Improving Checkpoint-Restart with Lossy Compression

4th NCSA Blue Waters Symposium for Petascale Science and Beyond

Jon Calhoun[1], Franck Cappello[1,2], Luke Olson[1], and Marc Snir[1,2]

[1]University of Illinois at Urbana-Champaign
[2]Argonne National Laboratory

June 13, 2016

Checkpoint-Restart

Checkpoint-restart is a key component of HPC applications:

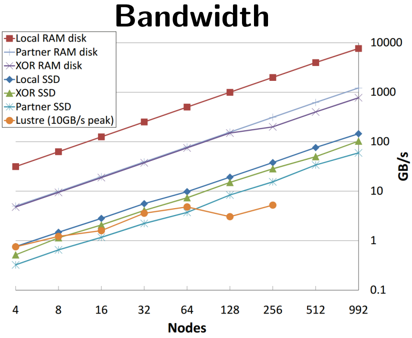
- Recover from application failure
- Extend job execution beyond a single time allocation
- Selectively diverge the simulation to test various parameters
- Visualization files

TSA Security line at ORD [CNN 2016]

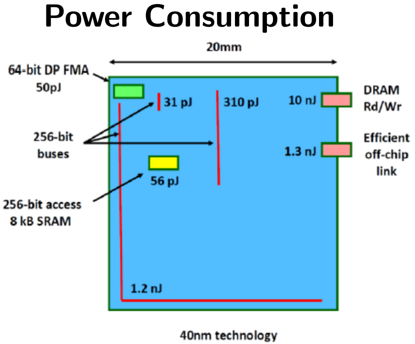
Efficient checkpoint-restart allows for efficient applications

Data Movement Problem

Data movement is set to become a limiter to HPC performance due to:



Aggregate write bandwidth [Moody et al. 2010]



Power consumption for CPU operations. [Keckler 2011]

Solutions for the Data Movement Problem

- Redundant computation recreates data that is too expensive to transfer
- Multi-level checkpoint schemes–e.g., SCR [Moody et al. 2010] and FTI [Bautista-Gomez 2011], effectively utilize the memory hierarchy
- Data compression–e.g., Gzip, reduces volume of data transferred

Combining known solutions yields new and potentially attractive solutions to this problem

Compression

Data compression techniques fall into two categories:

Lossless

- No data loss
- Small compression factors
- Can be costly (time)
- Research on integration in hardware memories [Pekhimenko et al. 2012] [Sardashti, Seznec, and Wood 2014]

Lossy

- Some data perturbation
- Larger compression factors
- Low/moderate cost
- Research on using in HPC checkpointing [Laney et al. 2013] [Ni et al. 2014] [Sasaki et al. 2015]

Solutions to many HPC applications are approximations

Let's investigate how lossy compression can be effectively used

Selecting the Lossy Compression Error Tolerance

All lossy compressors allow you to select some form of error tolerance, but interpretation and selection of this error is non-trivial opening numerous questions:

- What does the tolerance mean?
- Number of bits preserved?
- Absolute error?
- Relative error?
- What tolerance does my simulation need?

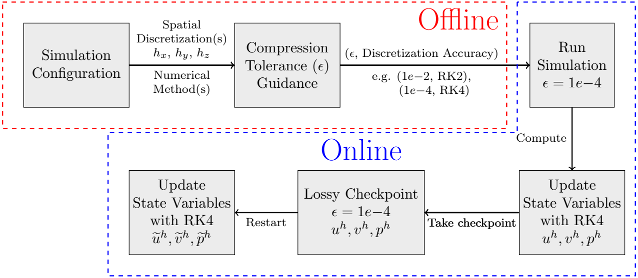
Understanding impact of error tolerance is critical for effective use of lossy compression

Let's think of compression error as numerical error

Selecting Lossy Compression Error Tolerance

Numerical method's approximation accuracy depends on order of method and spatial discretizations

- Use RK4 to solve 1-D PDE with $h_x = 0.1$
- Our solution u^h is accurate to $1e{-}4$
- $\widetilde{u}^h = u^h + \epsilon$ is equivalent to u^h if $\epsilon < 1e{-}4$.
- $\widetilde{u}^h$ with $\epsilon > 1e{-}4$ can be mapped to another related method – e.g., $\epsilon = 1e{-}2$ is RK2.



1-D Examples

Test Problems:

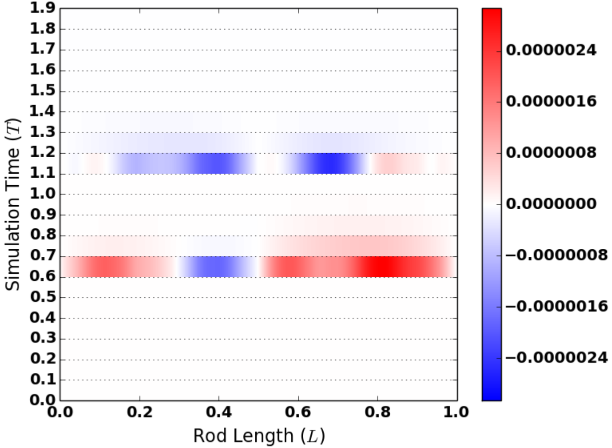
- 1-D heat equation - constant temperature boundaries with point source in middle of domain
- 1-D advection - sine wave moving to right with periodic boundary conditions

