Magnetic Actuation With Stationary Electromagnets Considering Power and Temperature Constraints

Ashkan Pourkand, *Student Member, IEEE*, and Jake J. Abbott, *Senior Member, IEEE*

***Abstract*—In systems comprising stationary electromagnets designed for magnetic manipulation and remote magnetic actuation, the standard method for achieving some desired system output (e.g., magnetic force and torque) is to form a matrix that maps the electromagnets' electrical currents to the resulting output, and then use the pseudoinverse of that matrix to solve for the currents. However, this method does not account for the nonlinear nature of the saturation limits of the amplifiers and power supply, nor does it account for the temperature in the electromagnets, both of which limit the system's overall performance in practice. In this letter, we propose improved current-solving methods that account for these real-world system limitations. We demonstrate how the performance of systems can be improved by increasing both the achievable output magnitudes and the total operating time to achieve those outputs, and how the risk of overheating can be eliminated.**

***Index Terms*—Medical robots and systems, micro/nano robots, optimization and optimal control, redundant robots.**

I. Introduction

After decades of research activity, there is now an extensive literature in the development and control of systems for magnetic manipulation and remote magnetic actuation [1]. These systems have often utilized a set of stationary electromagnets surrounding a workspace to apply force and torque to a (typically) ferromagnetic object in the workspace, with these objects ranging from medical devices inside the human body to microrobots under an optical microscope to untethered haptic interfaces.

It was shown in [2] how such a system, when acting on a permanent-magnet object, can be described by

$$Y = \mathbb{A}I \tag{1}$$

where $I \in \mathscr{R}^n$ (units A) is an array of currents flowing through the n electromagnets, $Y \in \mathscr{R}^m$ is the output of interest, and $\mathbb{A} \in \mathscr{R}^{m\times n}$ is a configuration-dependent so-called actuation matrix. One commonly used output considered for actuation is $Y = [\boldsymbol{\tau}^\top \ \boldsymbol{f}^\top]^\top \in \mathscr{R}^6$, where $\boldsymbol{\tau} \in \mathscr{R}^3$ (units N·m) and $\boldsymbol{f} \in \mathscr{R}^3$ (units N) are the resulting torque and force vectors, respectively. Other actuation matrices are possible, such as mapping currents to field and force rather than torque and force [1], but we can assume (1) without loss of generality. This model is valid for electromagnets with no ferromagnetic cores (i.e., air cores), or for systems with ferromagnetic cores, provided that the cores are not magnetically saturated. In practice, most systems are likely to meet one of these assumptions.

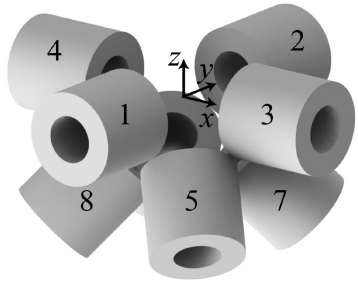

Fig. 1. The OctoMag configuration of electromagnets, which will be used as a test case throughout this paper. The center of the workspace is defined as the location where each of the electromagnet axes are coincident. Each coreless electromagnet is modeled with an outer radius $R = 22$ mm, an inner radius $R/2$, and a length $2R$. The face of each electromagnet is located at a distance of $1.7R$ from the center of the workspace. The wires are modeled with cross-sectional area of 0.52 mm^2, with a packing efficiency of 1. As a result, a current i (units A) is mapped to a current density j (units A·mm^{-2}) as $j = 1.9i$. Each coil has an amplifier with a saturation of 8.0 A (continuous). The power supply has a saturation of 30.0 A (continuous).

It was also shown in [2] that a viable method to solve for the currents in order to generated some desired output Y_{des} is to simply use the pseudoinverse (i.e., the Moore-Penrose generalized inverse) of $\mathbb{A}$:

$$I = \mathbb{A}^\dagger Y_{\text{des}} \tag{2}$$

Note that, although (2) is how we typically conceptualize the method, in practice one may choose to simply solve (1) using numerical methods. This method was first applied on the OctoMag system [2], which comprises eight electromagnets (see Fig. 1). It was later shown that at least eight electromagnets are required to ensure that there is always a solution to achieve some $Y_{\text{des}} = [\boldsymbol{\tau}_{\text{des}}^\top \ \boldsymbol{f}_{\text{des}}^\top]^\top$ in any configuration of the object being actuated [3]. In a given configuration, there is a null space in the solver with the potential for use in optimization. By using more than eight electromagnets, magnetic-actuation systems can achieve a better-conditioned workspace and increase the dimension of the null space.

Manuscript received May 19, 2020; accepted September 10, 2020. Date of publication September 25, 2020; date of current version October 7, 2020. This letter was recommended for publication by Associate Editor Y. Zhao and Editor D. Song upon evaluation of the Reviewers' comments. This work was supported by the National Science Foundation under Grant 1423273. *(Corresponding author: Ashkan Pourkand.)*

Ashkan Pourkand is with the School of Computing and the Robotics Center, University of Utah, Salt Lake City, UT 84112 USA (e-mail: ashkan.pourkand@gmail.com).

Jake J. Abbott is with the Department of Mechanical Engineering and the Robotics Center, University of Utah, Salt Lake City, UT 84112 USA (e-mail: jake.abbott@utah.edu).

