6th European Conference on Computational Mechanics (ECCM 6)
7th European Conference on Computational Fluid Dynamics (ECFD 7)
1115 June 2018, Glasgow, UK

OPTIMIZATION-BASED PROPERTY-PRESERVING SOLUTION RECOVERY FOR FAULT-TOLERANT SCALAR TRANSPORT

DENIS RIDZAL[1] AND PAVEL BOCHEV[2]

[1] Optimization and Uncertainty Quantification,
Sandia National Laboratories, MS-1320,
Albuquerque, NM 87185-1320, USA
dridzal@sandia.gov, http://www.sandia.gov/∼dridzal

[2] Computational Mathematics
Sandia National Laboratories, MS-1320,
Albuquerque, NM 87185-1320, USA
pbboche@sandia.gov, http://www.sandia.gov/∼pbboche

Key words: Optimization, scalar transport, finite element method, failure

Abstract. As the mean time between failures on the future high-performance computing platforms is expected to decrease to just a few minutes, the development of "smart", property-preserving checkpointing schemes becomes imperative to avoid dramatic decreases in application utilization. In this paper we formulate a generic optimization-based approach for fault-tolerant computations, which separates property preservation from the compression and recovery stages of the checkpointing processes. We then specialize the approach to obtain a fault recovery procedure for a model scalar transport equation, which preserves local solution bounds and total mass. Numerical examples showing solution recovery from a corrupted application state for three different failure modes illustrate the potential of the approach.

1 INTRODUCTION

Today, the mean time between failures on the leading petascale supercomputers is measured in hours. For example, IBM's design target for the overall mean time between failures for a 96-rack, 98,304-node Blue Gene/Q system is 72 hours [4]. The mean time between the occurrences of double-bit errors in graphics processing units (GPUs) on the Titan supercomputer is 160 hours [12], while the overall mean time between GPU failures is approximately 40 hours [13]. As high-performance computing (HPC) moves toward the exascale, the mean time between failures is projected to shrink to a few minutes [3, 5], while only small speed improvements are expected in the disk-based checkpointing of the full application state. As a result, the lack of "smart" fault tolerance technologies could become one of the *critical stumbling blocks* on the path to exascale. In particular, it could drive *application utilization*, a measure of useful work performed by HPC systems, to essentially zero. Schroeder et. al. [11] argue that taking faster checkpoints is the most effective and possibly the only viable strategy for sustained application utilization at extreme scales, recommending that *any application capable of compressing its checkpoint size should pursue this path.*

In this paper we present a new, optimization-based (OB) framework for efficient, property-preserving checkpoint compression and recovery algorithms for fault-tolerant simulations on HPC platforms. Our approach separates the preservation of the desired physical properties in the simulated solution from the data compression and recovery processes. For example, for compression one can minimize the misfit between "measurements" of the application state and a sparse representation of that state, subject to constraints that include hardware considerations, such as the available checkpoint space, as well as some judiciously chosen physical properties. In our model problem involving scalar transport, the physical properties of interest are, e.g., local extrema, mass in each physical subdomain and locations of discontinuities. For the OB property-preserving recovery, a similar optimization process can be applied, where the misfit function may be based on the best available (corrupted) application state, and where the checkpointed (compressed) data and the desired physical properties are used to formulate constraints. Therefore, our OB approach motivates a new family of *lossy, property-preserving compression and recovery algorithms*, which enable a graceful degradation of simulation accuracy in the presence of hardware faults.

