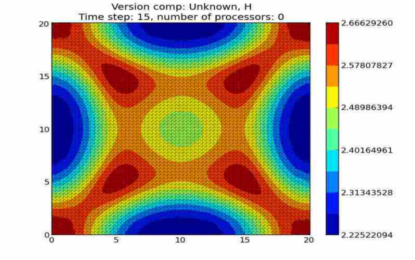
p=8



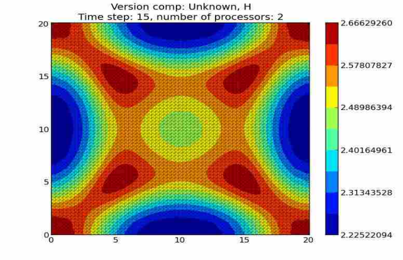
Reproducible Telemac2D!

Time step 15

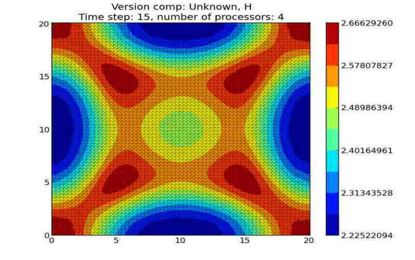
p=1



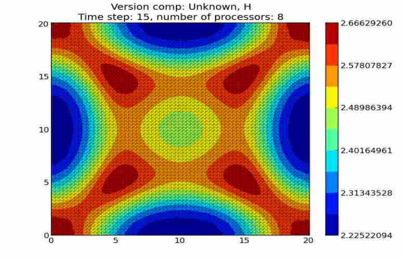
p=2



p=4



p=8



Efficient and reproducible BLAS 1

Numerical reproducibility for the BLAS

BLAS + correctly rounded sums

- Dot product of length $n \to$ sum of length $2n$
- A correctly rounded result is reproducible
- A large panel of algorithms for faithful or correctly rounded sums

Motivation

- How to benefit from these CR sums for reproducible BLAS?
- Is the over-cost acceptable in practice for reproducible BLAS?

Current results

- BLAS 1 : asum, dot, norm2
- openMP for shared memory
- Hybrid openMP-MPI for shared+distributed memory

Overview I

Our methodology

1. Optimization and choice of the best sequential CR sums
2. Deriving parallel CR sums
3. Application to reproducible BLAS-1 routines

The starting point: sequential summation algorithms

- Accurate: Sum-K [6]
- Faithful: AccSum [10], FastAccSum [9]
- Correctly rounded (in RtN): iFastSum, HybridSum [14], OnlineExact sum [15]

Overview II

Reproducible parallel BLAS-1: algorithmic choice

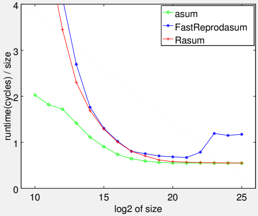
- Rasum: parallel Sum K as in [13] with $K = 2$ for $n \leq 10^7$.
- Rdot: FastAccSum (small n) or modified OnLineExact (large n)
- Rnrm2: Rdot+IEEE sqrt $\rightarrow$ reproducible only

All details in [1]

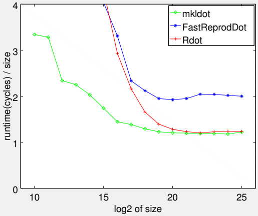
Efficiency of Reproducible Level 1 BLAS,
C. Chohra, Ph. L., D. Parello.
SCAN 2014 Post-Conference Proceedings
Lecture Notes of Computer Science (2015).
`http://hal-lirmm.ccsd.cnrs.fr/lirmm-01101723`

Parallel BLAS-1: Runtime overcost for reproducibility

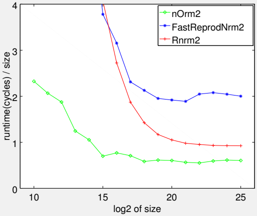
Runtime/size of parallel level 1 BLAS, up to 16 threads, cond=10^{32}



asum



dot



nrm2

Hardware and software env.

- socket: Xeon E5-2660 (L3 cache = 20 M).
- 2 cores, 8 cores on each socket.
- OpenMP 4.0 (Intra socket parallelism).
- Compare vs. Intel MKL 11

Parallel BLAS-1: Scalability

Hardware and software env.

- OCCIGEN: 26^{th} supercomputer in top500 list.
- 4212 cores, 12 cores on each socket.
- OpenMP 4.0 (Intra socket parallelism).
- OpenMPI (Inter socket communications).