Digital Object Identifier 10.1109/LRA.2020.3025512

The pseudoinverse of (2) has the desirable property that it returns the solution that minimizes $I^\top I$, which corresponds to a minimization of the instantaneous power consumption and heat generation if all of the electromagnetic coils have the same electrical resistance. Due to its effectiveness and simplicity, this method has essentially become the standard that has been used in many other works with only minor variations. However, this method does have some practical limitations.

When using the pseudoinverse of (2), the solution that is returned relies heavily on the best-conditioned electromagnets. This extensive use of a limited set of electromagnets can generate excessive heat in certain electromagnets, up to the point of needing to cease operation until temperatures return to a safe range. This phenomenon is clearly dependent on the history of current commands. In addition, when we consider temperature limits, we are not particularly concerned with the total heat in the system or the average temperature of the electromagnets; rather, we are most concerned with the temperature of the hottest electromagnets, as they alone will ultimately dictate whether the system must cease operation. Due to these temperature considerations, many magnetic-actuation systems are equipped with thermocouples embedded in the electromagnets to monitor their temperature and, in the case of overheating, shut down the system to avoid permanent damage. Magnetic-actuation systems have not typically utilized the thermal information, except in this binary fashion.

We will show how redundancy in actuation can be used to mitigate the negative effects of heating, using a weighted pseudoinverse that minimizes $I^\top \mathbb{W} I$ for a temperature-dependent weight matrix $\mathbb{W}$. This type of weighted pseudoinverse has long been used in robotic manipulation to resolve manipulator redundancy and optimize for some secondary objective [4]–[8]. In the context of magnetic-manipulation and remote magnetic actuation, this type of weighted pseudoinverse has been used to scale the inputs relative to their maximum ranges [9], which is also one of the traditional uses with robotic manipulators.

Magnetic-actuation systems are constructed with one amplifier per electromagnet, each of which has a respective current and voltage saturation limit. In addition, it is typical to power all of these amplifiers from a common power supply, which has its own current saturation limit. Previously, if (2) returned current values that violate the saturation limits of any of the amplifiers, it would be determined that the desired output was unachievable and must be attenuated by an appropriate amount so that no amplifier is above its saturation limit; we will refer to this as the traditional pseudoinverse solver (PINV). However, such a conclusion is not necessarily true. We find that in many cases in which an amplifier saturation is detected, as few as one amplifier is saturated, leaving many other amplifiers, as well as the power supply, unsaturated. The problem of power-supply saturation has not been explicitly addressed in prior work, but as the number of amplifiers powered by a single supply increases, the probability of reaching the power-supply saturation before any amplifier saturates increases.

We will show that, in cases in which (2) returns a solution that violates saturation limits, unused capacity in the remaining electromagnets can often be used to achieve the desired output. We propose that a recursive approach known as the redistributed pseudoinverse [10], with appropriate modifications, can be used to improve the peak capacity of systems, often by a substantial amount.

In this letter, we propose a new approach that combines the redistributed- and weighed-pseudoinverse approaches to provide a unified method that improves system performance by increasing both the achievable output magnitudes and the total operating time to achieve those outputs, and eliminates the risk of overheating. We also propose a cost function and temperature-dependent constraints that can be used in a constrained-optimization solver to achieve a similar result; this method results in further increases in performance, but at the cost of increased computation time.

II. Redistributed Weighted Pseudoinverse

The pseudoinverse relies heavily on the best conditioned coils, and it is memoryless. As a result, a simple actuation task such as levitating an object in place (i.e., generating a continuous upward force command) will continually utilize a limited set of coils, which will in turn result in continual heating in those coils. In addition, when a new actuation command is attempted, there is no acknowledgement of the command history and the resulting heat stored in the coils. The history of the current commands is somewhat irrelevant, in that they are only correlated with the quantities that are really of interest, which are the temperatures of the coils.

The weighted pseudoinverse (WPINV) uses a diagonal weight matrix $\mathbb{W}$ to penalizing the use of certain actuators in the pseudoinverse solution. In [9], those weights are used to scale the inputs with respect to their full-scale values. Here, we would also like to make those weights be a function of the coils' temperatures. Since the pseudoinverse is optimally efficient in the sense that it uses the most well-conditioned coils for any given command, it may be counterproductive to excessively intervene in that process. As a result, we have chosen to only implement penalties when the temperature $T(i)$ in coil i crosses some user-specified critical temperature $T_c(i)$:

$$\mathbb{W}(i,i) = \begin{cases} I_{\mathrm{a}}(i)^{-2} e^{\xi(T(i)-T_c)}, & \text{if } T(i) \geq T_c \\ I_{\mathrm{a}}(i)^{-2}, & \text{if } T(i) < T_c \end{cases} \tag{3}$$

where $I_{\mathrm{a}}(i)$ is the magnitude of the saturation limit of amplifier i and ξ is user-specified constant that governs how aggressively the penalty increases with an increase in temperature beyond the critical temperature. The values for T_c and ξ should be chosen using engineering judgement, but the critical temperature T_c must certainly be less than the value provided by the wire manufacturer at which breakdown of the wire's insulation occurs, with some factor of safety. The critical temperature should not be interpreted as the maximum temperature that will ever be reached in the coil; rather, it is the temperature below which no temperature-related intervention will be made by the solver. An additional check should be employed to shut down the system if any coil ever reaches some maximum temperature $T_{\max}$.

