Large-Scale Optimization-Based Non-negative Computational Framework for Diffusion Equations: Parallel Implementation and Performance Studies

J. Chang · S. Karra · K. B. Nakshatrala

Received: 1 July 2015 / Revised: 27 May 2016 / Accepted: 15 July 2016 / Published online: 26 July 2016

Abstract It is well-known that the standard Galerkin formulation, which is often the formulation of choice under the finite element method for solving self-adjoint diffusion equations, does not meet maximum principles and the non-negative constraint for anisotropic diffusion equations. Recently, optimization-based methodologies that satisfy maximum principles and the non-negative constraint for steady-state and transient diffusion-type equations have been proposed. To date, these methodologies have been tested only on small-scale academic problems. The purpose of this paper is to systematically study the performance of the non-negative methodology in the context of high performance computing (HPC). PETSc and TAO libraries are, respectively, used for the parallel environment and optimization solvers. For large-scale problems, it is important for computational scientists to understand the computational performance of current algorithms available in these scientific libraries. The numerical experiments are conducted on the state-of-the-art HPC systems, and a single-core performance model is used to better characterize the efficiency of the solvers. Our studies indicate that the proposed non-negative computational framework for diffusion-type equations exhibits excellent strong scaling for real-world large-scale problems.

Graphical Abstract This figure shows the fate of chromium after 180 days using the single-field Galerkin formulation. The white regions indicate the violation of the non-negative constraint.

✉ K. B. Nakshatrala knakshatrala@uh.edu http://www.cive.uh.edu/faculty/nakshatrala

1 Department of Civil and Environmental Engineering, University of Houston, Texas, TX 77204-4003, USA

2 Los Alamos National Laboratory, Los Alamos, NM, USA
Keywords  High performance computing · Anisotropic diffusion · Maximum principles · Non-negative constraint · Large-scale optimization

1 Introduction

The modeling of flow and transport in subsurface is vital for energy, climate and environmental applications. Examples include CO$_2$ migration in carbon-dioxide sequestration, enhanced geothermal systems, oil and gas production, radio-nuclide transport in a nuclear waste repository, groundwater contamination, and thermo-hydrology in the Arctic permafrost due to the recent climate change [12,16,18,25]. Several numerical codes (e.g., FEHM [45], TOUGH [40], PFLOTRAN [24]) have been developed to model flow and transport in subsurface at reservoir-scale. These codes typically solve unsteady Darcy equations for flow and advection–diffusion equation for transport. The predictive capability of a numerical simulator depends on the robustness of the underlying numerical methods. A necessary and essential requirement is to satisfy important mathematical principles and physical constraints. One such property in transport and reactive-transport problems is that the concentration of a chemical species cannot be negative. Mathematically, this translates to the satisfaction of the discrete maximum principle (DMP) for diffusion-type equations. Subsurface flow and transport applications typically encounter geological media that are highly heterogeneous and anisotropic in nature, and it is well-known that the classical finite element (or finite volume and finite difference, for that matter) formulations do not produce non-negative solutions on arbitrary meshes for such porous media [7,26,35,38].

Several studies over the years have focused on the development of methodologies that enforce the DMP and ensure non-negative solutions [35,36,38,39]. However, these studies did not address how these methods can be used for realistic large-scale subsurface problems that have millions of grid nodes. Furthermore, complex coupling between different physical processes as well as the presence of multiple species amplify the degrees-of-freedom (i.e., the number of unknowns). The aim of this paper is to develop a parallel computational framework that solves anisotropic diffusion equations on general meshes, ensures non-negative solutions, and can be employed to solve large-scale realistic problems.
Large-scale problems can be tackled by using recent advancements in high-performance computing (HPC) methods and toolkits that can be used on the state-of-the-art supercomputing architecture. One such toolkit is PETSc [3], which provides data structures and subroutines for setting up structured and unstructured grids, parallel communication, linear and non-linear solvers, and parallel I/O. These high-level data structures and subroutines help in faster development of parallel application codes and minimize the need to program low-level message passing, so that the domain scientists can focus more on the application. To this end, we develop a non-negative parallel framework by leveraging the existing capabilities within PETSc. Our framework ensures the DMP for anisotropic diffusion by using lower-order finite elements and the optimization-based approach in [26,35,38]. The TAO toolkit [34], which is built on top of PETSc, has a wide suite of commonly-used optimization solvers. The robustness of the proposed framework will be demonstrated by solving realistic large-scale problems.

It needs to be mentioned that there are other possible ways of solving the optimization problem at hand in large-scale setting, and a small discussion is warranted on this aspect. Herein, we pose the problem as a quadratic programming and employ tools that are primarily developed to handle such optimization problems. However, one can also pose the problem as a variational inequality and in particular as an obstacle problem [19]. A thorough discussion on the mathematical aspects of variational inequalities can be found in [9,19]. The state-of-the-art computational strategies for solving large-scale variational inequalities can be found in [4,42], and a discussion on the associated linear solvers and preconditioners can be found in [10]. A systematic study of the problem at hand using variational inequalities is beyond the scope of this paper and will be addressed in a sequel paper.

The rest of this paper is organized as follows. In Sect. 2, we present the governing equations and the classical single-field Galerkin finite element formulation for steady-state and transient diffusion equations. The optimization-based method to ensure non-negative concentrations is also outlined in this section. In Sect. 3, the parallel implementation procedure using PETSc and TAO is presented. We also highlight the relevant data structures used in this study and present a pseudo algorithm describing our parallel framework. In Sect. 4, a performance model loosely based on the roofline model is outlined. This model is used to estimated the efficiency with respect to computing hardware of currently available solvers within PETSc and TAO. In Sect. 5, we first verify our implementation using a 3D benchmark problem from the literature and present a detailed performance study using the proposed model. Then, we study a large-scale three-dimensional realistic problem involving the transport of chromium in the subsurface and document the numerical results of the non-negative methodology with the classical single-field Galerkin formulation. Conclusions are drawn in Sect. 6.

We shall denote all the continuum vectors by lowercase boldface unitalicized letters, and the vectors in the discrete setting are denoted by lowercase boldface italic letters. We shall denote all the continuum second-order tensors by boldface uppercase unitalicized letters, and all the finite element matrices are denoted by uppercase boldface italicized letters. Other notational conventions are introduced as needed in the paper.

2 Governing Equations and Associated Non-negative Numerical Methodologies

Let \( \Omega \subset \mathbb{R}^{nd} \) be a bounded open domain, where “\( nd \)” is the number of spatial dimensions. The boundary of the domain is denoted by \( \partial \Omega = \overline{\Omega} - \Omega \), which is assumed to be piecewise
smooth. A spatial point is denoted by \( x \in \Omega \). The gradient and divergence operators with respect to \( x \) are, respectively, denoted as \( \text{grad}[] \) and \( \text{div}[] \). As usual, the boundary is divided into two parts: \( \Gamma^D \) and \( \Gamma^N \). \( \Gamma^D \) is that part of the boundary on which Dirichlet boundary conditions are prescribed, and \( \Gamma^N \) is the part of the boundary on which Neumann boundary conditions are prescribed. For mathematical well-posedness, we assume \( \Gamma^D \cup \Gamma^N = \partial\Omega \) and \( \Gamma^D \cap \Gamma^N = \emptyset \). The unit outward normal to boundary is denoted as \( \hat{n}(x) \). The diffusivity tensor is denoted by \( D(x) \), which is assumed to symmetric, bounded above and uniformly elliptic. That is,

\[
D(x) = D^T(x) \quad \forall x \in \Omega
\]

and there exists two constants \( 0 < \xi_1 \leq \xi_2 < +\infty \) such that

\[
\xi_1 y^T y \leq y^T D(x) y \leq \xi_2 y^T y \quad \forall x \in \Omega \text{ and } \forall y \in \mathbb{R}^{nd}
\]

2.1 Governing Equations for Steady-State Response

We shall denote the steady-state concentration field by \( c(x) \). The governing equations can be written as follows:

\[\begin{align*}
-\text{div}[D(x)\text{grad}[c]] &= f(x) \quad \text{in } \Omega \\
c(x) &= c^p(x) \quad \text{on } \Gamma^D \\
-\hat{n}(x) \cdot D(x)\text{grad}[c] &= q^p(x) \quad \text{on } \Gamma^N
\end{align*}\] (2.3a, 2.3b, 2.3c)

where \( f(x) \) is the volumetric source/sink, \( c^p(x) \) is the prescribed concentration, and \( q^p(x) \) is the prescribed flux. For uniqueness, we assume \( \Gamma^D \neq \emptyset \).

