The FastPivot Lift-and-Project Strategy to Avoid Local Minima

Ang Li[1*], **Yuling Guan**[1], **Peter Wu**[1], **Sven Koenig**[2], **Stephan Haas**[1], **T. K. Satish Kumar**[1]

[1]University of Southern California, USA
[2]University of California, Irvine, USA
ali355@usc.edu, yulinggu@usc.edu, pcwu@usc.edu, sven.koenig@uci.edu, shaas@usc.edu, tkskwork@gmail.com

Abstract

In computational physics, mathematical equations allow us to accurately map a given physical system to its response. Conversely, inverse problems involve the task of designing a physical system that produces a target response. Inverse problems are typically cast as search problems and solved using the Monte Carlo method, which frequently encounters local minima despite using various noise strategies to escape them. FastPivot is a recent algorithm that is known to outperform the Monte Carlo method on certain kinds of inverse problems. It starts with a system state and invokes alternating forward and backward passes through the system variables. In a forward pass, it leads the current state of the system to its response. In the subsequent backward pass, a small amount of information percolates from the target response back to the system variables. In this paper, we first analyze the FastPivot algorithm and posit that its unique ability to percolate information from the target response allows it to avoid local minima by "attaching an elastic tether" to a global minimum. We then propose a lift-and-project strategy to enhance the FastPivot algorithm. This strategy exploits the observation that, with increasing number of dimensions, the FastPivot algorithm encounters fewer local minima while the Monte Carlo method encounters more local minima. Overall, FastPivot with the lift-and-project strategy is a very promising approach: It rarely encounters local minima on the quintessential inverse problem of placing atoms in a bounded region using a scanning tunneling microscope to achieve target responses in the density of states.

Introduction

The laws of physics are usually stated using mathematical equations at different spatiotemporal scales. These mathematical equations allow us to accurately map a given physical system to its response. For example, the mathematical equations that describe gravity allow us to predict the trajectories of planets in distant galaxies. Despite the usefulness of mathematics in "forward" reasoning, i.e., from the system state to its response, we often face the inverse question while building systems: How should we design a physical system that produces a target response? Such inverse problems arise at both macroscopic and microscopic scales.

Generally speaking, inverse problems in computational physics present the following challenges: (a) The inverse map from the response to the system is only implicitly defined, i.e., we cannot input the target response into a mathematical equation; (b) the inverse map is not unique; (c) the search space is very large; and (d) there may be a substantial involvement of complex numbers. Inverse problems are particularly hard to solve when the map from the system to its response involves many cascading steps. Each step may be hard to reverse or may introduce branching possibilities if its inverse is not unique.

For the above-mentioned reasons, inverse problems are combinatorial in nature and cannot always be solved using standard tools from mathematics. Hence, intelligent search techniques could be more relevant. While some inverse algorithms have been designed for specific inverse problems using data-driven methods like neural networks (Lucas et al. 2018; Voronin, Nosov, and Savin 2019; Gao et al. 2022), such methods are mostly tailored to a narrow scope of problems and are not broadly applicable.[1] So far, very few general principles have been developed for the design of inverse algorithms.

The Monte Carlo method (Kroese, Taimre, and Botev 2011) is one such general principle used in computational physics. However, despite its popularity, it has some important drawbacks: It frequently encounters local minima notwithstanding the various noise strategies that it uses to escape them. Moreover, with increasing number of dimensions, it encounters more local minima in the landscape of minimization: There are more directions at each point in a higher-dimensional space, which makes it harder to guess the directions that lead to the global minima.

FastPivot (Guan et al. 2022) is a recent algorithm that is known to outperform the Monte Carlo method on certain kinds of inverse problems. It starts with a system state and invokes alternating forward and backward passes through the system variables. In a forward pass, it leads the current state of the system to its response. In the subsequent backward pass, a small amount of information percolates from the target response back to the system variables. The inner loop implements several alternating forward and backward passes in the hope of convergence. The outer loop keeps

[*]Corresponding Author

[1]In fact, it is even difficult to train a neural network to learn the inverse mapping of a step that computes the eigenvalues of a matrix satisfying a certain property P. This is so because it is hard to identify a unique matrix that satisfies P given only its eigenvalues.

track of the best solution found so far and triggers algorithm termination based on the availability of computational resources. While the Monte Carlo method works on a discrete space, i.e., on a lattice, the FastPivot algorithm typically works on a continuous space and produces high-quality solutions efficiently.