For both compression and recovery, a representation of the system state is sought, which minimizes the distance to a best available *target* that may not be property preserving but is believed to be accurate, subject to *constraints* enforcing the desired physical properties and other available data. For the remainder of the paper, we focus primarily on the recovery process, and note that similar formulations can be developed for compression. In a nutshell, given $u_D \in \mathbb{R}^d$, likely a corrupted or low-resolution snapshot of the application state $u \in \mathbb{R}^n$, a nonlinear state-to-snapshot map $C : \mathbb{R}^n \to \mathbb{R}^d$, a measure of state-to-snapshot misfit $\mathcal{J} : \mathbb{R}^d \to \mathbb{R}$, and constraints $L(u)$, where the nonlinear operator $L : \mathbb{R}^n \to \mathbb{R}^m$ enforces discrete physical invariants and/or inequalities (generically labeled 'properties'), we recover the application state by solving the optimization problem

$$\underset{u \in \mathbb{R}^n}{\text{minimize}} \; \mathcal{J}(C(u) - u_D) \qquad \text{subject to} \qquad L(u) \geq 0 \,. \tag{1}$$

We note that u need not denote the full application state; where appropriate, local or partial recovery may be used; for instance, in [7] Heroux discusses a local failure local recovery paradigm for exascale applications. Alternatively, one may formally incorporate the known values of u into the constraint values $L(u)$. To shed some more light on our approach we proceed to dissect formulation (1):

- *State snaphot u_D.* The target u_D may be the available corrupted application state or a compressed representation thereof. For the latter, the use of conventional lossless and lossy compression, including file and image compression [8], and new ideas from super-resolution imaging [6] are fully compatible with (1). A more powerful approach is to solve a separate optimization problem for lossy compression, similar to (1), where we match the available state data to a sparse state representation or a reduced-basis model, subject to physical constraints. It is also possible to combine the two approaches and develop broadly applicable algorithms for constrained data compression. This subject will be studied in a future publication. In the following sections, we focus on the case where u_D is the available corrupted (uncompressed) application state.

- *Constraints* $L(u)$*.* The definition of the constraints requires an in-depth analysis of the application. In general, we must incorporate the available checkpoint data and additional application-dependent physical properties. In the context of conservation laws governing transport, the latter may be local extrema, mass/momentum balance and locations of discontinuities.
- *Objective function* $\mathcal{J}(C(u) - u_D)$*.* The choice of the objective function is primarily governed by numerical accuracy considerations; stability can be imparted through the constraints. A very important task is to analyze the accuracy of a numerical scheme subject to lossy compression and recovery. This subject will be studied in a future publication.

The remainder of the paper is organized as follows. In Section 2 we define a model problem motivated by scalar transport. The problem definition is accompanied by a discussion of the physical properties that are to be preserved by our compression/recovery schemes. In Section 3 we apply the abstract OB formulation (1) to the scalar transport problem. The numerical solution of the resulting optimization problem is showcased in the context of three simulated failure scenarios. In Section 4 we discuss our conclusions and point to some future work.

2 MODEL PROBLEM

We assume that the computational domain Ω is a simply-connected, bounded open domain in $\mathbb{R}^2$ or $\mathbb{R}^3$, with a Lipschitz-continuous boundary $\Gamma = \partial\Omega$, and consider the scalar transport equation

$$\left\{\begin{array}{rcll} \dfrac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) & = & 0 & \text{on } \Omega \times [0,T], \\ \rho(\boldsymbol{x},t) & = & \rho_b(\boldsymbol{x},t) & \text{on } \Gamma_{\mathrm{I}} \times [0,T], \\ \rho(\boldsymbol{x},0) & = & \rho^0(\boldsymbol{x}) & \text{on } \Omega. \end{array}\right. \tag{2}$$

In this equation $\mathbf{v}(\boldsymbol{x})$ is a solenoidal velocity field, $\rho = \rho(\boldsymbol{x},t)$ is a non-negative density function defined on $\Omega \times [0,T]$, $T > 0$ is the final time, $\rho^0(\boldsymbol{x}) \geq 0$ is a piecewise C^0 initial density distribution, and $\rho_b(\boldsymbol{x},t)$ is a given inflow boundary data. We assume that the domain Ω and the velocity field are such that the *inflow* boundary $\Gamma_{\mathrm{I}} = \{\boldsymbol{x} \in \Gamma \,|\, \boldsymbol{n}(\boldsymbol{x}) \cdot \mathbf{v}(\boldsymbol{x}) < 0\} \subset \Gamma$ is non-empty. The complement of Γ_{I} is the *outflow* boundary $\Gamma_{\mathrm{O}} = \Gamma \setminus \Gamma_{\mathrm{I}}$.

