Greedy permanent magnet optimization

Alan A. Kaptanoglu\(^1\),* \(\odot\), Rory Conlin\(^2\) \(\odot\) and Matt Landreman\(^1\) \(\odot\)

\(^1\) Institute for Research in Electronics and Applied Physics, University of Maryland, College Park, MD 20742, United States of America
\(^2\) Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ 08544, United States of America

E-mail: akaptano@umd.edu

Received 22 August 2022, revised 11 December 2022
Accepted for publication 19 January 2023
Published 3 February 2023

Abstract

A number of scientific fields rely on placing permanent magnets in order to produce a desired magnetic field. We have shown in recent work that the placement process can be formulated as sparse regression. However, binary, grid-aligned solutions are desired for realistic engineering designs. We now show that the binary permanent magnet problem can be formulated as a quadratic program with quadratic equality constraints, the binary, grid-aligned problem is equivalent to the quadratic knapsack problem with multiple knapsack constraints, and the single-orientation-only problem is equivalent to the unconstrained quadratic binary problem. We then provide a set of simple greedy algorithms for solving variants of permanent magnet optimization, and demonstrate their capabilities by designing magnets for stellarator plasmas. The algorithms can a-priori produce sparse, grid-aligned, binary solutions. Despite its simple design and greedy nature, we provide an algorithm that compares with or even outperforms the state-of-the-art algorithms while being substantially faster, more flexible, and easier to use.

Keywords: permanent magnets, stellarators, greedy algorithms, sparse regression, combinatorial optimization, binary quadratic programs, quadratic knapsack problems

(Some figures may appear in colour only in the online journal)

1. Introduction

A common scientific goal is to produce a target magnetic field in a prescribed volume, and this can be accomplished with coils, permanent magnets, or a combination of coils and magnets. Experiments in the field of plasma physics often require strong, three-dimensional magnetic fields, requiring very complex magnetic coils. One class of plasma experiments, stellarators, particularly relies on sophisticated coil design algorithms in order to produce carefully shaped magnetic fields that can provide high-quality confinement of charged particle trajectories and many other physics objectives. Stellarator optimization is typically divided into two stages. The first is a configuration optimization using fixed-boundary magnetohydrodynamic (MHD) equilibrium codes to obtain MHD equilibria with desirable physics properties [1–3]. After obtaining the optimal magnetic field in this first stage, complex coils must be designed to produce these fields [4] and this complexity raises the cost and difficulty of manufacturing. The primary cost of the W7-X and NCSX stellarator programs was the manufacture and assembly of these coils with very tight engineering tolerances [5, 6].

A recent proposal to circumvent this requirement is to simplify stellarator coil designs by surrounding a stellarator with a manifold of permanent magnets that can provide significant portions of the magnetic field [7]. These permanent magnets cannot be used to generate a net toroidal flux, so a set of simple magnetic coils are still required. Instead, the permanent magnets allow for significant reductions in the coil complexity and
cost, operate without power supplies, require minimal cooling, ameliorate magnetic ripple due to discrete coils, and facilitate improved diagnostic access. Some potential disadvantages include the inability to turn off the field, the possibility of demagnetization, and an upper limit on the achievable field strength.

These disadvantages may prevent permanent magnet designs for fusion-relevant devices, which typically require very large magnetic fields. However, there are a number of plausible scenarios for permanent magnets to contribute to fusion, including: recent advances in materials science [8] leading to higher field limits, placing magnets only on the low-field side, substituting permanent magnets for a very large number of small superconducting coils (i.e. dipoles without a limit on field strength that can also be optimized with the formalism in the present work), and substituting in other related concepts such as superconducting tiles [9]. Even if the barriers to using permanent magnets at fusion-relevant scales prove insurmountable, the low cost and simple manufacture of permanent magnet stellarators are attractive properties for building new university-scale experiments such as MUSE [10]. Such small to medium-scale experiments are often essential for improving nuclear-fusion devices anyways, since they provide fundamental research and training to new students and researchers who often go on to contribute at larger fusion facilities. Lastly, note that similar permanent magnet designs can be used outside the field of stellarator design, including for magnetic resonance imaging [11, 12], automobile manufacturing [13], and many other fields [14].

Using permanent magnets for magnetic field shaping comes with its own challenges. Sophisticated algorithms are still required to find high-quality configurations of permanent magnets. There are several different formulations and associated algorithms for addressing the permanent magnet optimization problem [10, 15–19], but the relationships between them are often unclear. Some of the optimization problems are multi-stage or use discrete optimization, and the best set of loss terms is an open question. Often additional post-processing optimization steps are taken to further improve the initial optimization solutions.

Greedy algorithms are a class of algorithms that solve an optimization problem by starting with the empty set and then iteratively adding the element that most minimizes the objective at each iteration. Greedy algorithms for stellarator permanent magnet optimization have been considered previously, e.g. Lu et al [20], but do not take advantage of the connection with sparse regression (and, as we show below, with the literature on quadratic binary problems) to design algorithms that have been shown in other scientific fields to exhibit various performance guarantees. This connection allows us to design a much simpler greedy algorithm that retains high performance but now requires only a single optimization stage and makes no approximation that the dipoles contribute purely local fields on the plasma surface. Common to essentially all of these algorithms is that a solution, representing a set of permanent magnet locations and associated dipole vectors, is desired that is easy to manufacture (grid-aligned), all maximum-strength (binary), and low-cost (uses as few binary magnets as possible, i.e. sparse).

In recent work [21], we formulated the permanent magnet optimization problem as sparse regression, enabling effective algorithms [22–25] to be adapted from this prolific scientific field. We provided a new algorithm that circumnavigates the sensitivity to the initial condition and produces accurate permanent magnet stellarators.

1.1. Contributions of the present work

Now that the permanent magnet optimization problem has been formulated as sparse regression, we show that the fully binary problem is equivalent to a quadratic program with quadratic equality constraints (QPCQ), the binary, grid-aligned problem is equivalent to the quadratic knapsack problem with multiple knapsack constraints (MdQKP), and the binary, single-orientation problem is a binary quadratic program (BQP). These formulations provide a route for the future application of more sophisticated greedy algorithms and performance bounds to permanent magnet optimization.

We then provide a simple greedy algorithm to solve the optimization. Our implementation is computationally fast, intuitive, easy-to-use, and open-source. It a-priori produces sparse, binary solutions that are independent of the initial guess, and additionally illustrates that greedy algorithms can perform comparably to or even better than the state-of-the-art. We propose some likely reasons for the unusual effectiveness of greedy algorithms in the context of permanent magnet optimization in section 3.6. In the present work, there are essentially no hyperparameters, no local approximations, and no multi-stage operations to perform. All of the present work’s methodology and results can be found in the SIMSOPT code [3]. As in our recent work, the entirety of the permanent magnet pipeline, i.e. the geometry, optimization tools, post-processing, etc is contained in this single open-source tool.