In this paper, we analyze and enhance the FastPivot algorithm. We first posit that the FastPivot algorithm's unique ability to percolate information from the target response allows it to avoid local minima by "attaching an elastic tether" to a global minimum. We then propose a lift-and-project strategy to enhance the FastPivot algorithm. This strategy exploits the observation that, with increasing number of dimensions, the FastPivot algorithm encounters fewer local minima while the Monte Carlo method encounters more local minima. We demonstrate the FastPivot algorithm's success on a quintessential inverse problem in computational physics: The problem of placing atoms in a bounded region using a scanning tunneling microscope (STM) to achieve target responses in the density of states (DoS). It is of paramount importance in nanoscience, where it is referred to as the DoS design problem. We show that the FastPivot algorithm outperforms the Monte Carlo method and is further enhanced by the lift-and-project strategy. Overall, FastPivot with the lift-and-project strategy is a very promising approach that rarely encounters local minima.

Background

In this section, we describe the quintessential DoS design problem used in computational physics to test various search methods (Levi and Haas 2009). We also describe the Monte Carlo method traditionally used for solving it. A more comprehensive description of the DoS design problem can be found in (Guan et al. 2022).

The STM is a type of microscope that can be used for imaging surfaces at the atomic level. It can also be used to manipulate individual atoms and place them at any desired location in a bounded region on a substrate. Figure 1 shows the schematic of an STM. In the context of using an STM for nanofabrication, the fundamental DoS design problem can be stated as follows: Given N atoms of a certain kind and a bounded region on a substrate, where should we place these atoms to achieve a target response in the DoS? In order to understand this question, we first need to understand the "forward" version of this problem: For a given placement (configuration) of N atoms, what is the DoS that it determines? This determination is done as described below.

First, we subscribe to the tight-binding model in solid-state physics (Rudenko and Katsnelson 2014; Nakhaee, Ketabi, and Peeters 2018). We consider a non-periodical system, which has the following Hamiltonian:

$$\hat{H} = -\sum_{i,j} t_{i,j} \left(\hat{c}_i^\dagger \hat{c}_j + \hat{c}_i \hat{c}_j^\dagger \right), \tag{1}$$

where $\hat{c}_i^\dagger$ and $\hat{c}_i$ represent the electron creation and annihilation operators, respectively, at site $\mathbf{r}_i$, and $t_{i,j}$ is the spatial decay long-range hopping term given by the power law:

$$t_{i,j} = \frac{t}{||\mathbf{r}_i - \mathbf{r}_j||^q}. \tag{2}$$

Here, t is a constant that is typically unity; $\mathbf{r}_i$ is the position vector of atom i, for $1 \leq i \leq N$; and $||\cdot||$ represents the L_2 norm of a vector. Moreover, q is the power decay parameter that reflects an algebraic variation of the overlap integral with inter-atomic separation. It is material-specific and measurable through experiments (Nazin, Qiu, and Ho 2003).

Given the position vectors $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$ of the N atoms, the steps required to determine the DoS are as follows:

1. Determine the Hamiltonian $\mathbf{H}$ as the $N \times N$ real matrix with $\mathbf{H}_{i,j} = -t_{i,j}$.
2. Determine the eigenvalues $e_1, e_2 \ldots e_N$ of $\mathbf{H}$.
3. Determine the DoS by placing uniform Lorentzian functions of appropriate height and width centered at each of the eigenvalues.

Essentially, the DoS describes the proportion of states occupied by the system at each energy level. It peaks at the eigenvalues of $\mathbf{H}$. Figure 1 shows the DoS for three different configurations of atoms. The inverse problem, of interest in this paper, is to find the position vectors $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$ in a bounded region given the target eigenvalues of $\mathbf{H}$, i.e., the positions of the peaks in the target DoS.