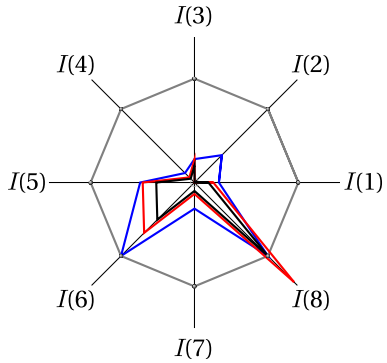

Fig. 2. The eight currents calculated by WPINV and RWPINV in a typical saturation event with the system of Fig. 1. The gray octagon depicts the saturation limit of each coil. The currents calculated by (2) are shown as the red net, which is beyond the saturation limit of coil 8. The attenuated solution of WPINV, which perfectly saturates coil 8, is shown as the black net; however, there is a proportional attenuation of the generated output relative to the desired output. The blue net shows the solution using RWPINV, which achieves the desired output.

In the weighted pseudoinverse [4], currents are calculated as

$$I = \mathbb{W}^{-1}\mathbb{A}^{\top}\left(\mathbb{A}\mathbb{W}^{-1}\mathbb{A}^{\top}\right)^{\dagger} Y_{\text{des}} \tag{4}$$

where we have replaced the inverse of the $\mathbb{A}\mathbb{W}^{-1}\mathbb{A}^{\top}$ term that was originally proposed in [4] with a pseudoinverse to ensure a solution is returned even in cases in which $\mathbb{A}$ becomes rank deficient and Y_{des} cannot be perfectly achieved.

Next, we incorporate the redistributed pseudoinverse, which will enable us to increase the achievable performance of magnetic-actuation systems beyond the linear saturation limit. That is, it will enable us to achieve instantaneous outputs (e.g., torques, forces) of higher magnitude by accounting for the saturation limits of the amplifiers and the power supply. We refer to the resulting method that combines the weighted- and redistributed-pseudoinverse as the redistributed weighted pseudoinverse (RWPINV).

Let us first qualitatively consider how WPINV (or PINV) behaves if a saturation event is detected. The red net in Fig. 2 depicts a typical example in which the solution to achieve some Y_{des} using (4) returns a solution for I in which the current in coil 8 is beyond its amplifier's limit, although the other seven coils are far below their maximum limits. To deal with this issue, WPINV would attenuate the magnitude of the requested Y_{des}, which attenuates the required currents in I proportionally, until coil 8 barely saturates, as depicted by the black net. The result is that the other seven coils are even farther from their maximum limits. In contrast, RWPINV, shown as the blue net, is able to achieve the original Y_{des} by considering the saturation limits more holistically.

Incorporating redistribution works as follows. In the event of current saturation in an amplifier when using (4) to solve for I, we first attenuate the requested Y_{des} linearly by a constant $0 < \delta < 1$ such that no current in δI is above its saturation limit and at least one current is at its saturation limit. We then verify that $\|\delta I\|_1$ does not exceed the limit of the power supply; if it does exceed the limit then δ is further reduced to the proper value to avoid exceeding the limit, the process is stopped, and $Y = \delta Y_{\text{des}}$. Until this point, the method is identical to WPINV.

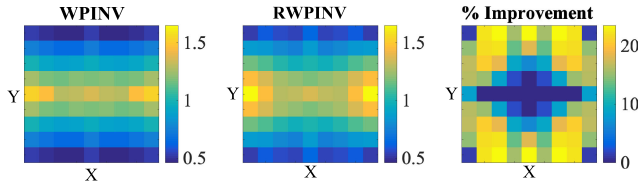

Fig. 3. Comparison of the maximum achievable force magnitude (units N) with WPINV and RWPINV, as well as the percentage increase when using RWPINV, for the plane $x \in [-R,\ R]$, $y \in [-R,\ R]$, $z = 0$ in the system of Fig. 1, in which a torque-free force is commanded in the $+z$ direction on a unit dipole that is oriented in the $+x$ direction, and none of the coils are overheated.

Let us proceed assuming the power-supply limit has not yet been reached. The resulting $Y = \delta Y_{\text{des}}$ has a shortfall from the desired Y_{des}, and we would like to make up for as much of this shortfall as possible using the remaining unsaturated coils. To do this, we recursively call (4) after making the following modifications: the column corresponding to the saturated coil is removed from $\mathbb{A}$; the available current intervals are shifted by subtracting δI from the previous intervals (this effectively creates new positive and negative saturation limits for each coil); and we set the new desired output as $(1-\delta)Y_{\text{des}}$. We again check for the power-supply saturation limit, following the same procedure described above. We continue this process recursively until either the current solution does not have any saturated values (indicating that we found a solution to achieve Y_{des}) or $\mathbb{A}$ runs out of null space (indicating that the requested Y_{des} is not achievable, and the sum of attenuated Y_{des} values from the previous steps is the maximum achievable output). Since the output is m-dimensional, the n-dimensional input provides an $(n-m)$-dimensional null space, capping the number of recursive calls at $n-m$, which will typically be a small number (e.g., $8-6=2$).

The improvement that can be expected from this method is configuration and command dependent. For example, Fig. 3 shows how RWPINV quantitatively compares to WPINV when actuating a unit-magnitude dipole (e.g., permanent magnet) throughout the workspace of the OctoMag system of Fig. 1 for one specific command that represents the levitation of a permanent-magnet object. We see that in some of these configurations RWPINV provides an increase in output magnitude of as much as 27%, whereas in other configurations the improvements were negligible, primarily due to the current saturation limit in the power supply. This example should not be assumed to have captured the cases in which RWPINV provides the largest benefit. Nevertheless, RWPINV will never perform worse than WINV, and it comes with only a small increase in computational cost.