2. Methodology

Consider $D$ permanent magnet locations, and a vector of the potential dipole components $m = [m_1, m_1^2, m_2, m_2^2, \ldots, m_D]$ as the variables for optimization. Cartesian coordinates are not a requirement and the choice of coordinate system determines which directions are considered grid-aligned. Note that it is trivial to extend this formulation to use a set of dipole components in greater than three directions, such as for the six possible dipole alignments considered in Hammond et al [26]. The natural choice for generating sparse vectors is to minimize an objective function with the $l_0$ norm $\|m\|_0 = \#$ of nonzero elements of $m$. We want to solve the following optimization problem,

$$m^* = \arg \min_m f_b(m) + f_m(m) + \alpha \|m\|_0,$$

subject to: $\|m_i\|_2^2 \leq (m_i^{\text{max}})^2$,

$$f_b \equiv \frac{1}{2} \|Am - b\|_2^2, \quad f_m \equiv \lambda \|m\|_2^2.$$  

(1)
\( A \in \mathbb{R}^{N \times 3D} \) encodes the magnet geometry and symmetries, \( b \in \mathbb{R}^{N} \) encodes the normal magnetic field on the plasma surface due to the fixed electromagnets and any plasma current, \( f_m \) is Tikhonov regularization that helps to reduce the ill-posedness of the problem, and the constraints come from the maximum magnetization of the material. \( f_B \) encodes the normal magnetic field errors on the stage-1-optimized plasma boundary defined at \( N \) points,

\[
f_B = \frac{1}{2} \int ((B_M + B_P + B_C) \cdot \hat{n})^2 \, d^2 x = \frac{S_A}{2} \langle |B_{\text{total}} \cdot \hat{n}|^2 \rangle.
\]

Here \( \hat{n} \) is the normal vector to the plasma boundary, \( \langle . \rangle \) denotes an average over the plasma boundary, \( S_A \) is the surface area of the plasma boundary, and \( B_P, B_C, \) and \( B_M \) are the magnetic fields generated by plasma current, the traditional coils, and the permanent magnets, respectively. For the purposes of this work, \( B_m \) is the field from a stage-1 optimized plasma boundary and \( B_C \) is the field generated from a minimal set of basic coils that produce the net toroidal flux. For a dimensionless metric, we will also occasionally report

\[
b_{\text{u}} \equiv \langle |B_{\text{total}} \cdot \hat{n}| \rangle / \langle |B_{\text{total}}| \rangle.
\]

Solving (1) is quite challenging because it is a nonsmooth, nonconvex problem in a very high-dimensional space. Approaches to high-dimensional sparse regression typically either convexify the problem, e.g., with the \( l_1 \) norm, or use a greedy algorithm, albeit often with weak guarantees on optimality [27]. The relax-and-split algorithm used in our previous work [21] arrives at binary, grid-aligned solutions by repeatedly solving (1) while slowly increasing the value of \( \alpha \) until only maximum-strength magnets remain.

One way to guarantee a solution with maximum-strength, but generic directionality dipoles is to reformulate the problem without the \( l_0 \) term and turn the inequality constraints into equality constraints:

\[
m^* = \arg \min_m \left[ f_B(m) + f_m(m) \right],
\]

subject to: \( \|m\|_2^2 = (m^\text{max})^2 \).

This is a formulation of binary permanent magnet optimization as a QPQC, but it is not yet sensible as written. There is a significant issue—the formulation assumes all the dipoles are maximal, so that there is no sparsity in the solution and \( f_m \) plays no role at all. Moreover, unlike with the inequality constraints in (1), the equality constraints are nonconvex. Nonetheless, we will show in section 2.3 this is a useful formulation that, together with a greedy algorithm, helps us to design a sparse, binary, but otherwise arbitrary direction set of dipoles.

Unfortunately, this is not such an interesting case for designing real-world permanent magnet stellarators. Constructing such a stellarator with a manifold of \( \sim 10000 \) arbitrarily oriented permanent magnets would be quite challenging, and is one of the reasons the MUSE grid requires dipoles that point entirely inwards or outwards in the minor radial direction.

### 2.1. Permanent magnet optimization as discrete optimization

In order to facilitate an analysis of the truly discrete version of (1), we normalize the \( m \) by their maximum strengths \( m_i \rightarrow m_i/m_i^\text{max}, A \rightarrow Am^\text{max} \), and replace the \( l_0 \) norm in (1) with a restriction to \( m_i \in \{ -1, 0, 1 \} \) on the optimization variables,

\[
\arg \min_{m \in \{ -1, 0, 1 \}} \frac{1}{2} \|Am - b\|_2^2 + \lambda \|Bm\|_2^2,
\]

\[
\|m\|_2^2 \leq 1.
\]

Here \( B = \text{diag}(m_i^\text{max}) \in \mathbb{R}^{3D \times 3D} \) and \( m_i^\text{max} \) is shorthand for the vector formed by triplets of the maximum strengths of each dipole, \([m_1^\text{max}, m_1^\text{max}, m_1^\text{max}, m_2^\text{max}, \ldots, m_D^\text{max}] \). Note that the grid-aligned property is enforced by the combination of the quadratic constraints and the discrete variables. For the remainder of the present work, all of the optimization variables \( m, A, \) etc will refer exclusively to the normalized quantities used in (5).

Despite the reduced form of (5), we have yet to connect it to the larger scientific literature detailing algorithms and performance bounds for constrained, binary minimization of the MSE. To do so, we transform the optimization in (5) to a variant of the well-studied quadratic knapsack problem (QKP) [28] with multiple constraints, MdQKP [29]. First, redefine the variables for optimization to \( m = [m_1^* m_1^* m_1^* m_1^* m_2^* m_2^* \ldots m_D^*] \), so that the problem can be restricted to binary variables. We can encode the grid-aligned property by trading in the quadratic constraints for affine constraints. After reshaping the \( A \) and \( B \) matrices, (5) becomes,

\[
\arg \min_{m \in \{ 0, 1 \}^D} \left( m^T \left[ \frac{1}{2} A^T A + \lambda B^T B \right] m - b^T Am \right),
\]

\[
\sum_{j=1}^{6D} C_{ij} m_j \leq 1, \quad \forall i \in \{ 1, \ldots, D \}.
\]

Here a choice for \( C_{ij} \) encodes a single constraint per dipole,

\[
0 \leq (m_i^* + m_i^- - 1) + (m_i^* + m_i^- - 1) + (m_i^- + m_i^*) \leq 1.
\]

Along with \( m_i \in \{ 0, 1 \} \) for all \( i \), the constraints are sufficient for grid-aligned dipoles, and (6) is now in the form of a MdQKP.

The largest MdQKP problem addressed by CPLEX’s mixed-integer quadratic program solver in 2012 from Wang et al [30] was of size \( D = 800 \) with 15 constraints and it took a few hours to compute on a personal computer. In the permanent magnet optimization scenario with \( D \sim 10^5 \) and \( \sim 10^6 \) constraints, we need a much simpler algorithm.

Before we get to a simpler algorithm, consider the scenario in which the dipoles are a-priori desired to be in a particular coordinate direction, as in the minor radial direction in the MUSE example in section 3. We can then use the optimization variables \( m = [m_1^*, m_1^*, m_2^* \ldots, m_D^*] \) and (6) becomes

\[
\arg \min_{m \in \{ 0, 1 \}^D} \left( m^T \left[ \frac{1}{2} A^T A + \lambda B^T B \right] m - b^T Am \right).
\]
The reduced constraint $0 \leq m_f^t + m_g^{-t} \leq 1$ is not required because $f_m$ prevents $m_f^t = m_g^{-t} = 1$. Assuming $B$ has been defined appropriately, i.e. so that $f_m \propto (m_f^t)^2 + (m_g^{-t})^2$, then the objective can be reduced by changing both $m_f^t$ and $m_g^{-t}$ from 1 to 0. This new solution leaves the $f_B$ term unaffected, while the $f_m$ term is reduced.