Let the target eigenvalues be $\lambda_1 \geq \lambda_2 \ldots \lambda_N$. All these eigenvalues are real since the Hamiltonian $\mathbf{H}$ is always a real symmetric square matrix. For position vectors $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$ of the N atoms in a K-dimensional space, let $\mathbf{R}$ denote the $K \times N$ position matrix with columns $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$. $\mathbf{R}$ represents a configuration of the N atoms in K-dimensional space, and K is usually 2 or 3. In general, let $\mathbf{H}$ be the Hamiltonian induced by a position matrix $\mathbf{R}$, and let $e_1 \geq e_2 \ldots e_N$ be its eigenvalues. $\mathbf{R}$ can be assigned a score to measure how close it is to achieving the target eigenvalues. One such score is the Root Mean Square Error (RMSE) given by:

$$\texttt{score}(\mathbf{R}) = \sqrt{\frac{\sum_{i=1}^{N} (e_i - \lambda_i)^2}{N}}. \tag{3}$$

The Monte Carlo Method

The Monte Carlo method works on a discretization of the continuous space and is inherently limited by the resulting discretized grid. It uses the Metropolis-Hastings importance sampling procedure (Kroese, Taimre, and Botev 2011) in its inner loop and generally works as follows. In the outer loop, the algorithm runs a user-specified number of trials. In the inner loop, i.e., in each trial, the algorithm starts by generating a position matrix $\mathbf{R}^0$ by choosing the position vector of each atom uniformly at random in the bounded region. The induced Hamiltonian $\mathbf{H}^0$ is also computed. Subsequently, with each increment in the iteration number i, the algorithm updates the current position matrix $\mathbf{R}^i$ and computes the induced Hamiltonian $\mathbf{H}^i$. $\mathbf{R}^{i+1}$ is constructed from $\mathbf{R}^i$ after considering to move a randomly chosen atom from its current position to a neighboring empty position on the discretized grid to obtain $\mathbf{R}'$. If

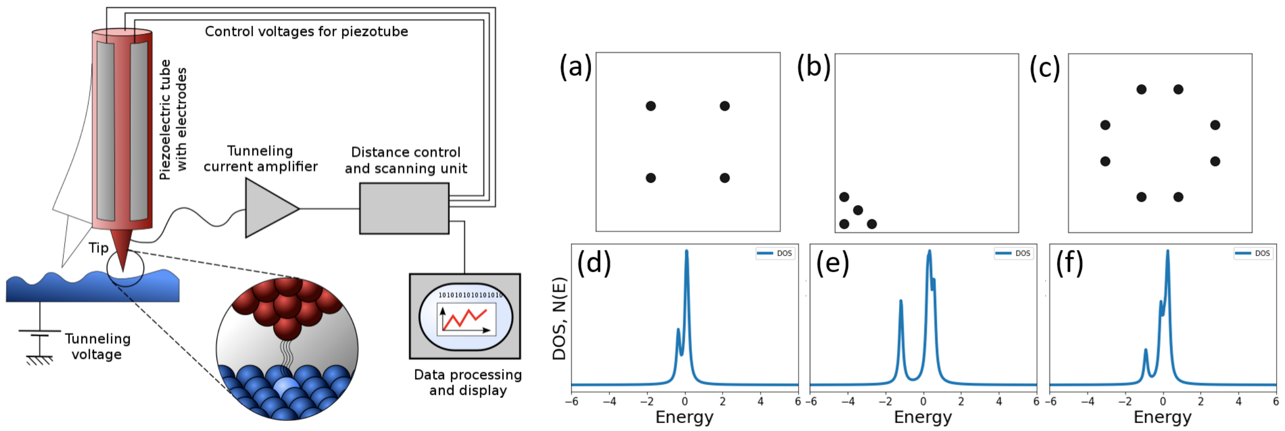

Control voltages for piezotube
Piezoelectric tube with electrodes
Tunneling current amplifier
Distance control and scanning unit
Tip
Tunneling voltage
Data processing and display
(a)
(b)
(c)
(d)
(e)
(f)
DOS
DOS, N(E)
Energy
−6 −4 −2 0 2 4 6

Figure 1: The left panel shows the schematic of an STM (Schmid and Pietrzak 2005). The right panel shows different configurations and their respective DoS: (a), (b), and (c) show three different configurations with 4, 4, and 8 atoms, respectively. (d), (e), and (f) show their respective DoS.