2.1.1 Maximum Principle and the Non-negative Constraint

The above boundary value problem is a self-adjoint second-order elliptic partial differential equation (PDE). It is well-known that such PDEs possess an important mathematical property—the classical maximum principle [8]. The mathematical statement of the classical maximum principle can be written as follows: If \( c(x) \in C^2(\Omega) \cap C^0(\overline{\Omega}) \), \( \partial\Omega = \Gamma^D \), and \( f(x) \leq 0 \) in \( \Omega \) then

\[
\max_{x \in \overline{\Omega}} c(x) = \max_{x \in \partial\Omega} c^p(x)
\]

(2.4)

Similarly, if \( f(x) \geq 0 \) in \( \Omega \) then

\[
\min_{x \in \overline{\Omega}} c(x) = \min_{x \in \partial\Omega} c^p(x)
\]

(2.5)

To make our presentation on maximum principles simple, we have assumed stronger regularity on the solution (i.e., \( c(x) \in C^2(\Omega) \cap C^0(\overline{\Omega}) \)), and assumed that Dirichlet boundary conditions are prescribed on the entire boundary. However, maximum principles requiring milder regularity conditions on the solution, even for the case when Neumann boundary conditions are prescribed on the boundary, can be found in literature (see [32, 33]).

If \( f(x) \geq 0 \) in \( \Omega \) and \( c^p(x) \geq 0 \) on the entire \( \partial\Omega \) then the maximum principle implies that \( c(x) \geq 0 \) in the entire domain, which is the non-negativity of the concentration field.

Springer
2.1.2 Single-Field Galerkin Weak Formulation

The following function spaces will be used in the rest of this paper:

\[ \mathcal{U} := \{ c(x) \in H^1(\Omega) \mid c(x) = c^p(x) \text{ on } \Gamma^D \} \]  
\[ \mathcal{W} := \{ w(x) \in H^1(\Omega) \mid w(x) = 0 \text{ on } \Gamma^D \} \]

where \( H^1(\Omega) \) is a standard Sobolev space \([1]\). The single-field Galerkin weak formulation corresponding to equations (2.3a)–(2.3c) reads: Find \( c(x) \in \mathcal{U} \) such that we have

\[ B(w; c) = L(w) \quad \forall w(x) \in \mathcal{W} \]  

where the bilinear form and linear functional are, respectively, defined as

\[ B(w; c) := \int_\Omega \text{grad}[w(x)] \cdot D(x) \text{grad}[c(x)] \, d\Omega \]  
\[ L(w) := \int_\Omega w(x) f(x) \, d\Omega + \int_{\Gamma^N} w(x) q^p(x) \, d\Gamma \]

Since \( D(x) \) is symmetric, by Vainberg’s theorem \([15]\), the single-field Galerkin weak formulation given by Eq. (2.8) is equivalent to the following variational problem:

\[ \min_{c(x) \in \mathcal{U}} \frac{1}{2} B(c; c) - L(c) \]  

2.1.3 A Methodology to Enforce the Maximum Principle for Steady-State Problems

Our methodology is based on the finite element method. We decompose the domain into “Ne le” non-overlapping open element sub-domains such that

\[ \Omega = \bigcup_{e=1}^{Ne le} \Omega^e \]

(Recall that a superposed bar denotes the set closure.) The boundary of \( \Omega^e \) is denoted by \( \partial \Omega^e := \overline{\Omega^e} - \Omega^e \). We shall define the following finite dimensional vector spaces of \( \mathcal{U} \) and \( \mathcal{W} \):

\[ \mathcal{U}^h := \{ c^h(x) \in \mathcal{U} \mid c^h(x) \in C^0(\overline{\Omega}), c^h(x) \mid_{\Omega^e} \in P^k(\Omega^e), e = 1, \ldots, Ne le \} \]  
\[ \mathcal{W}^h := \{ w^h(x) \in \mathcal{W} \mid w^h(x) \in C^0(\overline{\Omega}), w^h(x) \mid_{\Omega^e} \in P^k(\Omega^e), e = 1, \ldots, Ne le \} \]

where \( k \) is a non-negative integer, and \( P^k(\Omega^e) \) denotes the linear vector space spanned by polynomials up to \( k \)-th order defined on the sub-domain \( \Omega^e \). The finite element formulation for Eq. (2.8) can be written as: Find \( c^h(x) \in \mathcal{U}^h \) such that we have

\[ B(q^h; c^h) = L(q^h) \quad \forall q^h(x) \in \mathcal{Q}^h \]

It has been documented in the literature that the above finite element formulation violates the maximum principle and the non-negative constraint \([26,35,38]\).

We now outline an optimization-based methodology that satisfies the maximum principle and the non-negative constraint on general computational grids. To this end, we shall use the
symbols $\leq$ and $\geq$ to denote component-wise inequalities for vectors. That is, for given any two vectors $a$ and $b$

$$a \leq b \quad \text{means that} \quad a_i \leq b_i \quad \forall i$$ (2.14)

The symbol $\geq$ can be similarly defined as well. Let $\langle \cdot; \cdot \rangle$ denote the standard inner-product in Euclidean space. After finite element discretization, the discrete equations corresponding to Eq. (2.13) take the form

$$Kc = f$$ (2.15)

where $K$ is a symmetric positive definite matrix, $c$ is the vector containing nodal concentrations, and $f$ is the force vector. Equation (2.15) is equivalent to the following minimization problem

$$\min_{c \in \mathbb{R}^{ndofs}} \frac{1}{2} \langle c; Kc \rangle - \langle c; f \rangle$$ (2.16)

where “$ndofs$” denotes the number of degrees of freedom for concentration. Equation (2.15) can lead to unphysical negative solutions.

Following [26,35], a methodology corresponding to Eq. (2.16) that satisfies the non-negative constraint can be written as follows:

$$\min_{c \in \mathbb{R}^{ndofs}} \frac{1}{2} \langle c; Kc \rangle - \langle c; f \rangle$$ (2.17a)

subject to

$$0 \preceq c$$ (2.17b)

where $0$ is a vector of size $ndofs$ containing zeros. Since $K$ is positive definite, Eq. (2.17) has a unique global minimum [5]. Several robust numerical methods can be used to solve Eq. (2.17), which include active set strategy, interior point methods [5]. In this paper, to solve the resulting optimization problems, we shall use the parallel optimization toolkit TAO [34], which has the active-set Newton trust region (TRON) and quasi-Newton-based bounded limited memory variable metric (BLMVM) algorithms.