Now (8) is the unconstrained quadratic knapsack problem, which is to say it is the unconstrained BQP. Despite the significant reductions, this problem is NP-hard and exact methods of solution have been limited to problems of a few hundred variables. Even the heuristic and greedy methods reported in Kochenberger et al [31] were limited to $D \lesssim 10^5$ or so. Part of the reason for this limit is that often quite sophisticated heuristic methods are used, such as methods incorporating tabu search [32], local search schemes [33], and so on [34]. Similarly, variants with tabu search [35, 36] and genetic algorithms [37] are frequent choices for heuristically solving variants of QKP, but these algorithms are also typically applied to problems with a few hundred optimization variables.

In contrast, we start with a very simple and intuitive greedy algorithm, add some basic backtracking, and already can compete with or even outperform the state-of-the-art algorithms for binary permanent magnet optimization with $D \sim 10^5$. Similar algorithms have been used with some success in the BQP literature for decades [38, 39], but have not been applied for permanent magnet optimization. Reasons for the unusual effectiveness of these simple algorithms in binary permanent magnet optimization are explored in section 3.6. Lastly, the significant literature on heuristic and greedy algorithms for BQPs and QKPs provides a road map for improvements in future algorithms.

### 2.2. Greedy permanent magnet optimization (GPMO)

Similar in spirit to the binary matching pursuit algorithm in Wen and Li [40], we devise a greedy, binary algorithm for reducing the mean-squared error (MSE) in permanent magnet optimization. This algorithm may be used with objectives other than the MSE, but minimizing other objectives does not typically provide guarantees on the smallness of the MSE. This is problematic, because the MSE in permanent magnet optimization must be greatly reduced for a solution to be useful.

A stage-two stellarator optimization must match the normal magnetic field of a stage-1-optimized plasma boundary very closely in order to avoid producing significant deviations from the stage-1 flux surfaces and other optimized physical quantities such as the fast-particle confinement.

Initially, the greedy approach may seem easy to implement but unlikely to generate high-quality solutions. Calculating performance guarantees, e.g. the $f_B$ ratio between the true optimum and greedy solution, for greedy algorithms is not straightforward. However, a greedy solution that produces 100 times larger $f_B$ than the optimum might still be a very useful solution, since permanent magnet grids can sometimes be designed that exhibit solutions satisfying $f_B \sim 10^{-10} \text{T m}^2$. A brief review of the relevant mathematical concepts for greedy performance guarantees can be found in appendix. At least in the case of continuous optimization, recent results have shown that, when the MSE is used as the objective function, greedy algorithms retain some reasonable performance guarantees [41, 42]. These guarantees are one justification for using the MSE as an objective for greedy minimization; another justification is simply to show high performance is available using the MSE, as we do in section 3.

### 2.3. The GPMO algorithm

Our algorithm is as simple and intuitive as possible. One-by-one, place a maximum-strength permanent magnet (with only one nonzero component in order to be grid-aligned) that maximally reduces $f_B$. Then add all of the components of this magnet to the list of indices that are off limits to future magnet placement. This process continues until K magnets have been placed. Once a magnet is placed on the grid, it is immutable, so additional algorithm iterations can only add new magnets until the grid has no remaining unfilled locations. We will address this potential issue with our modified algorithm in the following section, which does allow for backtracking during the optimization process to correct for suboptimal magnet placements. The code also allows users to save a solution with $K$ magnets, and use it later as an initial condition for a new optimization that places more than $K$ magnets.

If the permanent magnets are all identical, as is the case for the MUSE grid used in section 3, minimizing $f_B + f_m$ is not required after rescaling because each dipole contributes an identical amount to the regularization $f_m$. If the grid does not consist of identical elements, the effect of $f_m$ is to preference smaller magnets that contribute most significantly to the solution. Algorithm 1 summarizes the greedy permanent magnet optimization (GPMO) approach.

In a greedy approach, we can use the original optimization variables $m = [m_f^1, \ldots , m_f^K, m_g^1, \ldots , m_g^K]$ because after each magnet placement, we can eliminate the related variable components from consideration. The computational bottleneck of iteration $k$ is evaluating $A_m$, which can be trivially parallelized with OpenMP. Moreover, the result of the previous matrix-vector

---

**Algorithm 1. GPMO.**

**Input:** All the physics encoded in $f_B$ and $K$ number of magnets to place. Initialize $\Gamma^{(1)} = \{1, \ldots , 3D\}, m^{(1)} = 0$.

**Output:** Solution $m^{(K)}$.

**procedure** GPMO($A$, $b$)

for $k = 1, \ldots , K$

\[ i^* = \arg \min_{i \in \Gamma^{(k)}} f_B(m^{(k)})|m_i = \pm 1\] \[ m^{(k+1)}_{i^*} = \pm 1, \] \[ \Gamma^{(k+1)} = \Gamma^{(k)} \setminus \{i^*, \text{remaining components of the dipole}\}, \]

end procedure

In words: find the component $i^*$ of $m$, that when set to $\pm 1$, minimizes $f_B$ the most. In order to generate grid-aligned solutions, subtract the chosen $i^*$ index and the remaining components of the same dipole from $\Gamma$. Continue until maximum-strength, grid-aligned dipoles have been placed.
product can be stored each iteration, so that in the next iteration only a simple sum is required for checking the contribution of dipole $j$. For concreteness, define $\Gamma^{(k)}$ as the set of available magnet locations during iteration $k$ and suppose $\Gamma^{(k+1)} = \Gamma^{(k)} - \{j\}$. Then the contribution of the $j$th element of $m$ to $f_B$ during iteration $(k+1)$ is,

$$\sum_{l \in \Gamma^{(k+1)}_c} A_{jl} m_l^{(k+1)} = A_{j} m_j + \sum_{l \in \Gamma^{(k)}} A_{jl} m_l^{(k)},$$

(9)

where $\Gamma_c$ denotes the complement of $\Gamma$ and the right-most term is known from the previous iteration. The worst case computational complexity is $\sim 3NDK$ floating-point operations, parallelized over $P$ OpenMP processes, and the calculation speeds up as $K \rightarrow D$. On the Cori KNL machine, placing a very large number ($K = 40000$) of magnets for the MUSE stellarator with $D = 75460$ potential magnet locations, $N = 64 \times 64$ quadrature points on the plasma boundary, and $P = 68$ OpenMP processes, takes about 20 min. Given a prescribed permanent magnet grid (and therefore $D$), the algorithm scales well from the linear dependence on $N$ and $K$ and straightforward parallelization. A computational speed comparison with other algorithms is illustrated in section 3.5.

Finally, GPMO can be modified to solve the binary, arbitrary-orientation formulation in (4). The optimal dipole to place during iteration $(k+1)$ is the dipole that produces the smallest value from solving

$$\min_{m} \left[ f_B(m) | m_i \neq 0 \right] + f_m(m) | m_i \neq 0 \right], \|m\|_2^2 = 1.$$  

(10)

Computing this optimal location requires solving $(D-k)$ problems in the form of (10) every iteration. Each problem is a size-three OP with a single quadratic constraint. In the single constraint case, each OP can be transformed into a linear program and efficiently solved [43]. We leave further exploration of this route to future work.