Equation (2) provides a simple, yet sufficiently representative setting for our purposes, which will allow us to present the key ideas of optimization-based (OB) compression and recovery without unnecessary technical distractions. In particular, we note that solutions of (2) have constant upper and lower bounds in any given Lagrangian volume $V(t)$, that is for all $0 < t \leq T$ there holds

$$\rho^V_{min} \leq \rho(\boldsymbol{x}, t) \leq \rho^V_{max} \qquad \forall \boldsymbol{x} \in V(t)\,, \tag{3}$$

where

$$\rho^V_{\min} = \min_{\boldsymbol{x}\in V(0)} \rho^0(\boldsymbol{x}) \quad \text{and} \quad \rho^V_{\max} = \max_{\boldsymbol{x}\in V(0)} \rho^0(\boldsymbol{x})\,.$$

The *local solution bounds* (3) define the first physical property that we shall aim to preserve in the compression/recovery process. The second physical property will be the total mass preservation in the given region Ω, i.e., we shall restrict attention to solutions of (2) for which

$$\frac{d}{dt}\int_\Omega \rho(\boldsymbol{x}, t)\, d\boldsymbol{x} = 0\,. \tag{4}$$

2.1 Discretization of the model problem

In this section we briefly discus the finite element discretization of (2) and define discrete analogues of the two physical properties, i.e., the local solution bounds (3) and the total mass preservation (4).

Let Ω^h be a shape-regular partition of Ω into finite elements K_n with an average size h. We discretize (2) in space using the classical SUPG scheme [2] implemented with standard linear, bilinear or trilinear C^0 nodal finite elements defined with respect to the mesh Ω^h. We discretize the resulting semi-discrete equations in time using the θ scheme. Thus, the fully discrete problem is given by the following system of linear algebraic equations

$$(M - \Delta t\theta K)\boldsymbol{\rho}^{k+1} = (M + \Delta t(1-\theta)K)\boldsymbol{\rho}^k \qquad k = 1, ..., N, \tag{5}$$

where $\boldsymbol{\rho}^k \in \mathbb{R}^n$ is a vector of nodal solution coefficients, $M \in \mathbb{R}^{n\times n}$ is an SUPG-stabilized mass matrix, $K \in \mathbb{R}^{n\times n}$ is an SUPG-stabilized advection matrix, Δt is the time step, N is the total number of time steps and $\boldsymbol{\rho}^0 \in \mathbb{R}^n$ is a vector of nodal coefficients for the initial data.

Discrete physical properties. To define a discrete analogue of (3) for the fully discrete equation (5) we note that for the finite element spaces used in this work local bounds on the nodal coefficients imply pointwise solution bounds on the finite element approximation. As a result, an appropriate discrete version of (3) is given by

$$(\boldsymbol{\rho}^{k+1}_{\min})_i \leq \boldsymbol{\rho}^{k+1}_i \leq (\boldsymbol{\rho}^{k+1}_{\max})_i\,, \tag{6}$$

where $\boldsymbol{\rho}^{k+1}_{\min} \in \mathbb{R}^n$ and $\boldsymbol{\rho}^{k+1}_{\max} \in \mathbb{R}^n$ are physically motivated lower and upper bounds for the nodal values at t_{k+1}. Here we define these bounds according to

$$(\boldsymbol{\rho}^{k+1}_{\min})_i = \min_{j\in\widetilde{\mathcal{N}}(\boldsymbol{x}_i)} (\boldsymbol{\rho}^k)_j, \quad \text{and} \quad (\boldsymbol{\rho}^{k+1}_{\max})_i = \max_{j\in\widetilde{\mathcal{N}}(\boldsymbol{x}_i)} (\boldsymbol{\rho}^k)_j, \tag{7}$$

where $\widetilde{\mathcal{N}}(\boldsymbol{x}_i)$ is the set of nearest neighbor nodes for a given mesh node $\boldsymbol{x}_i$. This definition guarantees that the finite element solution satisfies a discrete maximum principle [9]. We also note that (6) automatically implies the global solution bound $\mathbf{0} \leq \boldsymbol{\rho}^k \leq \mathbf{1}$, $\forall k$, where we have assumed $\mathbf{0} \leq \rho^0 \leq \mathbf{1}$.