III. Constrained Optimization

Although RWPINV provides a better solutions than PINV, it is not guaranteed to provide the optimal solution (where "optimal" is not particularly well defined). Another approach is to use an optimization approach in which some cost function

is minimized, subject to constraint equations. It is not trivial to balance our competing goals of minimizing output error and minimizing current consumption while also respecting amplifier, power supply, and temperature constraints, but we believe we have arrived at a good formulation of this problem:

$$\begin{aligned} \min_{I} \quad & (Y - Y_{\text{des}})^{\top} (Y - Y_{\text{des}}) \\ & + \frac{\epsilon I^{\top} \mathbb{W}_{\text{a}} I}{1 + ((Y - Y_{\text{des}})^{\top}(Y - Y_{\text{des}}))^{\kappa}} \\ \text{s.t.} \quad & \sum_{i=1}^{n} |I(i)| \leq I_{\text{p}} \\ & \frac{-I_{\text{a}}(i)}{w_T(i)} \leq I(i) \leq \frac{I_{\text{a}}(i)}{w_T(i)} \quad \text{for } i = 1, \ldots, n \end{aligned} \tag{5}$$

where I_{p} is the saturation limit of the power supply, $\mathbb{W}$ is the diagonal weight matrix that scales the currents relative to the amplifier saturation limits:

$$\mathbb{W}_{\text{a}}(i,i) = I_{\text{a}}(i)^{-2} \tag{6}$$

and

$$w_T(i) = \begin{cases} e^{\xi(T(i) - T_c)}, & \text{if } T(i) \geq T_c \\ 1, & \text{if } T(i) < T_c \end{cases} \tag{7}$$

are temperature-dependent weights that modify the amplifier saturation constraints. The cost function prioritizes minimizing the output error when it is large, and as the output error gets small, the cost function also tries to minimize the current consumption; $\epsilon > 0$ and $\kappa > 0$ are user-specified constants used to control the relative importance of the two terms.

IV. Comparison of Methods

In this section, we compare the relative performance of the proposed methods—in terms of output error, temperature regulation, and running time—with that of the typical PINV solver. All simulations are run in MATLAB R2019a on a 3.70 GHz core i7-8700k processor. For the implementation of the constrained-optimization method, we use MATLAB's fmincon function configured to run the sequential quadratic programming (SQP) algorithm, with *StepTolerance* $= 1\text{e}-4$, *Display* $=$ off. We use $\xi = 0.2$, $\epsilon = 10^{-5}$, and $\kappa = 1$, which we found provides a desirable response. We employ a simple thermal model, provided in Appendix A. We arbitrarily set the critical temperature for all coils at $T_c = 150°$.

We begin by considering a step actuation command of $Y_{\text{des}} = [\boldsymbol{\tau}_{\text{des}}^{\top} \; \boldsymbol{f}_{\text{des}}^{\top}]^{\top} = [0\,0\,0\,0\,0 \,\|\boldsymbol{f}_{\text{des}}\|]^{\top}$ on a dipole $\boldsymbol{m} = [1 \;\; 0 \;\; 0]^{\top}$ located at position $[x\,y\,z] = [-0.5R \;\; -R \, 0]$ in the OctoMag system of Fig. 1. By successively increasing $\|\boldsymbol{f}_{\text{des}}\|$ from 0.3 mN to 0.6 mN (without loss of generality), we were able to identify five distinct regimes, which we identify as ordinal difficulty levels DL1 through DL5, which are depicted in Fig. 4 and summarized in Table I.

In the DL1 regime ($\|\boldsymbol{f}_{\text{des}}\| = 0.3$ mN), all methods were able to achieve the desired force. In addition, all methods could sustain that force indefinitely without reaching the critical temperature.

TABLE I
Summary of the Results of Fig. 4 in Terms of Force Generation, Temperature Regulation, and Running Time. For Force Generation, a Smile Indicates That the Desired Force Was Achieved, a Frown Indicates That It Was Not Achieved, and a Timer Indicates That It Was Achieved For a Limited Time. For Temperature Regulation, a Smile Indicates That the Temperature was Always Safe, and a Warning Sign Indicates That the Temperature Eventually Exceeds Safety Limits

		DL1	DL2	DL3	DL4	DL5
PINV	$\|\boldsymbol{f}\|$	☺	☺	☹	☹	☹
	$\|T\|_\infty$	☺	⚠	⚠	⚠	⚠
	t (s)	5.4e-5	5.6e-5	5.6e-5	5.6e-5	5.6e-5
RWPINV	$\|\boldsymbol{f}\|$	☺	☺	☺	⏲	☹
	$\|T\|_\infty$	☺	☺	☺	☺	☺
	t (s)	8.5e-5	8.5e-5	1.0e-4	1.1e-4	1.2e-4
SQP	$\|\boldsymbol{f}\|$	☺	☺	☺	☺	⏲
	$\|T\|_\infty$	☺	☺	☺	☺	☺
	t (s)	1.6e-2	3.5e-2	3.7e-2	3.8e-2	5.6e-2

In the DL2 regime ($\|\boldsymbol{f}_{\text{des}}\| = 0.4$ mN), all methods were still able to achieve the desired force. However, PINV resulted in a coil passing the critical temperature unchecked after $\sim$ 3 minutes, whereas RWPINV held the temperature to 150 °C indefinitely, and SQP kept all of the coils below the critical temperature.