$\texttt{score}(\mathbf{R}') < \texttt{score}(\mathbf{R}^i)$, the move is accepted and $\mathbf{R}^{i+1}$ is set to $\mathbf{R}'$. Otherwise, the move is accepted with probability $e^{\frac{\texttt{score}(\mathbf{R}^i)-\texttt{score}(\mathbf{R}')}{k_B T}}$, where k_B is the Boltzmann constant and T is a “temperature” that starts high and decreases according to an annealing rule as the iteration number increases. The inner loop is repeated for a user-specified maximum number of iterations before the next trial of the outer loop is initiated. Upon termination, the algorithm returns the position matrix with the lowest recorded score.

The FastPivot Algorithm and Its Analysis

In this section, we analyze the FastPivot algorithm. A complete description of the FastPivot algorithm, including the pseudocode and an explanation of every line of it, can be found in (Guan et al. 2022).[2] However, our analysis of the FastPivot algorithm presented here goes beyond the mere description provided in (Guan et al. 2022): Our analysis is incisive and intuitively explains the superior performance of the FastPivot algorithm over the Monte Carlo method.

The FastPivot algorithm works on continuous spaces and does not have the inherent limitations of discretization. Its input mainly consists of: (a) the number of atoms N, (b) the target eigenvalues $\lambda_1 \geq \lambda_2 \ldots \lambda_N$, (c) the number of dimensions K, (d) the bounded region of space specified by the K-dimensional orthotope between the farthest corners $(0, 0 \ldots 0)$ and $(B_1, B_2 \ldots B_K)$, and (e) the power used in the Hamiltonian terms q. The output is a predicted position matrix $\mathbf{R}$ intended to achieve the target eigenvalues.

The fundamental principle of the FastPivot algorithm is different from that of the Monte Carlo method. Figure 2 illustrates this difference from two incisive perspectives. The left panel uses the perspective of forward and backward passes between a system state and its response. The right panel uses the perspective of foraging moves performed by the two approaches on the optimization landscape.

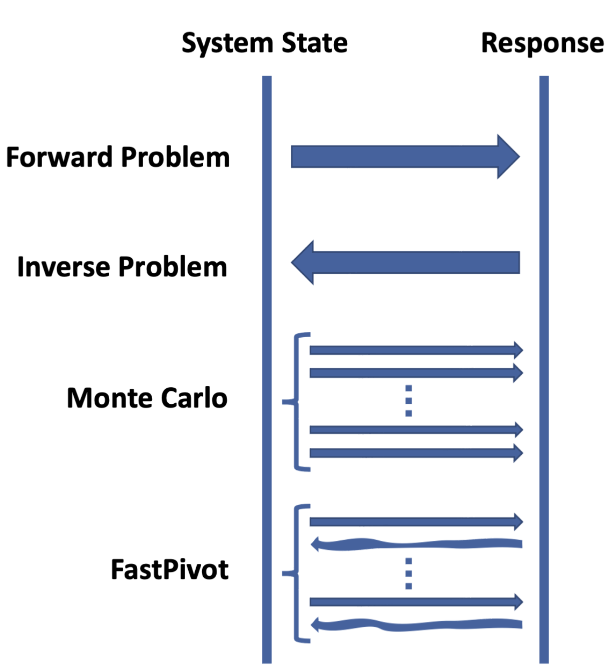
[2]The pseudocode is not repeated here. We refer the reader to it for our analysis.

The left panel of Figure 2 articulates a major difference between the FastPivot algorithm and the Monte Carlo method using forward and backward passes. A forward pass maps a system state to its response, and it often uses mathematical equations. In the DoS design problem, the forward pass first determines the Hamiltonian $\mathbf{H}$ from the given position vectors $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$, and then determines the eigenvalues of $\mathbf{H}$.[3] A backward pass is the theoretical inverse of the forward pass: It finds the system state that yields a target response. It often cannot be achieved using mathematical inverses. The DoS design problem is an inverse problem that corresponds to a backward pass; it is not amenable to mathematical inverses.