2.4. GPMO with multiple magnets per iteration (GPMOm)

A useful engineering constraint to implement in the GPMO algorithm is the placement of multiple magnets per iteration. Multiple index selections per iteration is now a common variant of the orthogonal matching pursuit and other greedy algorithms in the sparse regression field [44].

This is interesting from an engineering perspective because Algorithm 1 (GPMO) tends to produce solutions that exhibit a number of isolated magnets. This property is disadvantageous from a cost and engineering perspective, and often large portions of the surface of the plasma experiment need to be accessible for diagnostics. Instead of placing magnets one-by-one, Algorithm 1 is modified to place the set of $p$ magnets that most minimizes $f_B$ at each iteration. The set of $p$ magnets can be additionally required to be a continuous set of magnets so no isolated magnets are ever placed. However, there are a combinatorically large number of ways to divide the grid into sets of $p$ magnets, and within each set, an optimization over the $p$ magnets may still be required to best orient the $p$ magnets.

For simplicity, we choose a partitioning in which, at each iteration, a dipole is placed along with its closest available adjacent neighbors (some neighbor dipoles may have already been placed in a previous iteration so the algorithm sometimes chooses $N_{\text{adjacent}}$ somewhat more distant neighbors). For convenience, we define $N_{\text{adjacent}}$ by including the dipole in question, so that $N_{\text{adjacent}} = 1$ refers to using only the selected dipole. All of the neighbor dipoles are used with the same coordinate-aligned orientation as the selected dipole, and the dipole chunk chosen at each iteration is the one that minimizes $f_B$. This is a sort of coarse-grained approach of setting down large magnets all at once. We denote this algorithm GPMOm and if $N_{\text{adjacent}} = 1$ GPMOm reduces to the GPMO algorithm. $N_{\text{adjacent}}$ is the only hyperparameter.

There are a number of other ways to dramatically reduce the number of isolated dipoles, such as adding a loss term that penalizes dipole locations with no near neighbors, or requiring the dipoles must be placed next to existing dipoles after some $K_{\text{min}}$ number of dipoles have been placed with the ordinary GPMO algorithm. However, unlike GPMOm, these algorithms cannot guarantee that there will be exactly zero isolated dipoles in the final solution.

2.5. GPMO with backtracking (GPMOb)

Algorithm 1 is a greedy algorithm and therefore will sometimes make poor choices for the magnets. These poor choices are often corrected later in the optimization when many more magnets have been placed. In fact, it can be seen at around $K \sim 10000$ in figure 1 that the $r$-direction-only GPMO solution briefly outperforms the GPMO solution with all $(r, \phi, \theta)$ directions available. The latter algorithm cannot have generated an optimal solution.

The ill-conditioned nature of this problem actually facilitates a natural route for error correction; each unhelpful magnet is approximately zeroed out by placing an equal and opposite magnet directly next to it. These close cancellations are somewhat common, especially if many magnets have been placed, $K \sim D$. A natural way to improve these solutions is to perform some backtracking.

After each set of several hundred iterations, the algorithm can prune magnets that are adjacent to another magnet of opposite orientation. The pruning of a close-cancelling pair of dipoles barely changes the $f_B$ metric and is a systematic way to backtrack and correct poor magnet choices made earlier in the algorithm. The backtracking works as follows: $N_{\text{adjacent}}$ neighbors are obtained for each existing dipole, and then a search is made for the nearest oppositely oriented neighbor dipole. If one is found, the neighbor and the existing dipole are both removed from the solution, but the locations are allowed to host magnets in the future.
We denote this algorithm as GPMOb, and there are now in principle two hyperparameters: the maximum distance to consider between two oppositely oriented dipoles to be eliminated (here determined implicitly by choosing how many neighbor dipoles are considered adjacent via the $N_{\text{adjacent}}$ hyperparameter), and how often to perform the backtracking, $K_b$. However, it was empirically found that the $f_B$ performance was insensitive for values of $K_b \lesssim 500$ since this meant the backtracking was performed frequently enough to prevent large accumulations of poorly placed magnets. In the present work, the value $K_b = 200$ was used for generating all of the GPMOb results.

2.6. Other possible variants of GPMO

The greedy approach allows for useful engineering constraints to be straightforwardly incorporated into new permanent magnet configurations. Here, we briefly point out another opportunity with the greedy algorithms. One potentially advantageous property is to generate a configuration that is built from one continuous block of magnetic material. This can be generated greedily by, after placing the first magnet, always placing magnets directly next to existing magnets. With stellarator and field-period symmetries, this means the permanent magnet configuration is simply $2 \times N_{fp}$ geometrically identical permanent magnet blocks.

3. Results

We now present permanent magnet solutions for the MUSE stellarator. MUSE is a table-top stellarator experiment using permanent magnets that is currently under construction [10]. MUSE is stellarator-symmetric and two-fold field-period symmetric, so it is only required to design dipoles for a quarter of the toroidal angle extent, and then to repeat this configuration around the torus. All of the following algorithms are run with a set of high-resolution quadrature points on the unique part of the plasma boundary—64 points in the poloidal angle $\theta$, and 64 points in the toroidal angle $\phi$ ($\times 2N_{fp}$ for all $N_{fp}$ field periods).

MUSE was optimized for a high degree of quasi-symmetry, and the experiment’s permanent magnet configuration was optimized with the FAMUS algorithm. Substantial discrete optimization was used in FAMUS post-processing steps to achieve engineering constraints while preserving the physics.
illustrates that GPMO obtains comparable (but marginally
smaller) GPMO results because the grid consists of identical elements.

FAMUS, relax-and-split, and the greedy methods all use the same stage-1 optimized plasma surface, the same 16 planar toroidal field coils, and the same MUSE permanent magnet grid array of four toroidal quadrants with a simple toroidal coordinate system. For these optimizations, the number of dipoles is $D = 75460 \times 4$ via symmetries, each with three vector components, so $A \in \mathbb{R}^{4096 \times 226380}$. Tikhonov regularization is omitted from the relax-and-split algorithm for simplicity and it is not relevant for the greedy algorithms on the MUSE grid because the grid consists of identical elements.

3.1. GPMO results

The greedy algorithm performs remarkably well. Figure 1 illustrates that GPMO obtains comparable (but marginally worse) $f_B$ performance with the same number of magnets as the FAMUS and relax-and-split solutions. The GPMO solution with $K \approx 12,500$ and only $r$-aligned dipoles looks very similar to the FAMUS solution. But GPMO has no hyperparameters to tune and the solution took \sim5 min to compute. Its simplicity facilitates the exploration of more exotic configurations such as $\phi$-aligned or $\theta$-aligned magnet configurations. A reasonably accurate solution $f_B \approx 6 \times 10^{-7} \text{T}^2 \text{m}^2$ can even be generated with an all-$\phi$-aligned configuration.

Consider the remarkable fact that the GPMO, FAMUS, and relax-and-split solutions qualitatively match. The three algorithms have very different degrees of freedom, define different optimization problems, and solve the problem with disparate iterative approaches. There are two effects that seem like plausible reasons for this convergence between algorithms. First, certain grid locations may be essential for properly minimizing $f_B$, e.g., locations near highly-shaped magnetic fields on the plasma boundary. Second, the requirement on each of the algorithms that the end solution must consist of maximum strength, binary magnets effectively regularizes much of the permanent magnet optimization space. In other words, there may be a vast number of permanent magnet configurations that minimize $f_B$ to high-performance levels, but far fewer such configurations have binary distributions. The specific reasons for the high performance of GPMO is further remarked upon in section 3.6.