### 2.2 Governing Equations for Transient Response

We shall denote the time by $t \in [0, I]$, where $I$ denotes the length of the time interval of interest. We shall denote the time-dependent concentration by $c(x, t)$. The initial boundary value problem can be written as follows:

$$\frac{\partial c}{\partial t} = \text{div}[D(x)\text{grad}[c]] + f(x, t) \quad \text{in } \Omega \times (0, I)$$ (2.18a)

$$c(x, t) = c^p(x, t) \quad \text{on } \Gamma^D \times (0, I)$$ (2.18b)

$$-\hat{n}(x) \cdot D(x)\text{grad}[c] = q^p(x, t) \quad \text{on } \Gamma^N \times (0, I)$$ (2.18c)

$$c(x, 0) = c_0(x) \quad \text{in } \Omega$$ (2.18d)

where $c_0(x)$ is the prescribed initial concentration, $f(x, t)$ is the time-dependent volumetric source/sink, $c^p(x, t)$ is the time-dependent prescribed concentration on the boundary, and $q^p(x, t)$ is the prescribed time-dependent flux on the boundary.
2.2.1 Maximum Principle and the Non-negative Constraint

The maximum principle of a transient diffusion equation asserts that the maximum can occur only on the boundary of the domain or in the initial condition if \( f(x, t) \leq 0 \) and \( \Gamma^D = \partial \Omega \). Mathematically, a solution to Eqs. (2.18a)–(2.18d) will satisfy:

\[
c(x, t) \leq \max \left[ \max_{x \in \Omega} c_0(x), \max_{x \in \partial \Omega} c^p(x, t) \right] \quad \forall t \tag{2.19}
\]

provided \( f(x, t) \leq 0 \). Similarly, the minimum will occur either on the boundary or in the initial condition if \( f(x, t) \geq 0 \). That is, if \( f(x, t) \geq 0 \) then a solution to Eqs. (2.18a)–(2.18a) satisfies:

\[
c(x, t) \geq \min \left[ \min_{x \in \Omega} c_0(x), \min_{x \in \partial \Omega} c^p(x, t) \right] \quad \forall t \tag{2.20}
\]

If \( f(x, t) \geq 0 \) in \( \Omega \), \( c^p(x, t) \geq 0 \) on the entire \( \partial \Omega \), and \( c_0(x) \geq 0 \) in \( \Omega \) then the maximum principle implies that \( c(x, t) \geq 0 \) in the entire domain at all times, which is the non-negative constraint for the concentration field for transient problems.

2.2.2 A Methodology to Enforce the Maximum Principle for Transient Problems

We divide the time interval of interest into \( N \) sub-intervals. That is,

\[
[0, T] := \bigcup_{n=0}^{N} [t_n, t_{n+1}] \tag{2.21}
\]

where \( t_n \) denotes the \( n \)-th time-level. We assume that the time-step is uniform, which can be written as:

\[
\Delta t = t_{n+1} - t_n \tag{2.22}
\]

Following the recommendation provided in [37] to meet maximum principles, we employ the backward Euler method for temporal discretization. We shall denote the nodal concentrations at the \( n \)-th time-level by \( c^{(n)} \). We shall denote the minimum and maximum values for the concentration by \( c_{\text{min}} \) and \( c_{\text{max}} \), which will be provided by the maximum principle and the non-negative constraint. At each time-level, one has to solve the following convex quadratic program:

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \left( c^{(n+1)} ; \tilde{K} c^{(n+1)} \right) - \left( c^{(n+1)} ; \tilde{f}^{(n+1)} \right) \\
\text{subject to} & \quad c_{\text{min}} \mathbf{1} \preceq c^{(n+1)} \preceq c_{\text{max}} \mathbf{1} \tag{2.23a-b}
\end{align*}
\]

where

\[
\begin{align*}
\tilde{K} & := \frac{1}{\Delta t} M + K \tag{2.24} \\
\tilde{f}^{(n+1)} & := f^{(n+1)} + \frac{1}{\Delta t} M c^{(n+1)} \tag{2.25}
\end{align*}
\]

In the above equation, \( M \) is the capacity matrix [37].
3 Parallel Implementation

3.1 PETSc and TAO

We leverage on existing scientific libraries such as PETSc and TAO to formulate our large-scale computational framework. PETSc is a suite of data structures and routines for the parallel solution of scientific applications. It also provides interfaces to several other libraries such as Metis/ParMETIS[17] and HDF5 [41] for mesh partitioning and binary data format handling respectively. The Data Management (DM) data structure is used to manage all information including vectors and sparse matrices and compatible with binary data formats. To handle unstructured grids in parallel, a subset of the DM structure called DMPlex (see [3,22,23]), as shown in Fig. 1, uses the direct acyclic graph to organize all mesh information. This topology enables the freedom to mix and match various non vertex-based discretization such as the two-point flux finite volume method and the classical mixed formulations based on the lowest-order Raviart Thomas finite element space.

Another important feature within PETSc is TAO. The TAO library has a suite of data structures and routines that enable the solution of large-scale optimization problems. It can support any data structure or solver within PETSc. Our non-negative methodology will use both the Newton-Trust Region (TRON) and Bounded Limited-Memory Variable-Metric (BLMVM) solvers available within TAO. BLMVM is a quasi-Newton method that uses projected gradients to approximate the Hessian, which is useful for problems where the hessian is too complicated or expensive to compute. Other optimization algorithms such as TRON and the Gradient Projected Conjugate Gradient (GPCG) typically require Hessian information and more memory, but they are expected to converge more rapidly than BLMVM. Further details regarding the implementation of these various methods may be found in [34] and the references within.

3.2 Finite Element Implementation

PETSc abstractions for finite elements, quadrature rules, and function spaces have also been recently introduced and are suitable for the mesh topology within DMPlex. They are built upon the same framework as the Finite element Automatic Tabulator (FIAT) found within the FEniCS Project [20,28,29]. The finite element discretizations simply need the equations, auxiliary coefficients (e.g., permeability, diffusivity, etc.), and boundary conditions specified as point-wise functions. We express all discretizations in nonlinear form so let $r$ and $J$ denote the residual and Jacobian respectively.

Following the FEM model outlined in [21], we consider the weak form that depends on fields and gradients. The residual evaluation can be expressed as:

$$w^T r(c) \sim \int_{\Omega^e} \left[ w \cdot F_0(c, \nabla c) + \nabla w \cdot F_1(c, \nabla c) \right] d\Omega = 0 \quad (3.1)$$

where $F_0(c, \nabla c)$ and $F_1(c, \nabla c)$ are user-defined point-wise functions that capture the problem physics. This framework decouples the problem specification from the mesh and degree of freedom traversal. That is, the scientist need only focus on providing point function evaluations while letting the finite element library take care of meshing, quadrature points, basis function evaluation, and mixed forms if any. The discretization of the residual is written as:

$$r(c) = \sum_{e=1}^{Nele} \begin{bmatrix} N^T & B^T \end{bmatrix} W \begin{bmatrix} F_0(c_q, \nabla c_q) \\ F_1(c_q, \nabla c_q) \end{bmatrix} \quad (3.2)$$
where $A$ represents the standard assembly operator, $N$ and $B$ are matrix forms of basis functions that reduce over quadrature points, $W$ is a diagonal matrix of quadrature weights (including the geometric Jacobian determinant of the element), and $c_q$ is the field value at quadrature point $q$. The Jacobian of (3.2) needs only the derivatives of the point-wise functions:

Fig. 1 Representation of mesh points within the DMPlex data structure and their associated directed acyclic graphs. a Optimal 2D and 3D elements b interpolated 2D and 3D elements
\[ J(c) = \sum_{e=1}^{Nele} N^T B^T W \begin{bmatrix} \mathcal{F}_{0,0} & \mathcal{F}_{0,1} \\ \mathcal{F}_{1,0} & \mathcal{F}_{1,1} \end{bmatrix} \begin{bmatrix} N \\ B \end{bmatrix}, \quad [\mathcal{F}_{i,j}] = \begin{bmatrix} \frac{\partial \mathcal{F}_0}{\partial c} & \frac{\partial \mathcal{F}_0}{\partial \nabla c} \\ \frac{\partial \mathcal{F}_1}{\partial c} & \frac{\partial \mathcal{F}_1}{\partial \nabla c} \end{bmatrix} \begin{bmatrix} c_q, \nabla c_q \end{bmatrix} \] (3.3)