Lossy compressor:

- SZ version 0.5.14 [Di and Cappello 2016]
- Compression error tolerance $\mathcal{O}(h^p) > \epsilon$
- Use relative and absolute error bounds to enforce ϵ

Goal: Investigate if accuracy bound is preserved
$\mathcal{O}(h^p) > \| u^h - \widetilde{u}^h \|_\infty$

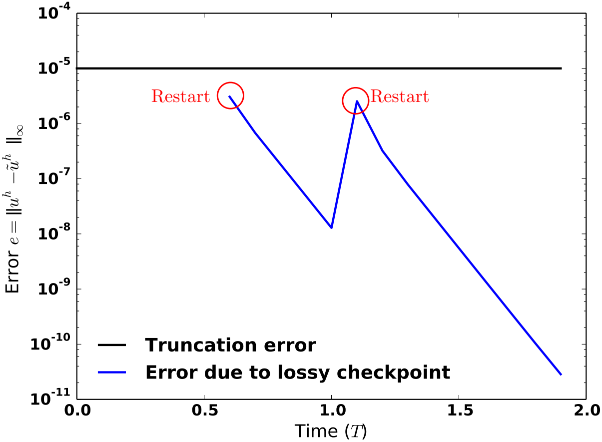
1-D Heat Example



Restart from lossy checkpoint at time 0.6 and 1.1

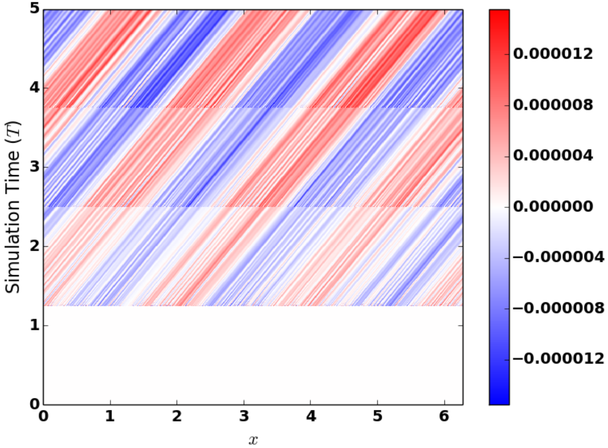
Error removed as we converge to time independent solution

1-D Heat Example



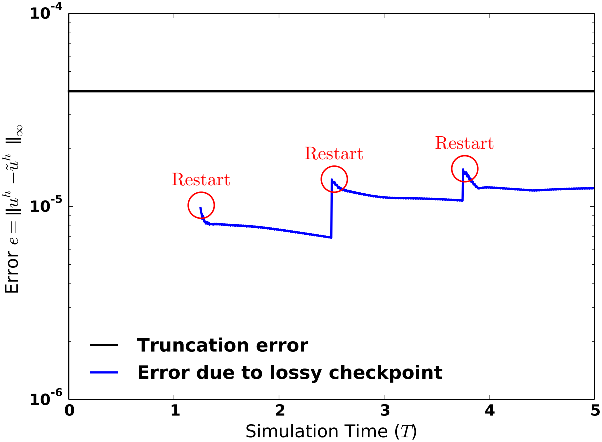
Error remains less than simulation accuracy (truncation error)

1-D Advection Example



Restart from lossy checkpoint at time 1.25, 2.5, and 3.75
Error accumulates and moves in direction of flow

1-D Advection Example



Error remains less than simulation accuracy (truncation error)
Accumulation of error with each checkpoint will eventually effect accuracy

1-D Summary

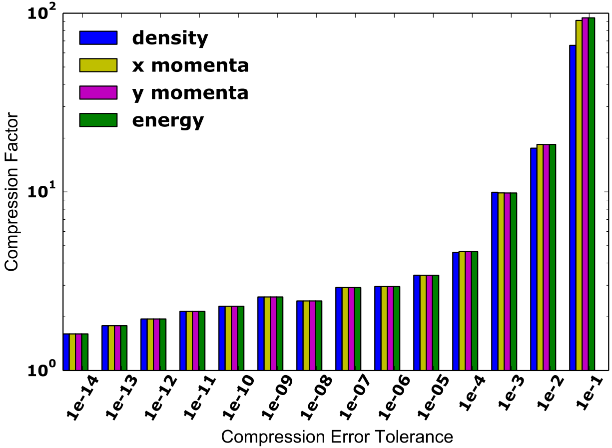
Compressing below accuracy level of simulation allows for no change in the simulation

PDE properties and boundary conditions can remove or propagate error

How do these results change when looking at full applications?