GPMO also shows that after $\sim 15,000$ magnets have been placed on this grid, placing additional magnets has fairly negligible impact on $f_B$. There appears to be a $f_B$ floor to the MUSE solutions, which we speculate comes from the binary nature of the magnets. The intuition that the solution can always be marginally improved by placing another magnet appears likely to hold only if the magnet directions and magnitudes can vary continuously. The greedy solution with 40000 magnets is illustrated in figure 1 to show that much more complicated permanent magnet configurations are produced by many more $\phi$ and $\theta$-aligned dipoles, but these changes have almost no impact on $f_B$.

Lastly, for all the optimizers, figure 1 exhibits some localized and regular $B \cdot n$ residuals on the plasma surface. The two local peaks at $\phi = 0$ and $\phi = \pi$, clearly visible in the FAMUS and GPMO radial-direction-only results, come from the MUSE grid. The grid has diagnostic ports at these locations, and it is challenging to compensate for these holes in the grid with radial dipoles placed nearby.

The plasma surface also exhibit stripes and other regular structures in most of the $B \cdot n$ residual plots, but it is hard to conclude about the source of these patterns. We posit that these patterns are once again reflective of the available locations on the MUSE grid, which is a set of nested toroidal shells. A more geometrically complex grid that mimics the plasma surface might provide permanent magnet locations that can better address these errors.

However, some localized errors are probably an inevitable consequence of using permanent magnets to minimize the global $f_B$ metric. The inverse cubic fall-off of a dipole field fundamentally implies that the magnets can only contribute quite localized fields to the plasma surface. It would be interesting and feasible to penalize additional terms related to, for example, the smoothness of the $B \cdot n$ residual on the plasma surface since this would amount to another linear least-squares term in the optimization. Another way to potentially address this issue is to use the full near-field expressions for permanent magnets. For rectangular cubes, the near-field expressions remain linear in the dipole moments, and therefore the optimization structure remains unchanged [45].

3.2. GPMOM results

Next, we compare the GPMOM approach against the baseline algorithm. Recall that the motivation for GPMOM is to produce solutions with very few isolated dipoles. Figure 2 illustrates the results using $N_{\text{adjacent}} = \{1, 2, 4, 7, 20\}$. The $f_B$ performance decreases fairly rapidly with increasing $N_{\text{adjacent}}$ because we are performing an increasingly coarse-grained optimization. However, if we regard each set of $N_{\text{adjacent}}$, identically grid-aligned magnets as a single larger permanent magnet, $f_B$ actually improves with increasing $N_{\text{adjacent}}$, although the $f_B$ performance floor still decreases. Examining the $m_r$ component of the GPMOM permanent magnet solutions shows that increasing $N_{\text{adjacent}}$ does lead to bigger magnetic chunks used in the solution. Interestingly, as $N_{\text{adjacent}}$ increases, so does the number of dipoles that have an equal and oppositely-oriented dipole directly adjacent, further motivating the backtracking approach from section 2.5. This property of the solutions may in fact be responsible for the reduced $f_B$ performance at larger $N_{\text{adjacent}}$; it is easier to make greedy mistakes with a coarser grid of larger magnets.
Figure 2. Summary of the GPMO algorithm results for varying $N_{\text{adjacent}}$. The $f_B$ performance uniformly worsens as a function of the individual magnets, but the $f_B$ performance is actually substantially improved if measured by the number of ‘magnet-equivalents’, i.e. treating each set of $N_{\text{adjacent}}$ identically aligned dipoles as a single, larger permanent magnet. Some of the visualizations give the appearance of magnet chunks containing fewer than $N_{\text{adjacent}}$ dipoles, but this was checked thoroughly and determined to be only a symptom of the viewing angle.

3.3. GPMO results

Here, we show that the greedy algorithm with backtracking actually consistently outperforms the state-of-the-art algorithms on the MUSE grid, and further below we repeat this achievement in section 3.5 on another permanent magnet configuration. Backtracking near cancellations, or even not-so-near cancellations, is a very effective way to generate high-quality solutions from greedy permanent magnet optimization. Systematically increasing $N_{\text{adjacent}}$ will sweep out a valley of solutions, since initially increasing $N_{\text{adjacent}}$ will improve solutions from the backtracking, but the limit $N_{\text{adjacent}} \rightarrow D$ will eventually cause reduced performance, since this limit requires that the algorithm use only three (instead of 6) possible directions. In other words, it will generate configurations with suboptimal directionality, e.g. producing dipoles that only point in the positive $r$ direction, the $-\theta$ direction, and the $-\phi$ direction.

Figure 3 illustrates the results of backtracking with different values of the hyperparameter $N_{\text{adjacent}}$. Small but important improvements to the MUSE solutions are available with GPMOb, such that, given the same number of magnets, GPMOb outperforms the binary FAMUS and relax-and-split solutions. Even with a very large value of $N_{\text{adjacent}} = 4000$, GPMOb can find an accurate solution. Furthermore, much larger improvements of several orders of magnitude reduction in $f_B$ appear accessible in other permanent magnet configurations, such as the NCSX example detailed below.
3.4. A rough comparison of algorithm run times

Lastly, we provide a comparison of the CPU run times for the FAMUS, relax-and-split, and GPMO algorithms for obtaining a target $f_B \sim 10^{-7} - 10^{-8}$ T$^2$ m$^2$. The CPU run time is approximated as wall time $\times$ number of CPUs, and allows us to compare run times while controlling for parallelization. We make no claims that any of the algorithms are optimally calculated or could not be sped up significantly. Conclusions about general computational speed are quite difficult to make because of a number of complications detailed below.

Significant variations in algorithm speed can occur simply from hyperparameter changes. For instance, the NCSX example described in section 3.5 is a particularly bad case for the relax-and-split algorithm, which currently relies on a suboptimization using projected gradient descent (PGD); PGD can progress very slowly when the solution lies near the constraint boundary. However, in the nonbinary, convex scenario, adding some Tikhonov regularization to the relax-and-split algorithm moves the optimum into the feasible region and then the algorithm is quite rapid.

For a direct comparison with GPMO, grid-aligned and binary FAMUS and relax-and-split solutions are presented, although generating binary solutions typically requires significant additional iterations for these algorithms. It follows that the run time estimates in table 1 for FAMUS and relax-and-split could be significantly sped up if fully binary solutions are not required. Despite these caveats, the takeaway is that GPMO calculates an accurate, binary, grid-aligned, MUSE solution much faster than the other algorithms. GPMO computes an accurate $r$-direction-only solution at least 10–15 times faster than FAMUS and an accurate grid-aligned solution at least 100 times faster than relax-and-split.