A discrete version of the second property (4) is straightforward. Given a discrete solution $\boldsymbol{\rho}^k$ at $t = t_k$ we require that the solution $\boldsymbol{\rho}^{k+1}$ at the next time step has the same total mass, i.e.,

$$\mathbf{1}^T \widetilde{M} \boldsymbol{\rho}^{k+1} = \mathbf{1}^T \widetilde{M} \boldsymbol{\rho}^k \quad k = 0, 1, \ldots N, \tag{8}$$

where $\widetilde{M} \in \mathbb{R}^{n \times n}$ is a consistent (unstabilized) mass matrix, and $\mathbf{1} \in \mathbb{R}^n$ is a vector of ones. This property is equivalent to a single linear equality constraint given by

$$\mathbf{1}^T(\widetilde{M}\boldsymbol{\rho}^k) = m\,, \forall k\,, \quad \text{where } m \text{is the total initial mass, i.e., } m = \mathbf{1}^T(\widetilde{M}\boldsymbol{\rho}^0)\,. \tag{9}$$

3 FAULT-TOLERANT SCALAR TRANSPORT

In this section we specialize the abstract optimization problem (1) to the discretization (5) of the model equations (2). We briefly discuss the compression and recovery schemes.

Compression. In this example we do not compress the application state directly. Instead, to reduce the checkpoint size we rely on a compression of the vectors of local bounds, used to define the constraints for the recovery scheme. It turns out that in this application a very coarse representation of local bounds is sufficient for robust property-preserving recovery. Specifically, to perform compression we downsample the local bounds uniformly on a mesh with 16 times fewer nodes, and linearly interpolate the values to obtain $\boldsymbol{\rho}^k_{\min}$ and $\boldsymbol{\rho}^k_{\max}$ during the recovery process. We note that in this case the property $\mathbf{0} \leq \boldsymbol{\rho}^k \leq \mathbf{1}$ still holds. Therefore, this is an example —albeit a very simple one— of property-preserving compression.

Recovery. The corruption of the application state $\boldsymbol{\rho}^k$ is assumed to happen through various scenarios, specified later. We denote the corrupted application snapshot by $\boldsymbol{\rho}^k_D \in \mathbb{R}^n$. In this case the state-to-snapshot map C introduced in (1) is the identity, and $n = d$. We define the objective function as the square of the ℓ^2-norm misfit between the desired 'reconstruction' $\boldsymbol{\rho}^k$ and the snapshot $\boldsymbol{\rho}^k_D$. The property-preserving recovery problem for scalar transport, labeled FPR-T, takes the following form:

$$\underset{\boldsymbol{\rho}^k \in \mathbb{R}^n}{\text{minimize}} \; \|\boldsymbol{\rho}^k - \boldsymbol{\rho}^k_D\|_2^2 \quad \text{subject to} \quad \boldsymbol{\rho}^k_{\min} \leq \boldsymbol{\rho}^k \leq \boldsymbol{\rho}^k_{\max} \;\text{ and }\; \mathbf{1}^T(\widetilde{M}\boldsymbol{\rho}^k) = m\,. \tag{10}$$

We recognize this problem as a singly linearly constrained quadratic program with simple bounds, and use the optimization algorithm developed in [1] for its solution. This algorithm is extremely efficient, amounting to purely local median computations followed by a few dot products. We also note the in the case of local failure we need not consider the full application state $\boldsymbol{\rho}^k \in \mathbb{R}^n$, but may instead solve smaller, local optimization problems.