In lieu of a backward pass, the Monte Carlo method uses a series of forward passes that are guided by a scoring function. A scoring function evaluates how close the response of a state is to the target response. In effect, the Monte Carlo method starts at a system state, extracts a set of neighboring states, scores the current state and its neighboring states, and updates the current state to one of its neighboring states for the next iteration. An update often follows a neighboring state that has a better score. However, probabilistic updates and random restarts are also used as strategies to escape local minima. Fundamentally, the Monte Carlo method uses only forward passes to solve an inverse problem.

Unlike the Monte Carlo method, the FastPivot algorithm uses alternating forward and backward passes, where each backward pass is “approximate” and enabled by the previous forward pass. For example, in the DoS design problem, the target eigenvalues $\lambda_1 \geq \lambda_2 \ldots \lambda_N$ cannot be uniquely “reversed” to find an appropriate Hamiltonian $\mathbf{H}$. While this reversal is hard in a standalone backward pass, it becomes easy in the context of the most recently completed forward

[3]The DoS is computed by placing uniform Lorentzian functions of appropriate height and width centered at each of the eigenvalues. This final step can be reversed easily using mathematical inverses, that is, given the DoS, the appropriate eigenvalues around which Lorentzian functions should be placed can be easily computed.



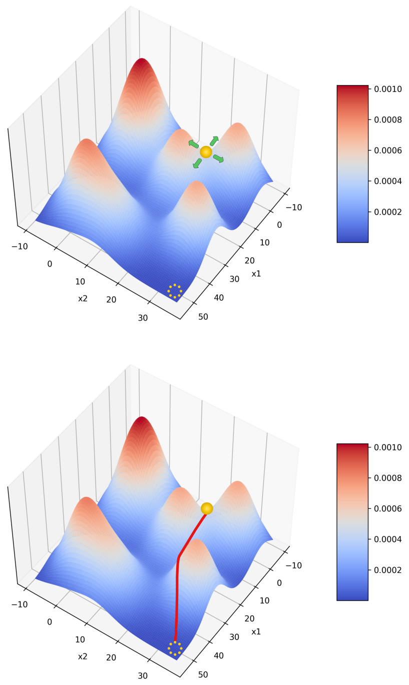


Figure 2: The left panel illustrates an overall analysis using forward and backward passes between a system state and its response. A forward pass maps a system state to its response using mathematical equations. An inverse problem corresponds to a backward pass that often cannot be achieved using mathematical inverses. Monte Carlo uses only forward passes in its forage of the optimization landscape. FastPivot uses alternating forward and backward passes, where each backward pass is approximate and enabled by the previous forward pass. The right panel shows a comparison of the foraging moves of Monte Carlo (top) and FastPivot (bottom) on the optimization landscape. The current system state is indicated using a yellow ball and the position of the global minimum is indicated using a dotted circle. Monte Carlo is oblivious to the global minimum, whereas FastPivot attaches an elastic tether to it by percolating information from it in the backward passes.

pass: Algorithm 3 of (Guan et al. 2022). In particular, the preceding forward pass computes an eigendecomposition of $\mathbf{H}$ and expresses it as $\mathbf{V} \times \mathbf{E} \times \mathbf{V}^{-1}$, where $\mathbf{E}$ is a diagonal matrix of non-increasing eigenvalues, and the columns of $\mathbf{V}$ represent the corresponding eigenvectors: Line 1 of Algorithm 2 of (Guan et al. 2022). Using this eigendecomposition, the desired reversal can be achieved, and a new Hamiltonian $\mathbf{D}$ can be reconstituted by merely replacing the diagonal matrix of eigenvalues $\mathbf{E}$ with the diagonal matrix of the target eigenvalues $\lambda_1 \geq \lambda_2 \ldots \lambda_N$: Line 2 of Algorithm 2 of (Guan et al. 2022).