### Table 1. Approximate CPU run times (calculated roughly as wall time $\times$ number of CPUs to control for different parallelization) for the different algorithms to achieve $f_B \sim 10^{-7} - 10^{-8}$ T$^2$ m$^2$ using the MUSE permanent magnet manifold.

| Algorithm           | Time (h) | Binary | Grid-aligned |
|---------------------|----------|--------|---------------|
| GPMO                | 13       | ✓      | ✓             |
| GPMOb               | 27       | ✓      | ✓             |
| GPMO (r-only)       | 6        | ✓      | ✓             |
| FAMUS (r-only)      | 100      | ✓      | ✓             |
| Relax-and-split (convex) | 1   | ✓      | ×             |
| Relax-and-split     | 1000     | ✓      | ✓             |

3.5. Significant performance gains with greedy backtracking on more complicated grids

To conclude the results section, we illustrate an example for which no algorithm to date has been able to generate an accurate, binary solution. This list includes the GPMO algorithm, and seems to suggest fairly strong evidence that there are permanent magnet grids that do not support binary solutions that can minimize $f_B$ to tolerable levels. There are a number of potential solutions available by altering the grid: defining grids that are more complex, closer to the plasma, or allow for more permanent magnet volume. Weakly breaking the binary constraint appears to be another way to obtain more accurate solutions. However, we show below that the greedy algorithm with backtracking can significantly improve the GPMO solution to tolerable levels, and we obtain the best binary solution for this grid by an order of magnitude in $f_B$.

NCSX was a planned quasi-axisymmetric stellarator that was partially built at the Princeton Plasma Physics Laboratory. It was originally designed with 18 modular coils and 18 planar coils. The equilibrium of interest, C09R00, was scaled to have an on-axis magnetic field strength of 0.5 T, which is the maximum field produced by the existing planar coils. C09R00 also exhibits a three-fold field symmetry, major radius of 1.44 m, minor radius of 0.32 m, and volume-averaged plasma beta $\langle \beta \rangle = 4.09\%$. However, for direct comparison with the FAMUS solution in [17] and the relax-and-split solution in [21], the C09R00 shape is used but with no plasma current, i.e. $\langle \beta \rangle = 0$, and the toroidal field was taken to be perfectly toroidal with no ripple. The coordinate system is cylindrical and the same grid is used for all the algorithms. The grid has a resolution of 14 points radially and 57,344 locations ($\times 2 N_{fp}$ for all $N_{fp}$ field periods), so $A \in [\mathbb{R}^{4096 \times 172032}$, FAMUS is used with a level of regularization for a plausible but nonbinary and non-grid-aligned solution $f_B \approx 1.6 \times 10^{-6}$ T$^2$ m$^2$ and relax-and-split omits Tikhonov regularization with a similar $f_B \approx 1.6 \times 10^{-6}$ T$^2$ m$^2$ from a weakly nonbinary and cylindrical-coordinate-aligned solution. The relax-and-split algorithm produces both a fully binary and weakly nonbinary solution (both aligned to cylindrical coordinates), but the fully binary solution achieves only $f_B \approx 5 \times 10^{-4}$ T$^2$ m$^2$. In order to compare the nonbinary solutions with the a-priori binary relax-and-split and greedy solutions, we define the effective volume of the permanent magnet region,

$$V_{eff} = \frac{1}{M_0} \sum_{j=1}^{D} ||m_j||^2,$$

where $M_0$ is the maximum magnetization of the magnets. FAMUS and relax-and-split both produce $V_{eff} \approx 2.3$ m$^3$. To measure how binary a solution is, we define the binary fraction

$$f_\delta = 1 - \frac{\#\{|i| \delta \leq ||m_i||_2 \leq 1 - \delta\}}{D}.$$

FAMUS exhibits $f_{0.01} = 0.57$ and the nonbinary relax-and-split solution exhibits $f_{0.01} = 0.84$.

Although the NCSX grid contains magnets of differing sizes, for simplicity Tikhonov regularization was omitted for the GPMO and GPMOb results presented in this section. All of the algorithm solutions are plotted together in figure 4. Even with many binary magnets, GPMO is not able to get below $f_B \sim 10^{-4}$ T$^2$ m$^2$, which is not sufficient for an accurate solution. Breaking the binary constraint, as in FAMUS or relax-and-split, allows for significantly improved $f_B$.

However, figure 4 additionally illustrates that the GPMOB results are able to obtain a much improved $f_B \sim 10^{-5}$ using
Figure 4. Illustration of the NCSX $f_B$ versus $K$ results using the greedy algorithm with and without backtracking. GPMO agrees well with the binary relax-and-split solution but GPMOb is able obtain a binary solution that outperforms other binary solutions by almost two orders of magnitude. The GPMOb $m$ solution components are illustrated for the cylindrical-coordinate-aligned optimization. The nonbinary FAMUS and relax-and-split solutions are able to produce significantly sparser and more accurate solutions than the binary methods. The initial $f_B \sim 10^{-1}$ is improved by more than four orders of magnitude by GPMOb, and achieves $b_n \approx 0.1\%$.

The $N_{\text{adjacent}} = 100$, albeit at somewhat larger values of $V_{\text{eff}} \approx 3.27 \text{ m}^{-3}$. This is the best binary solution for this configuration by more than an order of magnitude in $f_B$, and it easily meets the engineering tolerance for NCSX, defined in the PM4STELL collaboration [46, 47] as $b_n < 0.5\%$ (their definition of $b_n$ differs slightly with equation (3) but we found empirically that both definitions produce roughly the same value). Below this threshold the plasma boundary and other plasma properties agreed well with the reference equilibrium. The work by Hammond et al [26] performs permanent magnet optimization for a similar NCSX example, using finite $\beta$ and a set of rectangular permanent magnets, and is able to achieve marginally improved performance for $f_B$. However, it is hard to conclude about the relative merits of the algorithms since their method achieves improved performance but uses more directional degrees of freedom, is not fully binary, and requires a much larger $V_{\text{eff}} = 5.74 \text{ m}^{-3}$. Moreover, the volume may be larger primarily because of finite $\beta$ and the differences in the magnet geometry and orientations.

It is possible that FAMUS and relax-and-split can generate similarly high-performance binary solutions at these larger values of $V_{\text{eff}}$, but it was the speed and greedy design of GPMOb that facilitated exploring this higher-volume, less sparse part of the parameter space anyways. The illustration of the components of the $m$ solution reveals geometrically interesting magnet structures generated by the backtracking.

Despite the improvements with GPMOb, the single discovered high-quality solution is not particularly sparse. This example provides some evidence that the advantageous engineering properties of all-maximum-magnitude dipoles may not be worth a likely reduction in $f_B$ accuracy, reduction in sparsity, or increase in magnet volume. Purchasing a small number of weak permanent magnets along with a large number of binary permanent magnets may be a useful way to reduce $f_B$ significantly while still facilitating a magnet configuration that is low-cost and low-complexity.