To demonstrate the power of property-preserving recovery for lossy scalar transport we conduct numerical experiments based on three failure models: large yet infrequent failure; small yet frequent failure; and large and frequent failure. The failure models are described later. We take conventional SUPG with the forward-Euler time integration, i.e., (5) with $\theta = 0$ as our baseline. This scheme is not designed to handle failure, nor does it preserve the local solution bounds (6). Thus, we expect the SUPG state obtained through (5) to be unphysical; see Figure 1.

To define the second scheme we augment SUPG with FPR-T. Specifically, at every time step we run conventional SUPG, obtain a (corrupted) state, and use this state as the target snapshot $\boldsymbol{\rho}_D^k$ in (10). To recover the property-preserving state we solve (10). For brevity we overload the notation and call this hybrid scheme FPR-T. To summarize, the schemes are:

1. SUPG: Compute $\boldsymbol{\rho}^k$ by solving $M\boldsymbol{\rho}^k = (\Delta t K + M)\boldsymbol{\rho}^{k-1}$.
2. FPR-T: Compute $\boldsymbol{\rho}^k$ by solving $M\boldsymbol{\rho}^k = (\Delta t K + M)\boldsymbol{\rho}^{k-1}$. Set $\boldsymbol{\rho}_D^k = \boldsymbol{\rho}^k$. Solve (10) for the new $\boldsymbol{\rho}^k$.

The underlying numerical example is the solid body rotation defined by LeVeque [10, p.650]. Here an initial 'combo' density, comprising a smooth hump, a cone and a slotted cylinder, see Figure 1, is transported by the rotational velocity field $\mathbf{v} = (-x_2 + 1/2,\ x_1 - 1/2)$. We report results after one full revolution.

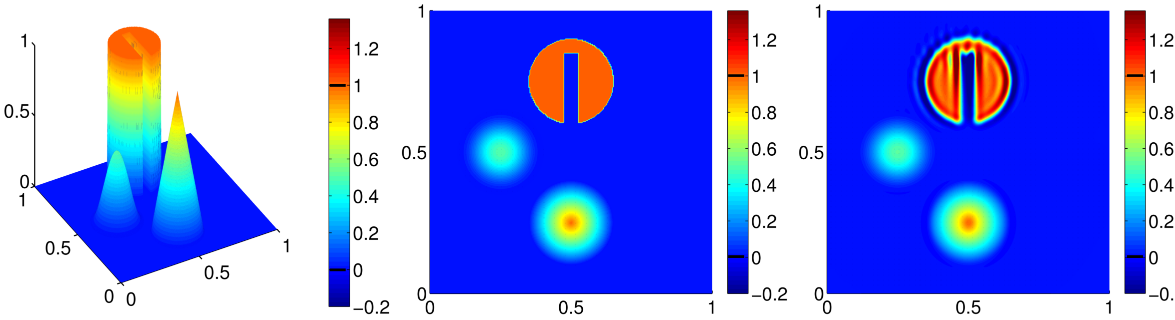

Figure 1: *Left to right:* (1) The 'combo' initial data for the solid body rotation tests [10]. (2) The plane view of the initial data. The initial physical bounds are Max=1.00 and Min=0.00. Fixed color scale. (3) SUPG solution after one full rotation. SUPG solution violates physical bounds (dark areas); Max=1.83, Min=−0.78.

Large yet infrequent failure. We simulate the failure of a system where an entire group of spatially linked compute nodes is the weak link. The application state is corrupted three times in the course of the simulation, at iterations $\lfloor N/4 \rfloor$, $\lfloor N/2 \rfloor$ and $\lfloor 3N/4 \rfloor$, in the same diamond-shaped region centered at $(0.3, 0.4)$. The value of the corrupted application state $\boldsymbol{\rho}_D^k$ is set to -1 in the failed compute region, where no additional state data is stored. As described above, we use a coarse representation of the local bounds, and mass conservation, to complete the FPR-T formulation. The results using conventional SUPG where no recovery is attempted and using FPR-T are shown in Figure 2. Conventional SUPG clearly exhibits imprinting of the failed compute region, while FPR-T appears robust to failure. We note that the effective checkpoint compression, as studied by Ibtesham et. al. [8] (with 0% denoting no compression), is 87.5%. This is due to (i) local recovery and (ii) storing only the coarse bounds and total mass, i.e., one eighth of the total state information and a single scalar, respectively.