In the DL3 regime ($\|\boldsymbol{f}_{\text{des}}\| = 0.5$ mN), PINV was no longer able to achieve the desired force, but all other methods could. PINV resulted in a coil passing the critical temperature unchecked after $\sim 1\frac{1}{2}$ minutes, whereas RWPINV and SQP held the temperature indefinitely to 151 °C and 150 °C, respectively.

In the DL4 regime ($\|\boldsymbol{f}_{\text{des}}\| = 0.55$ mN), PINV was never able to achieve the desired force, SQP was able to achieve the desired force indefinitely, and RWPINV was only able to achieve the desired force for a limited time, eventually reducing itself to a sustainable force of $\|\boldsymbol{f}\| = 0.51$ mN. PINV resulted in a coil passing the critical temperature unchecked after $\sim 1\frac{1}{2}$ minutes, whereas RWPINV and SQP held the temperature indefinitely to 151 °C and 150 °C, respectively.

In the DL5 regime ($\|\boldsymbol{f}_{\text{des}}\| = 0.6$ N), only SQP was ever able to achieve the desired force, but it eventually reduced itself to a sustainable force of $\|\boldsymbol{f}\| = 0.58$ mN. RWPINV was able to achieve a larger force ($\|\boldsymbol{f}\| = 0.57$ mN) than could PINV, but was only able to achieve this maximum force for a limited time, eventually reducing itself to a sustainable force of $\|\boldsymbol{f}\| = 0.51$ mN. PINV resulted in a coil passing the critical temperature unchecked after $\sim 1\frac{1}{2}$ minutes, whereas RWPINV

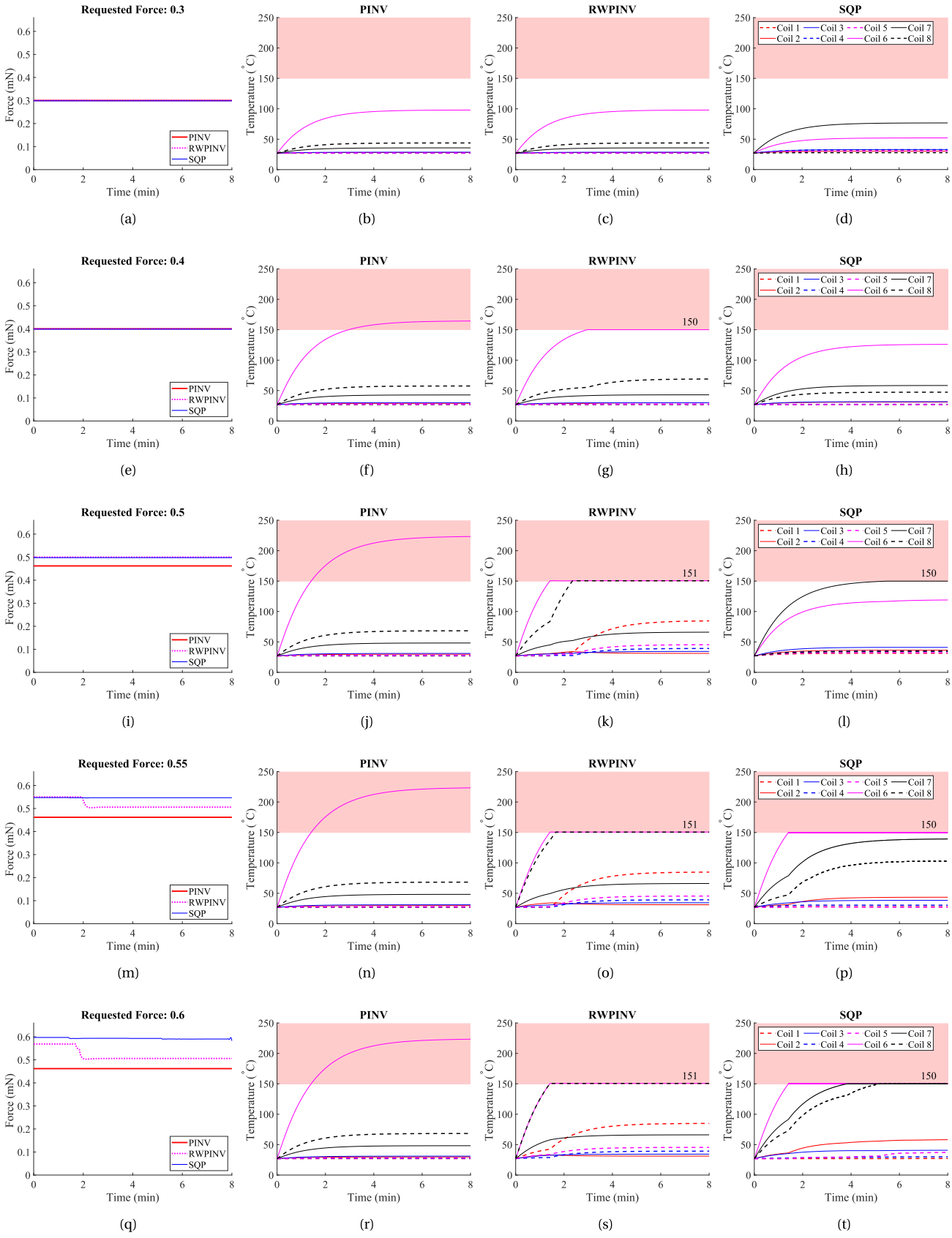

Fig. 4. Comparison of PINV, RWPINV, and SQP in force generation and temperature regulation in the system of Fig. 1. Each row represents a different requested force and difficulty level. In the temperature plots, the eight individual coil temperatures are shown, and the critical temperature is 150 °C. Results are summarized in Table I.