The backward pass also needs to execute a reversal step from the new Hamiltonian $\mathbf{D}$ to a new position matrix $\mathbf{R}$. This reversal is also hard in a standalone backward pass. However, in the context of the most recently completed forward pass, it becomes easier since the current values of the position vectors $\mathbf{r}_1, \mathbf{r}_2 \ldots \mathbf{r}_N$ are available. The approximate backward pass makes use of the observation that it suffices to change the position vectors by small amounts in the direction of $\mathbf{D}$, since doing so across many forward and backward passes achieves the desired macroscopic effect. Therefore, the approximate backward pass uses τ steps of gradient descent with respect to the position vectors aimed at minimizing the Frobenius norm of $\mathbf{D} - \mathbf{H}$: Lines 3–7 of Algorithm 2 of (Guan et al. 2022).

The right panel of Figure 2 shows a comparison of the foraging moves of the Monte Carlo method (top) and the FastPivot algorithm (bottom) on the optimization landscape. The Monte Carlo method employs only forward passes. Therefore, it is oblivious to the positions of the global minima and frequently encounters local minima. Moreover, with increasing number of dimensions, it encounters more local minima in the optimization landscape: There are more directions at each point in a higher-dimensional space, which makes it harder to guess the directions that lead to the global minima.

Unlike the Monte Carlo method, the FastPivot algorithm uses both forward and backward passes. Every backward pass, albeit approximate, percolates information from the target response back to the system variables to guide small changes in them for the next forward pass. This allows the FastPivot algorithm to virtually attach an elastic tether to a "nearby" global minimum. As a result, the elastic tether provides stronger guidance than a scoring function alone. Of course, there could still be "snags" in the tether caused by certain regions of the optimization landscape that prohibit the improvement of the score across consecutive iterations: Line 12 of Algorithm 1 of (Guan et al. 2022). However, there are far fewer snags along a tether compared to the number of local minima in the entire optimization landscape. Moreover, with increasing number of dimensions, the FastPivot algorithm encounters fewer snags: There is more flexibility and more ways to circumvent a snag in a higher-dimensional space.

The Lift-and-Project Strategy

In this section, we provide an enhancement of the FastPivot algorithm, called the lift-and-project strategy. It is based on exploiting the observation that, with increasing number of dimensions, the FastPivot algorithm encounters fewer local minima[4] while the Monte Carlo method encounters more local minima.

Given the DoS design problem in K dimensions, we first solve it in L dimensions, for $L > K$, utilizing the flexibility of a higher-dimensional space. For the initial state in L dimensions, we set the first K coordinates of each atom's position vector as described in (Guan et al. 2022). The remaining $L - K$ coordinates are set to 0. Doing so imparts the same distribution of distances between the atoms in L dimensions as in K dimensions while providing more degrees of freedom. The L-dimensional solution is projected onto $L - 1$ dimensions using an efficient dimensionality reduction procedure, such as FastMap (Faloutsos and Lin 1995). The projected solution, in turn, is used as a starting point for solving the same problem in $L - 1$ dimensions for the next iteration. The resulting solution is projected onto $L - 2$ dimensions, and so forth, until the required K-dimensional solution is obtained. A direct reduction from L to K dimensions is also conceivable, but the proposed step-by-step reduction utilizes more guidance since it repeatedly achieves the target response at every intermediate stage.

Experimental Results

In this section, we present experimental results comparing four competing algorithms. These include the Monte Carlo (MC) method, the FastPivot (FP) algorithm, and the FastPivot algorithm with the lift-and-project strategy (FPLP). FPLP uses $L = 4$. We also include another method that performs naive hill climbing (HC). HC is similar to MC but does not accept upward moves. Moreover, in each iteration, it tries moving each of the N atoms to a neighboring vacant position to check for a score improvement. All algorithms were implemented in Python 3 and the experiments were conducted on a laptop with an Apple M2 Max chip and 96 GB RAM.

We use the tight-binding model and place atoms in a 2 or 3-dimensional bounded region of space to achieve the target DoS eigenvalues. The space is discretized by unit amounts on each dimension for HC and MC while it is continuous for FP and FPLP. All test instances are "reverse engineered". That is, they are generated as follows. We first pick a random configuration of the N atoms in the discretized bounded region of space with $B_1, B_2 \dots B_K = 10$. We compute the Hamiltonian with $q = 1$ and then its eigenvalues where the DoS peaks. We provide these eigenvalues as input to all the algorithms and challenge them to reconstruct the original configuration. Of course, the algorithms are also free to construct any other configuration with the same eigenvalues. The reverse engineering merely validates each test instance and assures us that the DoS is achievable. We generated 50 different instances for each setting of $K \in \{2, 3\}$, $N \in \{8, 16, 32\}$, and $\epsilon \in \{0.02, 0.01, 0.005\}$, where ϵ is the target RMSE value.