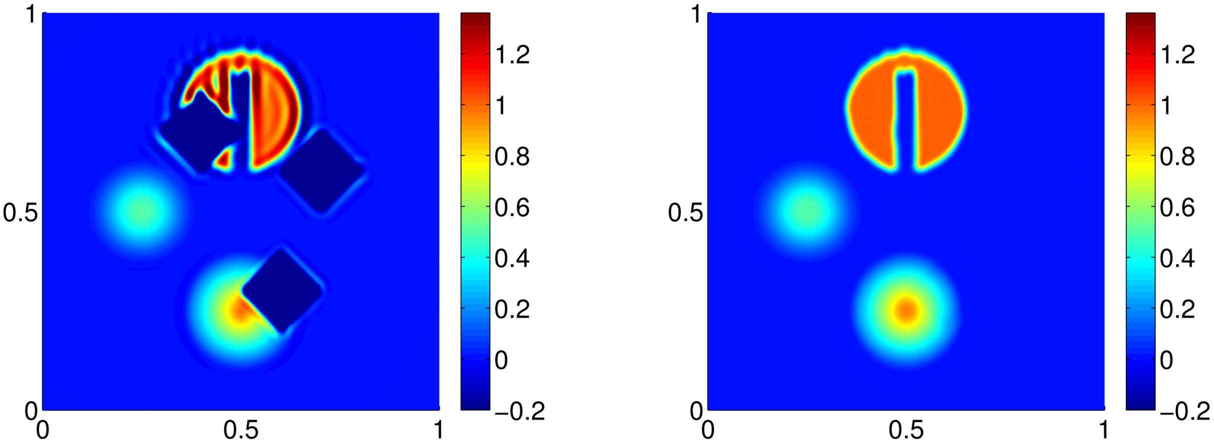

Figure 2: Large-yet-infrequent system failure for SUPG (left) and our FPR-T formulation (right). SUPG exhibits imprinting of the failed region; FPR-T is robust to failure with 87.5% compression ratio.

Small yet frequent failure. Here we study the effect of random node failures. As for the case of large yet infrequent failures, to define the constraints we use a coarse representation of the local bounds, and mass conservation. The value of the corrupted application state $\boldsymbol{\rho}_D^k$ is set to -1 at a randomly selected compute node at every tenth forward-Euler iteration. It is no surprise that conventional SUPG cannot cope with random node failure, see Figure 3. On the other hand, our FPR-T scheme, applied locally, reconstructs the application state almost exactly. Again, taking into account the added storage, the effective checkpoint compression is 87.5%. Finally, we note that for both failure cases (large-yet-infrequent and small-yet-frequent) we observed an L^2 convergence rate of the error between the initial data and the final state of approximately 1/4, on the mesh sequence $\{64\times 64,\ 128\times 128,\ 256\times 256\}$. This is consistent with the convergence rate (without faults) of a related optimization-based transport algorithm [1], which uses formulation similar to (10) but with target $\boldsymbol{\rho}_D^k$ given by a cell-centered finite volume approximation of the density.