The point-wise functions corresponding to the weak form in (2.9) would be:

\[ \mathcal{F}_0 = -f(x), \quad \mathcal{F}_1 = D(x) \nabla c_q \] (3.4a)

\[ \mathcal{F}_{0,0} = 0, \quad \mathcal{F}_{0,1} = 0, \quad \mathcal{F}_{1,0} = 0, \quad \mathcal{F}_{1,1} = D(x) \] (3.4b)

Similarly, the point-wise functions for the transient response are:

\[ \mathcal{F}_0 = \dot{c}_q - f(x, t), \quad \mathcal{F}_1 = D(x) \nabla c_q \] (3.5a)

\[ \mathcal{F}_{0,0} = \frac{1}{\Delta t}, \quad \mathcal{F}_{0,1} = 0, \quad \mathcal{F}_{1,0} = 0, \quad \mathcal{F}_{1,1} = D(x) \] (3.5b)

where \( \dot{c}_q \) denotes the time derivative. A similar discretization is used to project the Neumann boundary conditions into the residual vector. Assuming a fixed time-step, \([\mathcal{F}_{i,j}]\) and the Jacobian in Eq. (3.3) do not change with time and have to be computed only once. If \( n \) denotes the time level (\( n = 0 \) denotes initial condition) then the residual and Jacobian can be defined as:

\[ r^{(n)} \equiv r(c^{(n)}) \] (3.6)

\[ J \equiv J(c^{(0)}) \] (3.7)

To enforce the non-negative methodology, the following objective function \( b \) and gradient function \( g \) is provided:

Algorithm 1 Pseudocode for the large-scale transport solver

```plaintext
Create/input DAG on rank 0
Create/input cell-wise velocity on rank 0
if size > 1 then
    Partition mesh among all processors
end if
Refine distributed mesh if necessary
Create PetscSection and FE discretization
Set \( n = 0 \) and \( e^{(0)} = 10^{-8} \)
Insert Dirichlet BC constraints into \( e^{(0)} \)
Compute Jacobian \( J \)
while true do
    \( \triangleright \) Begin time-stepping scheme
    if Classical Galerkin then
        \( e^{(n+1)} = e^{(n)} - J\cdot r^{(n)} \) \( \triangleright \) Solve without non-negative methodology
    else
        TaoSolve() for \( e^{(n+1)} \) based on equations (3.8) and (3.9) \( \triangleright \) Solve with non-negative methodology
    end if
    if steady-state or \( n == \) total number of time steps then
        break
    else
        \( n += 1 \)
    end if
end while
```

\( \triangleright \) Springer
\[ b = \frac{1}{2} c^{(n+1)} \cdot J c^{(n+1)} + c^{(n+1)} \cdot \left[ r^{(n)} - J c^{(n)} \right] \]  
(3.8)

\[ g = J \left[ c^{(n+1)} - c^{(n)} \right] + r^{(n)} \]  
(3.9)

BLMVM relies only on the above two equations, whereas TRON needs the Hessian which is equivalent to \( J \). Algorithm 1 outlines the steps taken in our computational framework.

## 4 Performance Modeling

PETSc is a constantly evolving open-source library that brings out new features and algorithms almost every day. It has capabilities to interface with a large number of other open-source software and linear algebra packages. However, it is not always known which of these algorithms will have the best performance across multiple distributed memory HPC systems, especially if these packages have little documentation and have to be used as black-box solvers. Computational scientists would like to know which solvers or algorithms to use for their specific need before running jobs on the state-of-the-art HPC systems. The first and the trivial metric to look for in answering this question is the time-to-solution for a given solver or optimization method. However, additional information is needed in order to quantify the hardware and algorithmic efficiency as well as the potential scalability across multiple cores in the strong sense.

Hardware specifications of HPC systems significantly impact the performance of any numerical algorithm. Ideally we want our simulations to consume as little wall-clock time as possible as the number of processing cores increases (i.e., achieving good speedup), but several other factors including compiler vectorization, cache locality, memory bandwidth, and code implementation may drastically affect the performance. Table 1 lists the hardware specifications of the two HPC systems (Mustang and Wolf) that are used in our numerical experiments. The Mustang HPC system consists of relatively older generation of processors so it is expected to not perform as well. One could simply measure wall-clock time across

| Table 1 | List of HPC systems used in this study |
|-----------------|--------------------------------------|
| Mustang (MU)    | Wolf (WF)                            |
| Processor       | AMD Opteron 6176                     | Intel Xeon E5-2670 |
| Clock rate      | 2.3 GHz                              | 2.6 GHz             |
| FLOPS/cycle     | 4                                    | 8                    |
| Sockets per compute node | 2                                | 2                    |
| NUMA nodes per socket | 2                                | 1                    |
| Cores per socket | 12                                  | 8                    |
| Total cores (compute nodes) | 38400 (1600)  | 9856 (616) |
| Memory per compute node | 64 GB                              | 64 GB                |
| L1 cache per core | 128 KB                              | 32 KB                |
| L2 cache per core | 512 KB                              | 256 KB               |
| L3 cache per socket | 12 MB                              | 20 MB                |
| Interconnect    | 40 Gb/s                             | 40 Gb/s              |
multiple compute nodes on the respective HPC machines and determine the parallel efficiency of a certain algorithm, but we are interested in quantifying how different algorithms behave sequentially and what kind of parallel performance to expect before running numerical simulations on supercomputers. The wall-clock time of any simulation can generally be summed up as a function of three things: the workload, transfer of data between the memory and CPU register, and interprocess communication. Hardware efficiency in this context is defined as the amount of time spent performing work over waiting on memory fetching and cache registers to free up.

The memory bandwidth is the limiting factor of performance for modern computing architectures. That is, the floating point performance (FLOPS/s) of several numerical algorithms will never reach the theoretical peak performance (TPP). This limitation is particularly important for iterative solvers and optimization methods that rely on numerous sparse matrix-vector (SpMV) multiplications (see [30] and the references within). The frequent use of SpMV allows for little cache reuse and will result in a large number of very expensive cache misses. Such behavior is important to document when determining how efficient a scientific code is, so performance models such as the Roofline Model [27,44], which measures memory transfers, have been used to better quantify the efficiency with respect to the hardware. Performance models in general can help application developers identify bottlenecks and indicate which areas of the code can be further optimized. In other words, the code can be designed so that it maximizes the full benefits of the available computing resources. In the next section, we will demonstrate that such models can also be used to predict the parallel efficiency of various optimization solvers on the two very different LANL HPC systems. The key parameter for these performance models is the Arithmetic Intensity (AI) which is defined as:

\[
AI = \frac{\text{Total FLOPS}}{\text{Total Bytes Transferred}} \quad (4.1)
\]

where the Total Bytes Transferred (TBT) metric denotes the amount of bandwidth needed for a given floating point operation. The AI serves as a multiplier to the actual memory bandwidth and creates a “roofline” for the estimation of ideal peak performance. A cache model is needed in order to properly define the TBT.