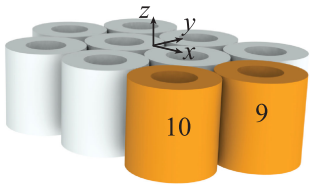

Fig. 5. A planar coil array, as in [11]. The coils, amplifiers, and power supply are modeled the same as those of the system of Fig. 1.

and SQP held the temperature indefinitely to 151 °C and 150 °C, respectively.

There is, of course, another regime (which we could call DL6) in which the commanded force is so large that no method can ever achieve it. In this regime, the results are qualitatively similar to that of DL5.

If we consider the running time of each of the methods, we find that running time tends to increase slightly with difficulty level for every new proposed method, that RWPINV requires approximately 2× running time compared to PINV, and that SQP requires approximately 1000× running time compared to RWPINV in some cases.

Each of the proposed methods utilizes redundancy (i.e., the null space in actuation) to improve system performance. As such, it would be reasonable to assume that an increase in the number of coils would lead to an increase in performance, but potentially at increased computational costs. To explore this, we consider the planar array of coils depicted in Fig. 5, inspired by [11]. We consider a step actuation command of $Y_{\text{des}} = [\boldsymbol{\tau}_{\text{des}}^{\top}\ \boldsymbol{f}_{\text{des}}^{\top}]^{\top} = [0\,0\,0\,0\,0\,\|\boldsymbol{f}_{\text{des}}\|]^{\top}$ on a dipole $\boldsymbol{m} = [1\ 0\ 0]^{\top}$ located at position $[x\,y\,z] = [0\,0\,0]$. We begin with an eight-coil array, and then increase to nine and ten, successively. In each case, we determine the maximum value of $\|\boldsymbol{f}_{\text{des}}\|$ that is achievable (to within 1% error). The results are provided in Table II, along with the angular force error, the torque magnitude (recall that no torque is desired), and the running time. We observe that the running time is insensitive to the number of coils in the RWPINV method, whereas the running time increases with a coil increase when using SQP. SQP also results substantially larger angular errors in the force, and undesired torques.

TABLE II
SUMMARY OF RESULTS WITH THE SYSTEM OF FIG. 5 AS A FUNCTION OF THE NUMBER OF ACTIVE COILS

		Numbers of coils		
		8	9	10
PINV	$\|\boldsymbol{f}\|$	0.54	0.58	0.64
	Error(°)	7.4e-17	2.6e-17	1.9e-18
	$\|\boldsymbol{\tau}\|$	3.8e-14	1.6e-14	2.5e-14
	t (s)	5.1e-5	5.5e-5	5.6e-5
RWPINV	$\|\boldsymbol{f}\|$	0.59	0.60	0.64
	Error(°)	8.0e-17	2.6e-17	8.7e-19
	$\|\boldsymbol{\tau}\|$	2.6e-14	2.5e-14	2.5e-14
	t (s)	9.5e-5	1.1e-4	1.1e-4
SQP	$\|\boldsymbol{f}\|$	0.71	0.71	0.71
	Error(°)	3.2	3.1	0.25
	$\|\boldsymbol{\tau}\|$	2.9e-3	2.9e-3	2.6e-3
	t (s)	4.0e-2	5.2e-2	5.8e-2

V. DISCUSSION

We have introduced a new method, which we refer to as the redistributed weighted pseudoinverse (RWPINV), to solve for the required currents to achieve a given actuation command in electromagnet systems designed for magnetic manipulation and remote magnetic actuation. The complete pseudocode to implement this new method is provided in Appendix B. We believe this new method should replace the pseudoinverse as the *de facto* solver since it: (1) protects the system from overheating, (2) can often increase the operating time to achieve a desired actuation command, (3) can often increase the magnitude of achievable actuation commands, (4) never performs worse than a traditional pseudoinverse solver, and (5) has minimal additional computational cost for modern microprocessors.

We have also proposed a cost function and temperature-dependent constraints to be used with a constrained-optimization solver, which showed even better performance, although at an substantial increase in computation cost. This method can be employed in instances in which the running time does not preclude its use. It should be noted that it may be possible to reduce the running time by improving the efficiency of the solver and by providing it with a good initial guess (if continuity can be assumed). However, it should also be noted that the output Y is not guaranteed to point in the same direction as Y_{des}, and this method has a number of free parameters that may need to be tuned for a given system.

Many systems are equipped with some form of temperature sensing (e.g., thermocouples). In this study, we utilized a purely model-based approach to estimate the temperature in the coils. It is also possible to use a hybrid approach (e.g., extended Kalman filter). In any case, it is prudent to incorporate some form of temperature monitoring to protect the coils from self-destruction, and it makes sense to also consider coil temperature in the current solver itself. A user of a system without temperature sensing should not be discouraged from using the methods proposed here.