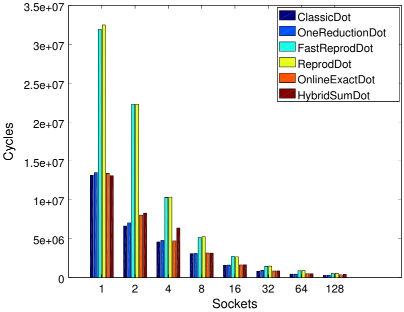
Parallel BLAS-1: Scalability

Configurations

- #sockets = 1 .. 128.
- #threads = 12 per socket.

Dataset

- Entry vectors size = 10^7.
- Condition number = 10^{32}.



Results

- Good scaling for large datasets.
- Two communications cost limits ReprodDot and FastReprodDot.
- We need only one communication for OneReduction, HybridSum and OnlineExact.

Time to conclude

To conclude

Feasibility ?

Do existing techniques easily provide reproducibility
to large scale industrial scientific software?

Efficiency ?

Do correctly rounded summation algorithms provide
efficient implementations of reproducible parallel BLAS routines?

Conclusion I

Numerical reproducibility

- How to remain confident facing the complexity of today's computational systems? and tomorrow?
- Sources: floating-point peculiarities and parallel reduction, data alignment, operator choices, compiler optimizations

Feasability

- Existing techniques are efficient enough and more or less easy to apply
- More precision, less errors or even exact computation

Reproducibility at the large scale: the openTelemac case

- Complex, large and real simulations are tractable
- Difficult to automatize but easier to pass the methodology on to software developpers
- Next step for Telemac 2D: more complex physical and numerical issues

Conclusion II

Efficiency

- Convincing reproducible BLAS level 1
- Hybrid openMP+MPI and scalability "in the large"
- Current step: Intel MIC (Xeon Phi)
- Next steps: BLAS-2, optimistic but BLAS-3, no future!

Numerical reproducibilty: cons/pros

- *Non reproducibility reveals bugs or numerical problems!*
- *just return 1 ... it is reproducible indeed!*
- Reproducibility is one factor towards numerical quality . . . as theorems, experiments, tools that yields error bounds, stability conditions, accuracy
- Reproducibility for the validation steps, not for the actual/operating mode
- Sequential $\rightarrow$ parallel implementation: reproducibility = no more bug . . . both implementations can be wrong!

Références I

C. Chohra, P. Langlois, and D. Parello.
Efficiency of Reproducible Level 1 BLAS.
In SCAN 2014 Post-Conference Proceedings, Lecture Notes of Computer Science, pages 1–10, July 2015.

J. M. Corden and D. Kreitzer.
Consistency of Floating-Point Results using the Intel Compiler or Why doesn't my application always give the same answer?
Intel Corporation, Aug. 2014.

J. W. Demmel and H. D. Nguyen.
Fast reproducible floating-point summation.
In Proc. 21th IEEE Symposium on Computer Arithmetic. Austin, Texas, USA, 2013.

Y. He and C. Ding.
Using accurate arithmetics to improve numerical reproducibility and stability in parallel applications.
J. Supercomput., 18:259–277, 2001.

J.-M. Hervouet.
Hydrodynamics of free surface flows: Modelling with the finite element method.
John Wiley & Sons, 2007.

T. Ogita, S. M. Rump, and S. Oishi.
Accurate sum and dot product.
SIAM J. Sci. Comput., 26(6):1955–1988, 2005.

Références II

Open TELEMAC-MASCARET. v.7.0, Release notes.
`www.opentelemac.org`, 2014.

R. W. Robey, J. M. Robey, and R. Aulwes.
In search of numerical consistency in parallel programming.
Parallel Comput., 37(4-5):217–229, 2011.

S. M. Rump.
Ultimately fast accurate summation.
SIAM J. Sci. Comput., 31(5):3466–3502, 2009.

S. M. Rump, T. Ogita, and S. Oishi.
Accurate floating-point summation – part I: Faithful rounding.
SIAM J. Sci. Comput., 31(1):189–224, 2008.

M. Taufer, O. Padron, P. Saponaro, and S. Patel.
Improving numerical reproducibility and stability in large-scale numerical simulations on gpus.
In IPDPS, pages 1–9. IEEE, 2010.

O. Villa, D. G. Chavarría-Miranda, V. Gurumoorthi, A. Márquez, and S. Krishnamoorthy.
Effects of floating-point non-associativity on numerical computations on massively multithreaded systems.
In CUG 2009 Proceedings, pages 1–11, 2009.

Références III

N. Yamanaka, T. Ogita, S. Rump, and S. Oishi.
A parallel algorithm for accurate dot product.
Parallel Comput., 34(6–8):392 – 410, 2008.

Y.-K. Zhu and W. B. Hayes.
Correct rounding and hybrid approach to exact floating-point summation.
SIAM J. Sci. Comput., 31(4):2981–3001, 2009.

Y.-K. Zhu and W. B. Hayes.
Algorithm 908: Online exact summation of floating-point streams.
ACM Trans. Math. Software, 37(3):37:1–37:13, Sept. 2010.