To this end, we propose a roofline-like performance model where the TBT assumes a “perfect cache” – each byte of the data needs to be fetched from DRAM (Dynamic Random Access Memory) only once. This assumption enables us to predict a slightly more realistic upper bound of the peak performance than by simply comparing to the TPP. Table 2 lists the key PETSc functions used for the solvers and their respective estimates of TBT based on the perfect cache assumption. The formula for SpMV follows the procedure outlined in [11]. We assume that the TBT formula for operations also involving a sparse matrix and vector like the incomplete lower-upper (ILU) factorization to be the same as MatMult(). Estimating the TBT for other important operations like the sparse matrix-matrix and triple matrix products (which are important for multi-grid methods) is an area of future work. In short, our AI formulation relies on the following four key assumptions:

(i) All floating-point operations (add, multiply, square roots, etc.) are treated equally and equate to one FLOP count.
(ii) There are no conflict misses. That is, each matrix and vector element is loaded into cache only once.
(iii) Processor never waits on a memory reference. That is, any number of loads and stores are satisfied in a single cycle.
(iv) Compilers are capable of storing scalar multipliers in the register only for pure streaming computations.
Table 2  Commonly used PETSc operations and their respective total bytes transferred

| PETSc function   | Operation                              | Total bytes transferred |
|------------------|----------------------------------------|-------------------------|
| VecNorm()        | \( a = \sqrt{\sum_{i=1}^{N} X(i)^2} \) | \( 8(N + 1) \)         |
| VecDot()         | \( a = \sum_{i=1}^{N} X(i) \ast Y(i) \) | \( 8(2N + 1) \)       |
| VecCopy()        | \( Y \leftarrow X \)                  | \( 8(2N) \)            |
| VecSet()         | \( Y(i) = a \)                         | \( 8(2N) \)            |
| VecScale()       | \( Y = a \ast Y \)                    | \( 8(2N) \)            |
| VecAXPY()        | \( Y = a \ast X + Y \)                | \( 8(3N) \)            |
| VecAYPX()        | \( Y = X + a \ast Y \)                | \( 8(3N) \)            |
| VecPointwiseMult() | \( Z(i) = X(i) \ast Y(i) \)     | \( 8(3N) \)            |
| MatMult()        | SpMV                                   | \( 4(N + nz) + 8(2N + nz) \) |

Here we note \( X, Y, Z \) as vectors with \( i = 1, \cdot \cdot \cdot, N \) entries, \( a \) is a scalar value, and \( nz \) denotes the total number of non-zeros.

We assume that the sizes of integers and doubles are 4 and 8 bytes respectively.

Fig. 2  Estimated memory bandwidth of a single Mustang and Wolf compute node based on the STREAMS Triad Benchmark

Therefore, the efficiency based on this new roofline-like performance model as:

\[
\text{Efficiency(\%)} = \frac{\text{Measured FLOPS/s}}{\min \left\{ \frac{\text{TPP}}{\text{AI} \times \text{STREAMS}} \right\}} \times 100 \quad (4.2)
\]

where the numerator is reported by the PETSc program and the denominator is the ideal performance upper-bounded by both the TPP and the product of AI and STREAMS bandwidth.

STREAMS Triad [31] is one of the most popular benchmarks for determining the achievable memory performance of a given machine. Figure 2 denotes the estimated memory bandwidth as a function of number of cores on a single Mustang and Wolf node. It is interesting to note that although the Wolf node has a greater bandwidth, there is no performance gain past eight cores. This means that an optimal use of a Wolf compute node for memory-bandwidth bound algorithms would be eight cores, whereas one would still see some performance gains when using all 24 cores on a Mustang node.

The performance model based on Eq. (4.2) is a serial model so the STREAMS metric for the Mustang and Wolf systems are 5.65 and 15.5 GB/s respectively. It should be noted that
this performance model does account for cache effects. That is, it does not quantify the useful bandwidth sustained for some level of cache. The true hardware and algorithmic efficiency is not reflected by this model, so our aim is to show relative performance between select PETSc and TAO solvers. Comparing the AI and the measured FLOPS/s with the STREAMS bandwidth will give us a better understanding of how high-performing the PETSc and TAO solvers are for select problems.

5 Representative Numerical Results

In this section, we compare the performance of our non-negative methodology using the TAO solver to that of the Galerkin formulation using the Krylov Subspace (KSP) solver. We examine the performance using two problems: 

(i) a unit cube with a hole under steady-state, and
(ii) a transient Chromium transport problem.

The diffusivity tensor is assumed to depend on the flow velocity through

\[
D(x) = \left( \alpha_T \|v\| + D_M \right) I + (\alpha_L - \alpha_T) \frac{v \otimes v}{\|v\|}
\]  

where \( \alpha_L, \alpha_T, \) and \( D_M \) denote the longitudinal dispersivity, transverse dispersivity and molecular diffusivity, respectively. We employ the conjugate gradient method and the block Jacobi/ILU(0) preconditioner for solving the linear system from the Galerkin formulation and employ TAO’s TRON and BLMVM methods for the non-negative methodology. The relative convergence tolerances for both KSP and TAO solvers are set to \( 10^{-6} \), and \( \Delta t \) for the transient response in the Chromium problem is initially set to 0.2 days. For strong-scaling studies shown here, we used OpenMPI v1.6.5 for message passing and bound processes to cores while mapping by sockets. ParaView[2] and VisIt[6] were used to generate all contour and mesh plots.

Remark Throughout the paper, the non-negative methodology that we refer to, is in fact a discrete maximum principle preserving methodology, in that, along with the non-negative constraint we also enforce that the concentrations are less than or equal to 1.

5.1 Anisotropic Diffusion in a Unit Cube with a Cubic Hole

Let the computational domain be a unit cube with a cubic hole of size \([4/9, 5/9] \times [4/9, 5/9] \times [4/9, 5/9]\). The concentration on the outer boundary is taken to be zero and the concentration on the interior boundary is taken to be unity. The volumetric source is taken as zero (i.e., \( f(x) = 0 \)). The velocity vector field for this problem is chosen to be

\[
v(x) = e_x + e_y + e_z
\]  

The diffusion parameters are set as: \( \alpha_L = 1, \alpha_T = 0.001, \) and \( D_M = 0 \). The pictorial description of the computational domain and the three mesh types composed of 4-node tetrahedrons are shown in Fig. 3. We consider three unstructured mesh types with three levels of element-wise mesh refinement, giving us nine total case studies of increasing problem size as shown in Table 3. Five different solver simulations were test for this study:

- Galerkin with CG/block Jacobi
- TRON1: with KSP tolerance of \( 10^{-1} \)
Fig. 3 Cube with a hole: pictorial description and the associated unstructured grids. a Location of the hole. b Mesh type A. c Mesh type B. d Mesh type C

- TRON2: with KSP tolerance of $10^{-2}$
- TRON3: with KSP tolerance of $10^{-3}$
- BLMVM

The TRON solvers also use the CG and block Jacobi preconditioner but with different KSP tolerances. Numerical results for both the Galerkin formulation and the non-negative methodologies for some of the mesh cases are shown in Fig. 4. The top row of figures arise from the Galerkin formulation where the white regions denote negative concentrations, and the bottom row arise from either TRON or BLMVM. Details concerning the violation of the DMP for each case study can be found in Table 4. Concentrations both negative and greater than one arise for all case studies. Moreover, simply refining the mesh does not resolve these issues; in fact, refinement worsens the violation. These numerical results indicate that our
Table 3  Cube with a hole: list of various mesh type and refinement level combinations used

| Case | Mesh type | Refinement level | Tetrahedrons | Vertices |
|------|-----------|------------------|--------------|----------|
| A1   | A         | 1                | 199,296      | 36,378   |
| B1   | B         | 1                | 409,848      | 75,427   |
| C1   | C         | 1                | 793,824      | 140,190  |
| A2   | A         | 2                | 1,594,368    | 278,194  |
| B2   | B         | 2                | 3,278,784    | 574,524  |
| C2   | C         | 2                | 6,350,592    | 1,089,562|
| A3   | A         | 3                | 12,754,994   | 2,175,330|
| B3   | B         | 3                | 26,230,272   | 4,483,126|
| C3   | C         | 3                | 50,804,736   | 9,172,044|

Fig. 4  Cube with a hole: numerical solution for cases A1 (left), B2 (middle), and C3 (right) using the Galerkin formulation (top row) and non-negative methodology (bottom row)

computational framework can successfully enforce the DMP for diffusion problems with highly anisotropic diffusivity.