In our proposed weight matrix in (3), we chose to only change the weights on coils once their temperatures crossed a critical temperature. In our initial explorations, we considered other weight matrices that penalized heating at all temperatures, but we found that they resulted in less desirable behavior. For a given actuation command, some coils are simply better conditioned than others. When they get used, they begin to heat. If they are immediately penalized, the cooler coils begin to be used, but because they are more poorly conditioned, the currents are relatively high and they heat up relatively quickly. Ultimately, we found that, although the temperature in the hottest coil could be kept slightly cooler, the overall heating in the system was significantly higher.

In the weight matrix used in our simulations, we used a somewhat arbitrary $\xi = 0.2$. When we consider that the maximum temperatures reached in all of the most difficult cases

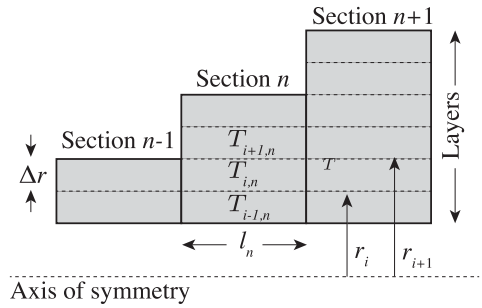

Fig. 6. Thermal-model parameters, based on [12].

were the same, rising no more than 1 °C above the critical temperature, it seems that the required factor of safety in choosing the critical temperature is quite insensitive to the specific actuation command. If we repeat the simulations of Fig. 4, but using $\xi = 0.4$ (i.e., doubled), the maximum temperature rise is only slightly reduced but the sustainable force is also slightly reduced. If we instead use $\xi = 0.1$ (i.e., halved), the maximum temperature rise is a full 12 °C above the critical temperature, with only a slight increase in the sustainable force. Thus, if ξ is reduced too low, it will harm the thermal protection offered by the weighted pseudoinverse, whereas as ξ is increased, we observe an asymptotic reduction in the temperature rise above the critical temperature.

APPENDIX A
A SIMPLE THERMAL MODEL FOR ELECTROMAGNET COILS

For the purpose of this study, we use a simple thermal model introduced by McIntosh and Ellis [12] to estimate the temperature throughout a coil, which we briefly summarize here. Their model uses finite differences, where each element is a ring with the heat exchange with the neighboring layers above and below (Fig. 6). Layer i has an inner radius of r_i and l_n is the length of each section. The time rate of change in temperature of an element in layer i and section n is governed by:

$$\frac{dT_i}{dt} = \frac{\gamma}{\beta} I^2 \left(1 + \alpha T_i\right) + \frac{1}{\beta r_i}[(T_{i+1} - T_i)(r_i + r_{i+1}) + (T_{i-1} - T_i)(r_i + r_{i-1})] \quad (8)$$

where I is the current in the wire, α is the temperature coefficient of the resistivity of the wire, assuming a linear model (which for copper is recommended as $\alpha = 3.9 \times 10^{-3}$ K^{-1}), and β and γ are coefficients that can be fit empirically. Note that there is no heat exchange between sections. A more complete parameterized model is provided in [12], but the simplified model in (8) is sufficient for our purposes.

In our study, we implemented a single section with 44 layers. We used $\beta = 65$ and $\gamma = 0.18$, which provided a dynamic response that is comparable to thermocouple readings that we have observed in real electromagnets. We returned the temperature of the hottest layer as the nominal temperature of the coil.

APPENDIX B
PSEUDOCODE FOR RWPINV

This appendix includes the complete pseudocode for the RWPINV method. The main function is provided as Algorithm 1, which calls the other algorithms. The method assumes the power supply's maximum current (units A) is stored in I_{p}, the magnitude of the amplifiers' current saturation limits are stored in the array I_{a}, the estimated temperatures of the coils are stored in the array T, the array of critical temperatures are stored in T_c, and the user-specified constant that governs how aggressively temperature is penalized after crossing T_c is ξ.

Note that Algorithm 2 implements the weight matrix proposed in (3), but this weight matrix could be modified as desired.

Algorithm 1: Redistributed Weighted Pseudoinverse (Main).

procedure RWPINV (Y_{des}, $\mathbb{A}$, I_{a}, I_{p}, T, T_c, ξ)
  $[m, n] \leftarrow$ size($\mathbb{A}$)
  $\rho \leftarrow$ rank($\mathbb{A}$)
  $C \leftarrow 1$ **to** n ▷list of available coils
  $I_{\mathrm{prev}} \leftarrow \mathrm{zeros}(n, 1)$
  $\mathbb{W} \leftarrow \mathrm{WEIGHT}(T, T_c, \xi, I_{\mathrm{a}})$
  return RECUR(Y_{des}, $\mathbb{A}$, I_{a}, $-I_{\mathrm{a}}$, I_{p}, $n - \rho$, I_{prev}, C, $\mathbb{W}$)

Algorithm 2: Calculation of Weight Matrix of Eq. (3).

procedure WEIGHT (T, T_c, ξ, I_{a})
  $n \leftarrow$ size(T)
  $\mathbb{W} \leftarrow$ identity(n)
  for i :=1 **to** n **do**
    if $T(i) > T_c$ **then**
      $\mathbb{W}(i, i) \leftarrow I_{\mathrm{a}}(i)^{-2} e^{\xi(T(i) - T_c)}$
    else
      $\mathbb{W}(i, i) \leftarrow I_{\mathrm{a}}(i)^{-2}$
  return $\mathbb{W}$

Algorithm 3: Recursive Function for RWPINV.