Our experiments are primarily directed towards understanding how intelligently the various algorithms avoid being stuck in local minima. Each algorithm is given a single trial on each of the instances. This compares their inner loops and factors out outer-loop strategies such as random restarts that are equally applicable in all algorithms. Therefore, HC terminates when it reaches a local minimum. MC is given $N\times$ the number of iterations of HC since HC tries to move each of the N atoms in every iteration. FP terminates when the score does not improve from the previous iteration to the current iteration. FPLP invokes FP with the same termination criterion in each higher-dimensional space.

The table in Figure 3 (left panel) shows the comparative performances of the four algorithms for different values of K, N, and ϵ. Each entry in the table shows the number of instances (out of 50) that the corresponding algorithm solved successfully within the RMSE threshold of ϵ.

First, we compare the performances of FP and FPLP against those of HC and MC. We observe that FP and FPLP significantly outperform HC and MC for both $K = 2$ and $K = 3$. In fact, the difference between their performances is particularly pronounced for high-precision solutions with $\epsilon = 0.005$: FP and FPLP are generally viable, but HC and MC are completely ineffective. Moreover, FP and FPLP get a significant boost for finding high-precision solutions when $K = 3$, even with increasing N. These observations show that FP and FPLP are not only significantly better than HC and MC at avoiding local minima but are also able to effectively exploit the flexibility of higher dimensions.

Second, we compare the performances of FP and FPLP against each other. On the one hand, for $K = 3$, we observe that both algorithms effectively exploit the flexibility of a 3-dimensional space and produce high-quality results. On the other hand, for $K = 2$, the DoS design problem is more constrained. Hence, instances with $K = 2$ are better suited to reveal the performance differences between FP and FPLP. When $N = 8$, the instances are easy to solve: not because of the value of K but because of the value of N. Thus, in this setting, the performances of the two algorithms are similar and of high quality. However, when $N = 32$, the instances are hard to solve: both because of the value of K and because of the value of N. Thus, in this setting, the performances of the two algorithms are similar but of low quality. When $N = 16$, the instances are critically posed to reveal the differences between the performances of the two algorithms. In this setting, we observe that FPLP outperforms FP, particularly for finding high-precision solutions with $\epsilon = 0.005$. These observations show that the lift-and-project strategy of FPLP is indeed an effective algorithmic ingredient.

Figure 3 (right panel) shows an example instance comparing the outputs of MC and FP. The outputs of HC and

[4]snags

N	Method	$K = 2$ and $\epsilon =$			$K = 3$ and $\epsilon =$		
		0.02	0.01	0.005	0.02	0.01	0.005
8	HC	14	2	0	46	25	5
	MC	23	3	0	45	32	4
	FP	47	43	40	50	50	49
	FPLP	46	46	44	50	50	50
16	HC	0	0	0	45	7	0
	MC	1	0	0	45	6	0
	FP	40	34	28	50	49	48
	FPLP	40	38	38	49	48	48
32	HC	0	0	0	40	2	0
	MC	0	0	0	33	0	0
	FP	10	6	8	49	49	49
	FPLP	10	9	6	50	50	49