3.6. Why do greedy algorithms seem to be so effective for permanent magnet optimization?

An explanation is warranted for GPMO’s strong performance. Part of the explanation derives from the strong ill-posedness of the problem, which itself derives from the grid satisfying $D \gg 1$. Unlike in many other optimization problems, there are no optimization variables such that, once a variable is determined, the solution is permanently and unrecoverably sub-standard. This capability for recovery comes from several sources: each dipole independently (no coupling between magnets) contributes a very small amount to minimizing $f_B$, there are no unique and irreplaceable dipole locations (this may not be the case for a permanent magnet grid where some permanent magnets are much larger or much smaller than others, or for a grid with complex or sharp geometrical features), and there are a very large number of degrees of freedom available to either drown out the contribution of a poorly-placed magnet or place an equal and oppositely-oriented magnet directly next to it. So the greedy solutions may use more magnets than necessary, but do not seem to suffer, as in many other scientific fields, from getting inexorably trapped in bad local minima.
Indeed, similar explanations hold for the greater performance of the GPMOb algorithm. Physically, the primary way that solutions can be non-optimized is that a dipole magnetic field is not being fully utilized for $f_B$ minimization because of the presence of a neighbor dipole field that is oppositely oriented. Some amount of oppositely oriented magnets is clearly a benefit for generating complicated magnetic fields on the plasma surface, but systematically pruning close, equal and opposite neighbors seems to work very well for improving the greedy solutions. Moreover, this backtracking is built into the greedy algorithm, i.e. there is no multi-stage process to perform, and adds minor computational cost to the calculation of the solution.

A secondary route for non-optimality appears to be partial cancellations occurring between magnets of different coordinate orientations, which can occur because the magnetic field contribution of a dipole is proportional to a term like $(m_i \cdot r_i) r_i$, where $r_i$ is the spatial location of the $i$th dipole. The GPMOb algorithm does not address this type of non-optimality, and we speculate here that this may be one reason why the $r$-direction only algorithm still outperforms the full $(r, \phi, \theta)$ GPMOb algorithm around $K \sim 10000$. Future work with more advanced backtracking could address this secondary avenue for non-optimality.

4. Discussion and conclusion

We have shown that a greedy algorithm can produce a-priori sparse, grid-aligned, binary solutions that accurately solve the permanent magnet optimization problem. Remarkably, the greedy algorithm performs similarly to or even better than the state-of-the-art permanent magnet optimizations, and qualitatively reproduces the MUSE permanent magnet configurations generated by very different algorithms. The greedy approach allows for a wide range of engineering constraints, and permanent magnet configurations on new grid geometries can be rapidly computed. The greedy solutions can additionally be used as initial conditions for more sophisticated permanent magnet optimization algorithms.

In addition, we have shown that the binary permanent magnet problem is a QPQC, the binary, grid-aligned problem is a MdQKP, and the binary, single-orientation problem is a BQP. This is a useful result because the sparse regression literature primarily focuses on algorithms and associated performance bounds for continuous optimization, while the QKP literature provides algorithms and performance bounds specifically for discrete optimization over quadratic programs. There is a large literature of exact and heuristic methods for solving variants of QKPs, and this provides a useful road map for more sophisticated greedy algorithms (such as those based on tabu search, genetic or evolutionary algorithms, etc) applied to permanent magnet optimization. Future work includes scaling these more sophisticated but still heuristic approaches to the very high-dimensional problems demanded by permanent magnet optimization for stellarators. Additional future work in this direction includes deriving faster methods for computing performance bounds.

Future work could explore alternative objective functions. It is possible that, for some permanent magnet configurations, submodular objective functions [48] could be greedily minimized that have the byproduct of tolerable MSE values. More generally, the field of stellarator optimization could benefit significantly from taking submodularity more seriously. There are a number of multi-stage optimization problems where one set of coils is optimized, followed by another set of coils or a set of permanent magnets. Many of these problems could be seen as a form of greedy optimization, and theoretical performance guarantees may be available for those that borrow from the literature on greedy algorithms.

One reason that permanent magnet optimization can be computed more easily than traditional stellarator coil optimization is that $f_B$ is a linear least-squares term with respect to the $m$ optimization variables. In principle, the simple form of $f_B$ is not required for a greedy algorithm, although then the problem is no longer a variant of QKP. This could be an interesting opportunity to try additional physics-related loss terms alongside $f_B$. Moreover, greedy coil optimization could be a very fast way to explore the optimization parameter space, although high-quality solutions and natural error correction via backtracking might be significantly harder to achieve. Lastly, there has been some interest in generating stellarator magnetic fields with a large array of passive superconductor tiles [9] but optimization was difficult because of the high-dimension and more complicated calculation for $f_B$; greedy algorithms could be useful for optimizing these arrays.

There are some weaknesses of the greedy approach in the present work. There are a number of scenarios for which the greedy approach appears ill-suited, including solutions generated with a range of continuous magnet strengths, stochastic optimization, or permanent magnet grid locations that vary widely (in terms of geometry, distance from the plasma, etc). It is also unclear if the greedy algorithms can continue to perform near-optimally when some coupling is introduced between dipoles through finite permeability.

As measured by the effective magnet volume, significantly higher accuracy and sparser solutions are available by relaxing the binary constraint, and should prompt reconsideration if this constraint is truly a requisite for any experimental permanent magnet configuration. If the binary constraints are abandoned, or at least weakly broken, continuous optimization algorithms, rather than greedy discrete algorithms such as GPMO, will be required for obtaining near-optimal solutions.

Acknowledgments

A K would like to acknowledge important conversations with Tony Qian and Stefan Buller, and coding assistance from Florian Wechsung. Thanks to Djin Patch for estimates regarding FAMUS run times. This work was supported by the U.S. Department of Energy under Awards DEFG0293ER54197 and DE-SC00022005, and through a grant from the Simons Foundation (560651, ML). This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley
National Laboratory, operated under Contract No. DE-AC02-05CH11231.

Appendix. A brief review of greedy algorithm guarantees

Greedy algorithms are powerful tools for computing fast, reasonably accurate solutions to high-dimensional optimization problems. They often have performance guarantees, and the strongest guarantees are available for objectives that are monotone, nondecreasing, and submodular. The mutual coherence is a common submodular objective to optimize that results in the orthogonal matching pursuit algorithm [49, 50]. However, the performance guarantees are guarantees on the minimization of the mutual coherence, not on finding a minimum of the MSE. So minimizing submodular functions such as the mutual coherence may not achieve sufficient MSE reduction for an accurate solution.

Fortunately, there are also non-submodular objectives that have performance guarantees [51–53], such as supermodular functions. Reasonable bounds on the performance of the MSE have only recently derived by showing it is an ‘approximately supermodular’ (or \( \alpha \)-supermodular) function. This is a potentially useful result for permanent magnet optimization, where the MSE is the fundamental objective that we would like to minimize, and it is essential that the MSE is greatly reduced on the plasma surface.

Unfortunately, given a large number of optimization variables, computing reasonably strict lower bounds on the performance is also a hard problem. An additional complication comes from the binary and grid-aligned nature of GPMO; these assumptions may alter some of the performance bound analysis coming from the sparse regression literature. Fortunately, for the binary case there are reasonable performance bounds from the QKP literature [28] that could prove useful in future work, although computing these bounds also typically requires the solution of nontrivial, high-dimensional optimization problems.