procedure RECUR (Y_{des}, $\mathbb{A}$, $I_{\mathrm{a}+}$, $I_{\mathrm{a}-}$, I_{p}, η, I_{prev}, AC, $\mathbb{W}$)
  $\mathbb{A}_C \leftarrow \mathbb{A}(:, C)$ ▷$\mathbb{A}$ of available coils
  $\mathbb{W}_C \leftarrow$ zeros(length(C), length(C))
  for i :=1 **to** length(C) **do**
    $\mathbb{W}_C(i, i) \leftarrow \mathbb{W}(C(i), C(i))$
  $I_{\mathrm{temp}} \leftarrow \mathbb{W}_C^{-1} \mathbb{A}_C^{\top} (\mathbb{A}_C \mathbb{W}_C^{-1} \mathbb{A}_C^{\top})^{\dagger} Y_{\mathrm{des}}$
  $I \leftarrow$ zeros(length(I_{prev}),1)
  for i :=1 **to** length(C) **do**
    $I(C(i)) \leftarrow I_{\mathrm{temp}}(i)$
  $[\delta_{\mathrm{a}}, \mathrm{arg}] \leftarrow$ SCALEAMP($I_{\mathrm{a}+} - I_{\mathrm{prev}}, I_{\mathrm{a}-} - I_{\mathrm{prev}}, I$)
  $I \leftarrow \delta_{\mathrm{a}} I$
  $\delta_{\mathrm{p}} \leftarrow$ SCALEPOWER($I_{\mathrm{p}}, I_{\mathrm{prev}}, I$)
  if $\delta_{\mathrm{p}} \leq 1$ **then**
    $I \leftarrow \delta_p I$ ▷power supply saturation
  else if $\delta_{\mathrm{a}} < 1$ **and** $\eta \neq 0$

```
    Y_des ← Y_des − 𝔸I
    I_prev ← I_prev + I
    C.remove(arg)                    ▷remove coil from list
    η ← η − 1                        ▷reduce dimension of null space
    I_r ← RECUR(Y_des, 𝔸, I_a+, I_a−, I_p, η, I_prev, C, 𝕎)
    I ← I + I_r
return I
```

Algorithm 4: Amplifier's Scale Function.

```
procedure SCALEAMP (I_maxHigh, I_maxLow, I
    δ_a ← 1)
    arg ← []
    for i :=1 to length(I) do
      if I(i) > 0  and  I_maxHigh(i)/I(i) < δ_a then
        δ_a ← I_maxHigh(i)/I(i)
        arg = i
      if I(i) < 0  and  I_maxLow(i)/I(i) < δ_a then
        δ_a ← I_maxLow(i)/I(i)
        arg = i
    return δ_a, arg
```

Algorithm 5: Power Supply's Scale Function.

```
procedure SCALEPOWER (I_p, I_prev, I)
    res ← 0.001        ▷resolution of current commands
    lower ← I_p/‖I_prev + I‖_1
    if lower ≥ 1 then
      return 1
    else
      upper ← (I_p − ‖I_prev‖_1)/‖I ‖_1
      while TRUE do                  ▷bisection method
        δ_p ← mean(lower, upper)
        error ← I_p − ‖I_prev + δ_p I‖_1
        if error < 0 then
          lower ← δ_p
        else
          upper ← δ_p
        if |lower − upper| < res
          return δ_p
```

REFERENCES

[1] J. J. Abbott, E. Diller, and A. J. Petruska, "Magnetic methods in robotics," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 3, pp. 57–90, 2020.

[2] M. P. Kummer, J. J. Abbott, B. E. Kratochvil, R. Borer, A. Sengul, and B. J. Nelson, "OctoMag: An electromagnetic system for 5-DOF wireless micromanipulation," *IEEE Trans. Robot.*, vol. 26, no. 6, pp. 1006–1017, Dec. 2010.

[3] A. J. Petruska and B. J. Nelson, "Minimum bounds on the number of electromagnets required for remote magnetic manipulation," *IEEE Trans. Robot.*, vol. 31, no. 3, pp. 714–722, Jun. 2015.

[4] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Trans. Man-Machine Syst.*, vol. MMS-10, pp. 47–53, Jun. 1969.

[5] C. A. Klein and C.-H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern.*, vol. SMC-13, no. 3, pp. 245–250, Mar.-Apr. 1983.

[6] J. M. Hollerbach and K. C. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE J. Robot. Autom.*, vol. RA-3, no. 4, pp. 308–316, Aug. 1987.

[7] B. Hu, C. L. Teo, and H. P. Lee, "Local optimization of weighted joint torques for redundant robotic manipulators," *IEEE Trans. Robot. Autom.*, vol. 11, no. 3, pp. 422–425, Jun. 1995.

[8] J. Park, Y. Choi, W. K. Chung, and Y. Youm, "Multiple tasks kinematics using weighted pseudo-inverse for kinematically redundant manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2001, pp. 4041–4047.

[9] J. Edelmann, A. J. Petruska, and B. J. Nelson, "Magnetic control of continuum devices," *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 68–85, 2017.

[10] W. S. Levine, *The Control Handbook: Control System Fundamentals*, 2nd ed. CRC Press, 2010.

[11] M. Miyasaka and P. Berkelman, "Magnetic levitation with unlimited omnidirectional rotation range," *Mechatronics*, vol. 24, no. 3, pp. 252–264, 2014.

[12] E. M. McIntosh and J. Ellis, "Note: A simple model for thermal management in solenoids," *Rev. Sci. Instrum.*, vol. 84, p. 116106, 2013.