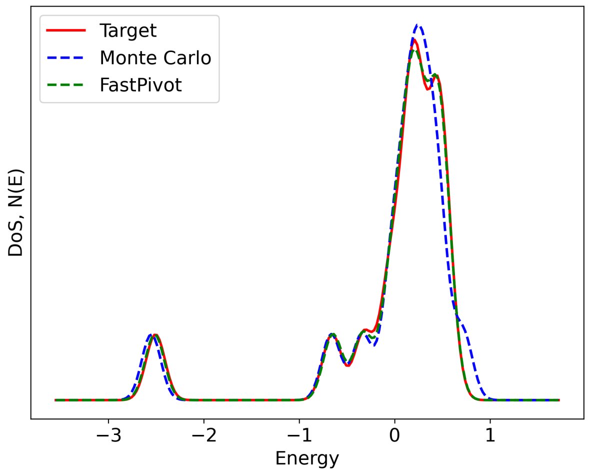


Figure 3: The left panel shows a comparison of the success rates of HC, MC, FP, and FPLP on 50 instances of the DoS design problem for each setting of $K \in \{2, 3\}$, $N \in \{8, 16, 32\}$, and $\epsilon \in \{0.02, 0.01, 0.005\}$. The right panel shows an example instance comparing the outputs of MC and FP. Here, $K = 3$, $N = 16$, and $\epsilon = 0.01$.

FPLP are excluded to avoid clutter. The output DoS curve of FP traces the target DoS curve very closely, including the bimodal high-peaks. The output DoS curve of MC deviates from the target DoS curve at various places. It also misses the bimodal high-peaks, representing it with just one high-peak.

Conclusions and Future Work

FastPivot is a recently developed algorithm for solving inverse problems in computational physics. In this paper, we first provided an incisive analysis of its behavior in comparison to the popularly used Monte Carlo method. While the Monte Carlo method uses only forward passes, the FastPivot algorithm uses alternating forward and backward passes through the system variables. In a forward pass, it computes the system's response from its current state. In the subsequent backward pass, it percolates a small amount of information from the target response back to the system variables. This unique ability to percolate information from the target response allows it to avoid local minima by "attaching an elastic tether" to a global minimum. We then proposed a lift-and-project strategy to enhance the FastPivot algorithm, exploiting its increased effectiveness in higher-dimensional spaces. Through various experiments, we demonstrated the superior performance of the FastPivot algorithm and the lift-and-project strategy. Overall, the FastPivot framework rarely encounters local minima on the DoS design problem. In future work, we will apply this framework to solve other problems of significance in computational physics: including the plasmon design problem.

Acknowledgments

This research work was supported by NSF under grant number 2112533 and by the Department of Energy (DOE-ASCR) Award DE-SC0026337.

References

Faloutsos, C.; and Lin, K.-I. 1995. FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

Gao, Y.; Liu, H.; Wang, X.; and Zhang, K. 2022. On an Artificial Neural Network for Inverse Scattering Problems. *Journal of Computational Physics*, 448.

Guan, Y.; Li, A.; Koenig, S.; Haas, S.; and Kumar, T. K. S. 2022. FastPivot: An Algorithm for Inverse Problems. In *Proceedings of the IEEE International Conference on Automation Science and Engineering*.

Kroese, D. P.; Taimre, T.; and Botev, Z. I. 2011. *Handbook of Monte Carlo Methods*. New York: Wiley Series in Probability and Statistics, John Wiley and Sons.

Levi, A.; and Haas, S. 2009. *Optimal Device Design*. Cambridge University Press.

Lucas, A.; Iliadis, M.; Molina, R.; and Katsaggelos, A. K. 2018. Using Deep Neural Networks for Inverse Problems in Imaging: Beyond Analytical Methods. *IEEE Signal Processing Magazine*, 35(1).

Nakhaee, M.; Ketabi, S. A.; and Peeters, F. M. 2018. Tight-Binding Model for Borophene and Borophane. *Phys. Rev. B*, 97.

Nazin, G. V.; Qiu, X. H.; and Ho, W. 2003. Visualization and Spectroscopy of a Metal-Molecule-Metal Bridge. *Science*, 302(5642).

Rudenko, A. N.; and Katsnelson, M. I. 2014. Quasiparticle Band Structure and Tight-Binding Model for Single- and Bilayer Black Phosphorus. *Phys. Rev. B*, 89.

Schmid, M.; and Pietrzak, G. 2005. Rastertunnelmikroskop-schema.svg. https://commons.wikimedia.org/wiki/File:Rastertunnelmikroskop-schema.svg. Accessed: 2025-10-03.

Voronin, E. A.; Nosov, V. N.; and Savin, A. S. 2019. Neural Network Approach to Solving the Inverse Problem of Surface-Waves Generation. *Journal of Physics: Conference Series*.