5.1.1 Performance Modeling

We first consider the wall-clock time spent in the solvers on a single core. Tables 5 and 6 depict the solver time for each mesh, and we first note that Mustang system requires significantly more wall-clock time to obtain a solution than Wolf; this behavior is expected due to the difference in HPC hardware specifications listed in Table 1, specifically, Mustang has the
Table 4  Cube with a hole: minimum and maximum concentrations for each case

| Case | Min. concentration | Max. concentration | % nodes violated |
|------|--------------------|--------------------|-----------------|
| A1   | −0.0224825         | 1.00000            | 9,518/36,378 → 26.2 % |
| B1   | −0.0139559         | 1.00000            | 32,247/43,180 → 42.8 % |
| C1   | −0.0125979         | 1.00000            | 57,272/140,190 → 40.9 % |
| A2   | −0.0311518         | 1.00103            | 82,983/278,194 → 29.2 % |
| B2   | −0.0143857         | 1.00000            | 255,640/574,524 → 44.9 % |
| C2   | −0.0119539         | 1.00972            | 453,766/1,089,562 → 41.6 % |
| A3   | −0.0258559         | 1.00646            | 643,083/2,175,330 → 29.6 % |
| B3   | −0.0115908         | 1.00192            | 2,073,934/4,483,126 → 46.3 % |
| C3   | −0.0096186         | 1.00545            | 4,932,551/9,172,044 → 53.8 % |

Table 5  Cube with a hole: wall-clock times (seconds) on Mustang for each solver

| Case | Galerkin | TRON1 | TRON2 | TRON3 | BLMVM |
|------|----------|-------|-------|-------|-------|
| A1   | 0.337    | 0.933 | 0.981 | 1.14  | 2.62  |
| B1   | 0.790    | 1.72  | 2.06  | 2.71  | 5.04  |
| C1   | 2.24     | 4.34  | 5.80  | 7.74  | 13.5  |
| A2   | 7.21     | 15.2  | 21.7  | 32.5  | 72.0  |
| B2   | 15.4     | 30.0  | 43.7  | 57.5  | 109   |
| C2   | 40.4     | 67.8  | 113   | 118   | 286   |
| A3   | 121      | 225   | 414   | 599   | 1167  |
| B3   | 315      | 498   | 1061  | 1344  | 2524  |
| C3   | 997      | 1539  | 2490  | 4365  | 9679  |

Table 6  Cube with a hole: wall-clock times (seconds) on Wolf for each solver

| Case | Galerkin | TRON1 | TRON2 | TRON3 | BLMVM |
|------|----------|-------|-------|-------|-------|
| A1   | 0.126    | 0.388 | 0.396 | 0.449 | 1.01  |
| B1   | 0.314    | 0.720 | 0.853 | 1.07  | 2.03  |
| C1   | 0.888    | 1.91  | 2.47  | 3.31  | 5.71  |
| A2   | 2.58     | 6.34  | 8.74  | 12.8  | 26.2  |
| B2   | 5.90     | 12.9  | 17.8  | 22.8  | 46    |
| C2   | 16.2     | 30.1  | 47.3  | 48.9  | 133   |
| A3   | 48.0     | 98.4  | 129   | 247   | 609   |
| B3   | 107      | 171   | 342   | 435   | 1060  |
| C3   | 281      | 467   | 870   | 1245  | 3131  |

lower clock rate and lower bandwidth (as determined through STREAMS Triad). It can also be seen that the various non-negative solvers consume varying amounts of wall-clock time. BLMVM can require as much as ten times the amount of wall-clock time as the standard Galerkin method. TRON on the other hand, does not consume nearly as much time but tightening the KSP tolerances will gradually increase the amount of time. We are interested
in determining why these optimization solvers consume more wall-clock time, whether it be mostly due to additional workload associated with optimization-based techniques or due to the presence of relatively more complicated and expensive data structures compared to the standard solvers used for the Galerkin formulation. The first step is noting the total KSP and TAO iterations needed and how they vary with respect to problem size. Figure 5 depicts the KSP and TAO iterations for the Galerkin and BLMVM methods respectively. It is well-known that block Jacobi (also known as ILU(0)) requires more iterates as the size of the problem increases. In other words, the solver may exhibit poor scaling for extremely large problems, but we see that the BLMVM algorithm has an even poorer scaling rate of the solver iterates. For the TRON solvers, we document both the KSP and TAO iterates as shown in Fig. 6. We see that tightening the KSP tolerance increases the number of KSP iterates but requires slightly fewer TAO iterates. This behavior indicates that the more accurate the computed gradient projection is, the fewer optimization loops the solver has to perform.

We also examine the measured floating-point rate provided by the PETSc performance logs, as shown in Fig. 7, of all five solvers across their respective machines, and the floating point performance decreases as the problem size grows. One could compare these numbers to the TPP and see that the hardware efficiencies are no greater than 5%, but it is difficult to draw any other conclusions with regard to the computational performance. The calculated AI, based on our proposed performance model, is shown in Fig. 8. It is interesting to note that the AI remains largely invariant with problem size unlike the wall-clock time, solver iterations, and floating point rates. According to the perfect cache model, the Galerkin formulation’s
AI is greater than any of the non-negative methodologies. The optimization-based algorithm based on TAO’s BLMVM solver has significantly more streaming/vector operations which explains the relatively lower AI. Using these metrics in Eq. (4.2) as well as the STREAMS Triad bandwidth of one core as shown in Fig. 2, the estimated roofline-based efficiencies are shown in Fig. 9. Although the raw floating-point rate of BLMVM is lower than the Galerkin method, the roofline model suggests that BLMVM is actually more efficient in the hardware sense. The TRON methods have much lower floating-point rates, but these metrics can be improved or “gamed” by tightening the KSP tolerances. This behavior leads us to believe that there is some latency associated with setting up the data structures needed to compute gradient descent projections.

5.1.2 Strong-Scaling

The metric of most interest to many computational scientists is the strong-scaling potential of any numerical framework. We conduct strong-scaling studies to measure the speedup of
Fig. 9 Cube with a hole: estimated floating-point efficiency with respect to the arithmetic intensity and measured memory bandwidth from STREAMS. a Mustang. b Wolf

all nine case studies over 64 cores. Four Mustang nodes with 16 cores each and 8 Wolf nodes with 8 cores each are allocated for this study. We do not fully saturate the compute nodes because the STREAMS benchmark indicates that there is little or no gain in memory performance when using a full node. Figure 10 depicts the speedup on the Mustang system, and Fig. 11 depicts the speedup on the Wolf system. First, we note that the parallel efficiency (actual speedup over ideal speedup) increases with problem size due to Amdahl’s Law. We also note that Wolf exhibits better strong-scaling due to the faster speedups for the same test studies. For all problems and machines, the TRON simulations are slightly less efficient in the parallel sense but can be improved by tightening the KSP tolerances. Interestingly, the BLMVM algorithm not only has the best roofline efficiency but also the best parallel speedup. We can infer from these results that although BLMVM is the more efficient optimization in the hardware sense, TRON is more efficient in the algorithmic sense due to its lesser time-to-solution. Our study has shown that one can draw correlations between the performance models conducted on a single-core and the actual speedup across multiple distributed memory nodes. As future solvers and algorithms are implemented within PETSc, we can use this performance model to assess how efficient they are in both the hardware and algorithmic sense and how efficiently they will scale in a parallel setting.

5.2 Transport of Chromium in Subsurface

Subsurface clean-up due to anthropogenic contamination is a big challenge [43]. Remediation studies [12,13] need accurate predictions of transport of the involved chemical species, which are obtained using limited data at monitoring wells and through numerical simulations. To accurately predict the fate of the contaminant, a transport solver that: (a) is robust, in that it will not give unphysical solutions, and (b) can handle field-scale scenarios, is needed.
The computational framework that is proposed in this paper is an ideal candidate for such problems. We now consider a realistic large-scale problem to predict the fate of chromium in the Los Alamos, New Mexico area. The chromium was released into the Sandia canyon in the 50s up to early 70s. Back then chromium was used as an anti-corrosion agent for the cooling towers at a power plant at the Los Alamos National Laboratory (see [14] and references therein for details).