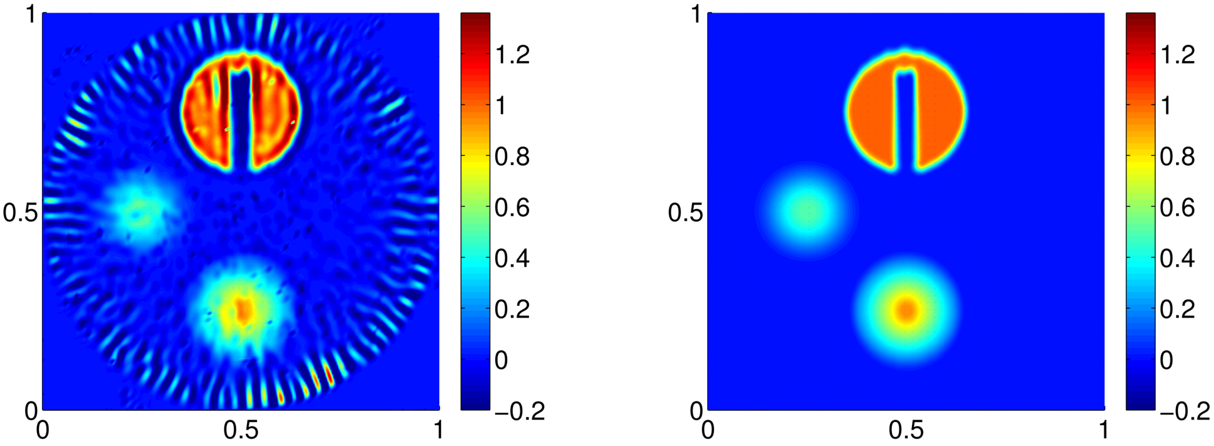

Figure 3: Small-yet-frequent system failure for SUPG (left) and our FPR-T formulation (right). SUPG cannot cope with random data corruption; FPR-T is robust to failure.

Large and frequent failure. This case is very similar to the first, except for an increased frequency of failure. The application state is corrupted at every 200th iteration, amounting to 63 large-scale failures. As there are no recovery mechanisms in conventional SUPG, an entire annulus is corrupted, see Figure 4. Our FPR-T scheme clearly does not fair as well as before, however, that is expected. This points to the need to develop alternate definitions of application data $\boldsymbol{\rho}_D^k$, where some state information (and not simply bounds) are incorporated into the FPR-T formulation. Nonetheless, we observe that at final time $\mathbf{0} \le \boldsymbol{\rho}^N \le \mathbf{1}$ and $|\mathbf{1}^T(\widetilde{M}\boldsymbol{\rho}^N) - \mathbf{1}^T(\widetilde{M}\boldsymbol{\rho}^0)| \approx 10^{-14}$, i.e., the global bounds and the total mass are conserved! Moreover, we cannot emphasize enough that for all three examples the overhead of solving the property-preserving recovery problem (10) amounts to only a few percent of the total computational time. Lastly, it is evident that the proposed formulation can be used not only for physics-based checkpointing but as a fault-tolerant transport scheme in its own right, with the added benefits of mass and monotonicity preservation.

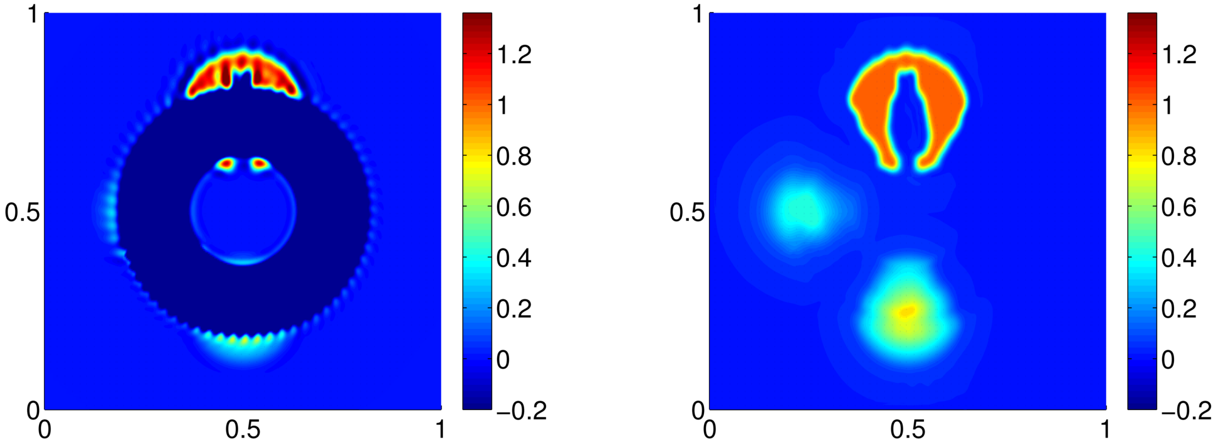

Figure 4: Large-and-frequent system failure for SUPG (left) and our FPR-T formulation (right). SUPG exhibits a corrupted annulus; FPR-T fairs significantly better.