ORCID iDs

Alan A. Kaptanoglu https://orcid.org/0000-0002-6337-2907
Rory Conlin https://orcid.org/0000-0001-8366-2111
Matt Landreman https://orcid.org/0000-0002-7233-577X

References

[1] Drevlak M., Beidler C., Geiger J., Helander P. and Turkin Y. 2018 Nucl. Fusion 58 016010
[2] Lazerson S. et al 2020 Stelldopt Technical Report (Princeton, NJ: Princeton Plasma Physics Lab. (PPPL))
[3] Landreman M., Medasani B., Wechsung F., Giuliani A., Jorge R. and Zhu C. 2021 J. Open Source Softw. 6 3525
[4] Zhu C., Hudson S.R., Song Y. and Wan Y. 2017 Nucl. Fusion 58 016008
[5] Erckmann V. et al 1997 17th IEEE/NPSS Symp. Fusion Engineering (Cat. No. 97CH36134) (San Diego, CA, 6–10 October 1997) vol 1 (IEEE) pp 40–48
[6] Strykovsky R., Brown T., Chrzanowski J., Cole M., Heitzenroeder P., Neilson G.H., Rej D. and Viohl M. 2009 2009 23rd IEEE/NPSS Symp. on Fusion Engineering (San Diego, CA, 1–3 June 2009) (IEEE) pp 1–4
[7] Helander P., Drevlak M., Zarnstorff M. and Cowley S. 2020 Phys. Rev. Lett. 124 095001
[8] Zhang H.-T. and Zhang X. 2022 Mater. Res. Lett. 10 1
[9] Neilson G. et al 2011 Progress toward attractive stellarators Technical Report (Princeton, NJ: Princeton Plasma Physics Lab. (PPPL))
[10] Qian T., Zarnstorff M., Bishop D., Chamblis A., Dominguez A., Pagano C., Patch D. and Zhu C. 2022 Nucl. Fusion 62 084001
[11] Cooley C.Z., Haskell M.W., Cauley S.F., Sappo C., Lapierre C.D., Ha C.G., Stockmann J.P. and Wald L.L. 2017 IEEE Trans. Magn. 54 1
[12] Ren Z.H., Mu W.C. and Huang S.Y. 2019 IEEE Trans. Magn. 55 1
[13] Di Barba P., Mognaschi M.E., Palka R., Paplicki P. and Szolonyi S. 2012 Int. J. Appl. Electromagn. Mech. 39 889
[14] Coey J. 2002 J. Magn. Magn. Mater. 248 441
[15] Zhu C., Hammond K., Brown T., Gates D., Zarnstorff M., Corrigian K., Sibilla M. and Feibush E. 2020 Nucl. Fusion 60 106002
[16] Zhu C., Zarnstorff M., Gates D. and Brooks A. 2020 Nucl. Fusion 60 076016
[17] Landreman M. and Zhu C. 2021 Plasma Phys. Control. Fusion 63 035001
[18] Xu G., Lu Z., Chen D., Chen L., Zhang X., Wu X., Ye M. and Wan B. 2021 Nucl. Fusion 61 026025
[19] Lu Z., Xu G., Chen D., Zhang X., Chen L., Ye M., Guo H. and Wan B. 2022 Cell Rep. Phys. Sci. 3 100709
[20] Lu Z., Xu G., Chen D., Chen L., Zhang X., Ye M. and Wan B. 2021 Nucl. Fusion 61 106028
[21] Kaptanoglu A.A., Qian T., Wechsung F. and Landreman M. 2022 Phys. Rev. Appl. 18 044006
[22] Zheng P. and Aravkin A. 2020 Inverse Problems 36 095013
[23] Champion K., Zheng P., Aravkin A.Y., Brunton S.L. and Kutz J.N. 2020 IEEE Access 8 169259
[24] Kaptanoglu A.A., Morgan K.D., Hansen C.J. and Brunton S.L. 2021 Phys. Rev. E 104 015206
[25] Kaptanoglu A.A., Callaham J.L., Aravkin A., Hansen C.J. and Brunton S.L. 2021 Phys. Rev. Fluids 6 094401
[26] Hammond K., Zhu C., Corrigian K., Gates D., Lown R., Mercurio K., Qian T. and Zarnstorff M. 2022 arXiv:2204.07648
[27] Bruckstein A.M., Donoho D.L. and Elad M. 2009 SIAM Rev. 51 34
[28] Pisinger D. 2007 Discrete Appl. Math. 155 623
[29] Cacciabue V., Iori M., Locatelli A. and Martello S. 2002 Comput. Oper. Res. 143 105693
[30] Wang H., Kochenberger G. and Glover F. 2012 Comput. Oper. Res. 39 3
[31] Kochenberger G., Hao J.-K., Glover F., Lewis M., Lü Z., Wang H. and Wang Y. 2014 J. Comb. Optim. 28 58
[32] Palubeckis G. 2006 Informatica 17 279
[33] Boros E., Hammer P.L. and Tavares G. 2007 J. Heuristics 13 99
[34] Glover F. 2013 Eur. J. Oper. Res. 230 212
[35] García-Martínez C., Rodriguez F.J. and Lozano M. 2014 Eur. J. Oper. Res. 232 454
[36] Qin J., Xu X., Wu Q. and Cheng T. 2016 Comput. Oper. Res. 66 199
[37] Julstrom B.A. 2005 Proc. 7th Annual Conf. on Genetic and Evolutionary Computation (Washington DC, USA, 25–29 June) pp 607–14
[38] Kernighan B.W. and Lin S. 1970 Bell Syst. Tech. J. 49 291
[39] Merz P. and Freisleben B. 2002 J. Heuristics 8 197
[40] Wen J. and Li H. 2021 Inverse Problems 37 065014
[41] Chamon L.F., Pappas G.J. and Ribeiro A. 2017 2017 IEEE 56th Annual Conf. on Decision and Control (CDC) (Melbourne, Australia, 12–15 December) (IEEE) pp 343–50
[42] Kohara A., Okano K., Hirata K. and Nakamura Y. 2020 2020 59th IEEE Conf. on Decision and Control (CDC) (Jeju, South Korea, 14–18 December) (IEEE) pp 1706–11
[43] Hmam H. 2010 Quadratic optimisation with one quadratic equality constraint Technical Report Defence Science and Technology Organisation Edinburgh (Australia) Electronic
[44] Wang J., Kwon S. and Shim B. 2012 IEEE Trans. Signal Process. 60 6202
[45] Xiao-Fan G., Yong Y. and Xiao-Jing Z. 2004 Appl. Math. Mech. 25 297
[46] Zhu C. et al. 2022 Phys. Plasmas 29 112501
[47] Rutkowski A., Hammond K., Zhu C., Gates D.A. and Chambliss A. 2023 A novel scheme for error field correction in permanent magnet stellarators Nucl. Fusion 63 026027
[48] Das A. and Kempe D. 2011 Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection (arXiv:1102.3975)
[49] Pati Y.C., Rezaiifar R. and Krishnaprasad P.S. 1993 Proc. 27th Asilomar Conf. on Signals, Systems and Computers (Pacific Grove, 1–3 November) (IEEE) pp 40–44
[50] Tropp J.A. and Gilbert A.C. 2007 IEEE Trans. Inf. Theory 53 4655
[51] Bian A.A., Buhmann J.M., Krause A. and Tschiatschek S. 2017 Int. Conf. on Machine Learning (PMLR) (Sydney, Australia, 6–11 August) pp 498–507 (available at: http://proceedings.mlr.press/v70/)
[52] Bogunovic I., Zhao J. and Cevher V. 2018 Int. Conf. on Artificial Intelligence and Statistics (PMLR) (Playa Blanca, Lanzarote, Canary Islands, 9–11 April) pp 890–9 (available at: https://proceedings.mlr.press/v84/)
[53] Summers T. and Kamgarpour M. 2019 2019 18th European Control Conf. (ECC) (Limassol, Cyprus, 25–28 June) (IEEE) pp 2796–801