- PlasComCM: 2D Navier-Stokes flow past fixed cylinder
- Nek5000: 2D simulation of a passive heat transfer in a pipe flow

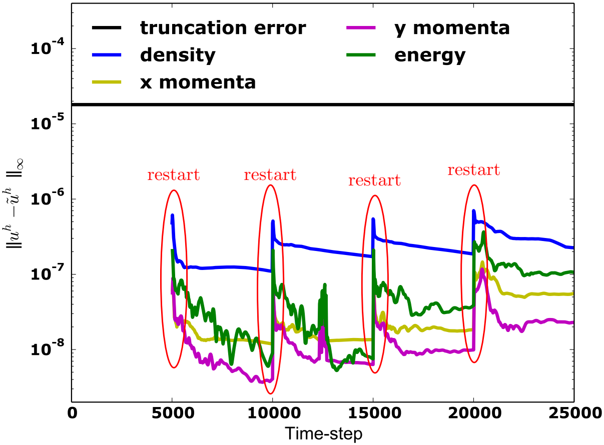
PlasComCM - Compression Factor



Compression factor depends greatly on compression error tolerance

3.25x compression factor for our tolerance $1e{-}5$

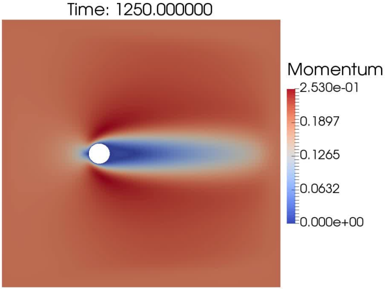
PlasComCM - Max Error



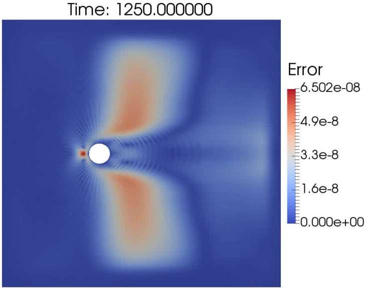
Error accumulates with each checkpoint, but is slowly removed
Accumulation is still less than truncation error

PlasComCM - Simulation

Simulation

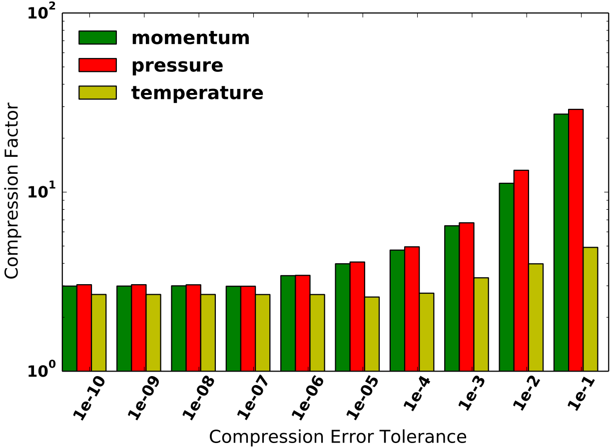


Error



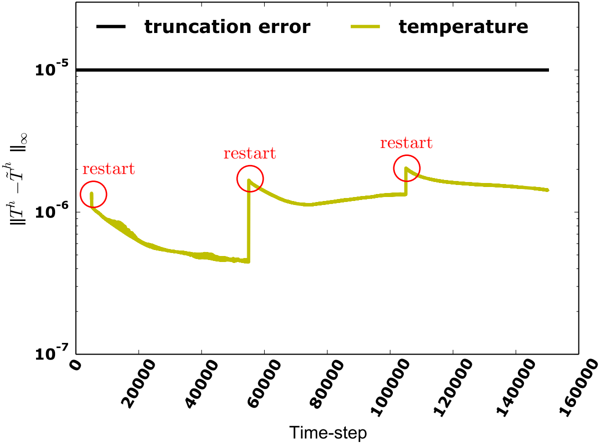
$\mathcal{O}(h^p) = 1.8e{-5}$ $\epsilon = 1e{-5}$

Nek5000 - Compression Factor



Compression factor falls in two regimes (artifact of compressor)
2.7x compression factor for our tolerance $1e{-6}$

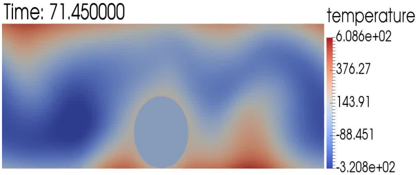
Nek5000 - Max Error

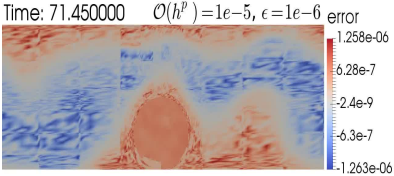


Error accumulates with each checkpoint, but is never removed
With enough checkpoints, error will exceed accuracy threshold

Nek5000 - Simulation

Simulation **Error**





Conclusions

Simulation accuracy can be effectively used to select the compression error tolerance

Compressing below accuracy level of simulation allows for no change in the simulation

PDE properties and boundary conditions can remove or propagate error

Error accumulation depends on physical domain simulated

Future Work

Build performance model to determine when lossy compression is viable for checkpoint-restart

Evaluate other methods of selecting compression tolerances

Look at using multiple compression tolerances based on physical/simulation properties

Acknowledgments

- This work was sponsored by the Air Force Office of Scientific Research under grant FA9550-12-1-0478
- This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications

Thank you

Any questions?