4 CONCLUSIONS

We have formulated a new optimization-based approach for "smart" compression and recovery for fault-tolerant high-performance computing. Our approach implements a divide-and-conquer strategy, which separates compression and recovery from the preservation of the desired physical properties. This allows one to combine the approach with virtually any available compression and/or recovery scheme. In this paper we have focused on demonstrating the approach by using it to define a fault recovery procedure for scalar transport, which preserves local solution bounds and total mass. Numerical tests simulating different failure modes demonstrate the ability of the optimization-based approach to recover a reasonable application state even under severe failure, while always preserving the desired physical properties. In future publications we will analyze the numerical accuracy of the optimization-based approach for specific compression and recovery schemes. Additionally, we plan to explore a variety of new property-preserving compression algorithms.

Acknowledgment

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research.

REFERENCES

[1] Pavel Bochev, Denis Ridzal, and Mikhail Shashkov. Fast optimization-based conservative remap of scalar fields through aggregate mass transfer. *Journal of Computational Physics*, 246(0):37 – 57, 2013.

[2] A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. *Comp. Meth. Appl. Mech. Engng.*, 32:199–259, 1982.

[3] Franck Cappello. Fault tolerance in petascale/ exascale systems: Current knowledge, challenges and research opportunities. *The International Journal of High Performance Computing Applications*, 23(3):212–226, 2009.

[4] P. W. Coteus, S. A. Hall, T. Takken, R. A. Rand, S. Tian, G. V. Kopcsay, R. Bickford, F. P. Giordano, C. M. Marroquin, and M. J. Jeanson. Packaging the IBM Blue Gene/Q supercomputer. *IBM Journal of Research and Development*, 57(1/2):2:1–2:13, Jan 2013.

[5] D. W. Dauwe, S. Pasricha, A. A. Maciejewski, and H. J. Siegel. Resilience-aware resource management for exascale computing systems. *IEEE Transactions on Sustainable Computing*, PP(99):1–1, 2018.

[6] S. Farsiu, M.D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super resolution. *Image Processing, IEEE Transactions on*, 13(10):1327–1344, Oct 2004.

[7] M. A. Heroux. Toward resilient algorithms and applications. Technical report, arXiv:1402.3809, 2014.

[8] D. Ibtesham, D. Arnold, K.B. Ferreira, and P.G. Bridges. On the viability of checkpoint compression for extreme scale fault tolerance. In *Euro-Par 2011*, LNCS (7156), pages 302–311. Springer, 2012.

[9] D. Kuzmin, R. Löhner, and S. Turek, editors. *Flux-Corrected Transport. Principles, Algorithms and Applications.* Springer Verlag, Berlin, Heidelberg, 2005.

[10] Randall J. LeVeque. High-resolution conservative algorithms for advection in incompressible flow. *SIAM Journal on Numerical Analysis*, 33(2):627–665, 1996.

[11] B. Schroeder and G.A. Gibson. Understanding failures in petascale computers. *Journal of Physics: Conference Series*, 78:012022, 2007.

[12] Devesh Tiwari, Saurabh Gupta, George Gallarno, Jim Rogers, and Don Maxwell. Reliability Lessons Learned from GPU Experience with the Titan Supercomputer at Oak Ridge Leadership Computing Facility. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, pages 38:1–38:12, New York, NY, USA, 2015. ACM.

[13] Devesh Tiwari, Saurabh Gupta, James H. Rogers, Don Maxwell, Paolo Rech, Sudharshan S. Vazhkudai, Daniel A. G. de Oliveira, Dave Londo, Nathan DeBardeleben, Philippe Olivier Alexandre Navaux, Luigi Carro, and Arthur S. Bland. Understanding GPU errors on large-scale HPC systems and the implications for system design and operation. *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 331–342, 2015.