Here we study the effectiveness of our proposed framework to this real-world scenario of predicting the extent of chromium plume. The following is a conceptual model domain that is considered: A domain of size $[496 \text{ km}, 503 \text{ km}] \times [536 \text{ km}, 542 \text{ km}] \times [0, 100 \text{ m}]$ with the permeability field (m$^2$) as shown in Fig. 12. R42 in Fig. 12 is estimated to be the contaminant source location and a pumping well is located at R28. The parameters used for this problem are shown in Table 7, and we employ the following boundary conditions:

$$c^p(x = 496 \text{ km}, y, z) = c^p(x = 503 \text{ km}, y, z) = c^p(x, y = 536 \text{ km}, z) = 0$$

(5.3)

For this highly heterogeneous problem, we employ PETSc’s algebraic multi-grid preconditioner (GAMG) and couple this with the TRON algorithm for the non-negative solver. Our goal is to examine its strong-scaling potential across 1024 cores. We first solve the steady flow
equation (based on mass balance and Darcy’s model to relate pressure and mass flux) with the pumping well located at R28. Cell-wise velocity is obtained from the resulting pressure field and used to calculated element-wise dispersion tensor. We then solve the transient diffusion problem (with tensorial dispersion) with a constant contaminant source located at R40 for
Table 7 Chromium plume migration in the subsurface: parameters

| Parameter                  | Value                      |
|----------------------------|----------------------------|
| $\alpha_L$                 | 100 m                      |
| $\alpha_T$                 | 0.1 m                      |
| Contaminant source (R42)   | $1 \times 10^{-4}$ kg/m$^2$s$^2$ |
| $\Delta t$                 | 0.2 days                   |
| Domain size                | $7000 \text{ km} \times 6000 \text{ km} \times 100 \text{ m}$ |
| $D_M$                      | $1 \times 10^{-9}$ m$^2$/s |
| Permeability               | Varies                     |
| Pumping well (R28)         | $-0.01$ kg/m$^2$s$^2$      |
| Total hexahedrons          | 1,984,512                  |
| Total vertices             | 2,487,765                  |
| $v$                        | Varies with position       |
| Viscosity                  | $3.95 \times 10^{-5}$ Pa s |

up to 180 days. The concentrations at select time levels for Galerkin formulation and non-negative methodology are shown in Figs. 13 and 14, respectively. Negative concentrations arise with the Galerkin formulation even as the solution approaches steady-state.

Figure 15 depicts the amount of wall-clock time with respect to the number of cores at the first time level. It is interesting to note that there is minimal performance improvement when using up to 1024 cores because the problem size of 2.5 million degrees-of-freedom is comparatively small. That is, each core only manages approximately 2400 degrees-of-freedom so communication time will likely outweigh the computation time. However, we note that the parallel efficiency at the 64 core mark is roughly 88 percent, whereas the relative parallel efficiency from 16 to 64 cores for the unit cube A3 TRON3 case estimates an 80 percent parallel efficiency. These metrics suggest that TRON with GAMG demonstrates strong-scaling potential comparable to that of the TRON with ILU combination. A more thorough analysis of the potential bottlenecks presented by TRON and GAMG method can be done when a performance model for sparse matrix-matrix is developed.

It is also interesting to note that the wall-clock time is affected by whether or not a compute node is fully saturated. When a Wolf node utilizes all 16 cores, the performance with respect to wall-clock time is worse than when a Wolf node utilizes only 8 of its cores. The reason for this change can be explained by the lack of memory performance improvement when all 16 cores are used as shown from Fig. 2. Therefore, it is recommended that one maximize the performance of these optimization-based solvers by using partially saturated compute nodes. The Mustang system seemingly has better parallel performance metrics despite its older and less powerful computing resources, but this does not necessarily mean that TRON with GAMG preconditioning is better suited for such HPC platforms. It is important to note that newer platforms like the Wolf system conduct work more efficiently, so communication time will be more likely to also outweigh computation time.

Another metric of interest is the number of solver iterations required for convergence across various numbers of MPI processes. Figure 16 depicts the number of KSP solver iterations and TAO solver iterations across 1024 cores, and we notice that there are significant fluctuations. This trend is largely attributed to the accumulation of numerical round-offs from the TRON algorithm. One can reduce these fluctuations by tightening the solver tolerances, but the strong-scaling remains largely unaffected even for the results shown. This study suggests
Fig. 13 Chromium plume migration in the subsurface: concentrations at select times using the Galerkin formulation
Fig. 14  Chromium plume migration in the subsurface: concentrations at select times using the non-negative methodology
that the proposed non-negative methodology using TRON with GAMG preconditioning is suitable for large-scale transient heterogeneous and anisotropic diffusion equations.

6 Concluding Remarks

We presented a parallel non-negative computational framework suitable for solving large-scale steady-state and transient anisotropic diffusion equations. The main contribution is that the proposed parallel computational framework satisfies the discrete maximum principles for large-scale diffusion-type equations even on general computational grids. The parallel
framework is built upon PETSc’s DMplex data structure, which can handle unstructured meshes, and TAO for solving the resulting optimization problems from the discretization formulation. We have conducted systematic performance modeling and strong-scaling studies to demonstrate the efficiency, both in the parallel and hardware sense of the computational framework. The robustness of the proposed framework has been illustrated by solving a large-scale realistic problem involving the transport of chromium in the subsurface at Los Alamos, New Mexico. Future areas of research include: (a) extending the proposed parallel framework to advective-diffusive and advective-diffusive-reactive systems, and (b) posing the discrete problem as a variational inequality, which will be valid even for non-self-adjoint operators, and use other PETSc capabilities to solve such variational inequalities.

Acknowledgements The authors thank Matthew G. Knepley (Rice University) for his invaluable advice. The authors also thank the Los Alamos National Laboratory (LANL) Institutional Computing program. JC and KBN acknowledge the financial support from the Houston Endowment Fund and from the Department of Energy through Subsurface Biogeochemical Research Program. SK thanks the LANL LDRD program and the LANL Environmental Programs Directorate for their support. The opinions expressed in this paper are those of the authors and do not necessarily reflect that of the sponsors.

References

1. Adams, R.J., Fournier, J.J.F.: Sobolev Spaces, 2nd edn. Academic press, Amsterdam (2003)
2. Ayachit, U.: The ParaView Guide: A Parallel Visualization Application. Kitware, Clifton Park (2015)
3. Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Eijkhout, V., Gropp, W.D., Kaushik, D., Knepley, M.G., McInnes, L.C., Rupp, K., Smith, B.F., Zampini, S., Zhang, H.: PETSc users manual. Technical Report ANL-95/11 - Revision 3.6, Argonne National Laboratory, 2015
4. Benson, S.J., Munson, T.S.: Flexible complementary solvers for large-scale applications. Optim. Methods Softw. 21, 155–168 (2006)
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge, UK (2004)
6. Childs, H., Brugger, E., Whitlock, B., Meredith, J., Ahern, S., Pugmire, D., Biagas, K., Miller, M., Harrison, C., Weber, G.H., Krishnan, H., Fogal, T., Sanderson, A., Garth, C., Bethel, E.W., Camp, D., Rübel, O., Durant, M., Favre, J.M., Navrátil, P.: VisIt: An end-user tool for visualizing and analyzing very large data. In: High Performance Visualization-Enabling Extreme-Scale Scientific Insight, (pp. 357–372) (2012). https://wci.llnl.gov/simulation/computer-codes/visit
7. Ciariet, P.G., Raviart, P.-A.: Maximum principle and uniform convergence for the finite element method. Comput. Methods Appl. Methods Eng. 2, 17–31 (1973)
8. Evans, L.C.: Partial Differential Equations. American Mathematical Society, Providence (1998)
9. Facchinei, F., Pang, J.-S.: Finite-Dimensional Variational Inequalities and Complementarity Problems. Springer, New York (2003)
10. Graser, C., Kornhuber, R.: Multigrid methods for obstacle problems. J. Comput. Math. 27, 1–44 (2009)
11. Gropp, W.D., Kaushik, D.K., Keyes, D.E., Smith, B.F.: Toward realistic performance bounds for implicit CFD codes. In Proceedings of Parallel CFD ’99, pp. 233–240. Elsevier, (1999)
12. Hammond, G.E., Lichtner, P.C.: Field-scale model for the natural attenuation of uranium at the Hanford 300 Area using high-performance computing. Water Resour. Res. 46, W09602 (2010)
13. Harp, D.R., Vesselinov, V.V.: Contaminant remediation decision analysis using information gap theory. Stoch. Environ. Risk Assess. 27(1), 159–168 (2013)
14. Heikoop, J.M., Johnson, T.M., Birdsell, K.H., Longmire, P., Hickmott, D.D., Jacobs, E.P., Broxton, D.E., Katzman, D., Vesselinov, V.V., Ding, M., Vaniman, D.T., Reneaua, S.L., Goering, T.J., Glessnrb, J., Basu, A.: Isotopic evidence for reduction of anthropogenic hexavalent chromium in Los Alamos National Laboratory groundwater. Chem. Geol. 373, 1–9 (2014)
15. Hjelmstad, K.D.: Fundamentals of Structural Mechanics, 2nd edn. Springer, New York (2005)
16. Karra, S., Painter, S.L., Lichtner, P.C.: Three-phase numerical model for subsurface hydrology in permafrost-affected regions (PFLOTRAN-ICE v1.0), Cryosphere 8(5), 1935–1950 (2014)
17. Karypis, G., Kumar, V.: A fast and highly quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. 20, 359–392 (1999)
18. Kelkar, S., Lewis, K., Karra, S., Zyvoloski, G., Rapaka, S., Viswanathan, H., Mishra, P.K., Chu, S., Coblenz, D., Pawar, R.: A simulator for modeling coupled thermo-hydro-mechanical processes in subsurface geological media. Int. J. Rock Mech. Min. Sci. 70, 569–580 (2014)

19. Kinderlehrer, D., Stampacchia, G.: An Introduction to Variational Inequalities and Their Applications. SIAM Classics in Applied Mathematics, New York (2000)

20. Kirby, R.C.: FIAT: Numerical Construction of Finite Element Basis Functions, Chapter 13. Springer, New York (2012)

21. Knepley, M.G., Brown, J., Rupp, K., Smith, B.F.: Achieving High Performance with Unified Residual Evaluation. arXiv:1309.1204, September (2013)

22. Knepley, M.G., Karpeev, D.A.: Mesh algorithms for PDE with vieve I: Mesh distribution. Sci. Program. 17, 215–230 (2009)

23. Lange, M., Knepley, M.G., Gorman., G.J.: Flexible, scalable mesh and data management using PETSc DM. In Proceedings of the 3rd International Conference on Exascale Applications and Software, EASC ‘15, pp. 71–76. University of Edinburgh, (2015)

24. Lichtner, P.C., Hammond, G.E., Lu, C., Karra, S., Bisht, G., Andre, B., Mills, R.T., Kumar, J.: PFLOTRAN user manual: A massively parallel reactive flow and transport model for describing surface and subsurface processes. Technical Report Report No.: LA-UR-15-20403, Los Alamos National Laboratory, (2015)

25. Lichtner, P.C., Karra, S.: Modeling multiscale-multiphase-multicomponent reactive flows in porous media: Application to CO2 sequestration and enhanced geothermal energy using pilotran. In R. Al-Khoury and J. Bundschuh, editors, Computational Models for CO2 Geo-sequestration & Compressed Air Energy Storage, pp. 81–136. CRC Press, http://www.crcnetbase.com/doi/pdfplus/10.1201/b16790-6, (2014)

26. Liska, R., Shashkov, M.: Enforcing the discrete maximum principle for linear finite element solutions of second-order elliptic problems. Commun. Comput. Phys. 3(4), 852–877 (2008)

27. Lo, Y.J., Williams, S., Straalen, B.V., Ligocki, T.J., Cordery, M.J., Wright, N.J., Hall, M.W., Oliker, L.: Roofline: an insightful visual performance model for multicore architectures. High Perform. Comput. Syst. Perform. Model. Benchmarking Simul. 8966, 129–148 (2015)

28. Logg, A.: Efficient representation of computational Meshes. Int. J. Comput. Sci. Eng. 4, 283–295 (2009)

29. Logg, A., Mardal, K.A., Wells, G.N.: Automated Solution of Differential Equations by the Finite Element Method. Springer, New York (2012)

30. May, D.A., Brown, J., Laetitia, L.L.: pTatin3D: High-performance methods for long-term lithospheric dynamics. In Proceedings of the International Conference for High Performance Computing, Network, Storage and Analysis, SC ’14, pp. 274–284. IEEE Press, (2014)

31. McCulpine, J.: STREAM: sustainable memory bandwidth in high performance computers, (1995). https://www.cs.virginia.edu/stream/

32. Mudunuru, M.K., Nakshatrala, K.B.: On enforcing maximum principles and achieving element-wise species balance for advection-diffusion-reaction equations under the finite element method. J. Comput. Phys. 305, 448–493 (2016)

33. Mudunuru, M.K., Nakshatrala, K.B.: On mesh restrictions to satisfy comparison principles, maximum principles, and the non-negative constraint: Recent developments and new results. Mechanics of Advanced Materials and Structures (2016). doi: 10.1080/15502287.2016.1166160

34. Munson, T., Sarich, J., Wild, S., Benson, S., McInnes, L.C.: TAO 2.0 users manual. Technical Report ANL/MCS-TM-322, Mathematics and Computer Science Division, Argonne National Laboratory. http://www.mcs.anl.gov/tao (2012)

35. Nagarajan, H., Nakshatrala, K.B.: Enforcing the non-negativity constraint and maximum principles for diffusion with decay on general computational grids. Int. J. Numer. Methods Fluids 67, 820–847 (2011)

36. Nakshatrala, K.B., Mudunuru, M.K., Valocchi, A.J.: A numerical framework for diffusion-controlled bimolecular-reactive systems to enforce maximum principles and the non-negative constraint. J. Comput. Phys. 253, 278–307 (2013)

37. Nakshatrala, K.B., Nagarajan, H., Shabouei, M.: A numerical methodology for enforcing maximum principles and the non-negative constraint for transient diffusion equations. Commun. Comput. Phys. 19, 53–93 (2016)

38. Nakshatrala, K.B., Valocchi, A.J.: Non-negative mixed finite element formulations for a tensorial diffusion equation. J. Comput. Phys. 228, 6726–6752 (2009)

39. Payette, G.S., Nakshatrala, K.B., Reddy, J.N.: On the performance of high-order finite elements with respect to maximum principles and the nonnegative constraint for diffusion-type equations. Int. J. Numer. Methods Eng. 91, 742–771 (2012)

40. Pruess, K.: The TOUGH codes—a family of simulation tools for multiphase flow and transport processes in permeable media. Vadose Zone J. 3(3), 738–746 (2004)

41. The HDF Group. Hierarchical Data Format, version 5. http://www.hdfgroup.org/HDF5/ (1997–2015)
42. Ulbrich, M.: Semismooth Newton Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces. SIAM, Philadelphia (2011)
43. US EPA. Cleaning up the nation’s waste sites: Markets and technology trends. Technical Report EPA 542-R-04-015, (2004)
44. Williams, S., Waterman, A., Patterson, D.: Roofline: an insightful visual performance model for multicore architectures. Commun. ACM 52, 65–76 (2009)
45. Zyvoloski, G.: FEHM: A control volume finite element code for simulating subsurface multi-phase multi-fluid heat and mass transfer. Los Alamos Unclassified Report LA-UR-07-3359, (2007)