RESEARCH ARTICLE

Reverse annealing for nonnegative/binary matrix factorization

John Golden[1]*, **Daniel O'Malley**[1,2]

1 Computational Earth Sciences Group, Los Alamos National Laboratory, Los Alamos, NM, United States of America, **2** Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, MD, United States of America

* golden@lanl.gov

OPEN ACCESS

Citation: Golden J, O'Malley D (2021) Reverse annealing for nonnegative/binary matrix factorization. PLoS ONE 16(1): e0244026. https://doi.org/10.1371/journal.pone.0244026

Editor: Nicholas Chancellor, Durham University, UNITED KINGDOM

Received: August 24, 2020

Accepted: December 1, 2020

Published: January 6, 2021

Data Availability Statement: The data underlying the results presented in the study are available from MIT: http://www.ai.mit.edu/courses/6.899/lectures/faces.tar.gz.

Funding: This work was funded by Los Alamos National Laboratory's Laboratory Directed Research and Development Program (LDRD) and the National Nuclear Security Administration's Advanced Simulation and Computing (ASC) Program.

Competing interests: The authors have declared that no competing interests exist.

Abstract

It was recently shown that quantum annealing can be used as an effective, fast subroutine in certain types of matrix factorization algorithms. The quantum annealing algorithm performed best for quick, approximate answers, but performance rapidly plateaued. In this paper, we utilize reverse annealing instead of forward annealing in the quantum annealing subroutine for nonnegative/binary matrix factorization problems. After an initial global search with forward annealing, reverse annealing performs a series of local searches that refine existing solutions. The combination of forward and reverse annealing significantly improves performance compared to forward annealing alone for all but the shortest run times.

Introduction

Due to the slowing progress of classical computing [1], new computational architectures [2–4] have gained much interest in recent years. One such architecture is quantum annealing [5]. Recently, D-Wave's quantum annealing hardware [6, 7] has introduced a new form of annealing—reverse annealing [8, 9]. Here, we explore the use of reverse annealing in the context of Nonnegative/Binary Matrix Factoriztion (NBMF), which has shown some promise in combination with quantum annealing [10].

The NBMF algorithm factors a matrix A into the product of a nonnegative, real-valued matrix B and a binary matrix C. The NBMF algorithm is a variant of the Nonnegative Matrix Factorization (NMF) algorithm (which allows C to be real-valued rather than just binary). NMF, and by extension NBMF, are useful in machine learning contexts that seek to decompose a large data set into a set of features along with a mixing matrix, e.g. learning facial features [11], text mining [12], and hyperspectral imaging [13]. Our implementation of the NBMF algorithm employs an alternating least-squares approach, where each iteration includes the solution of a binary least squares problem and a nonnegative least squares problem. The binary least squares problem is solved with the quantum annealer. It has been shown that a quantum annealer provided noticeable speed-up compared to two classical solvers for the binary least squares problem [10].

One downside of the quantum annealer approach is that improvement in solution quality from iteration to iteration quickly plateaus. This is because the forward annealing approach that was used previously could only perform global searches when solving the binary least squares problem. This ignores the results of the solutions from previous iterations, which is likely a good starting point for the next iteration. Instead, the annealing process almost always produces a factor matrix that is very different from the factor matrix at the previous iteration. In practice, this means that the algorithm hops around solution space at random, quickly finding good solutions but never refining them beyond a certain level of accuracy.

Fortunately, the latest iteration of the D-Wave hardware, the 2000Q, allows us to explore solutions around some initial classical state. This process is known as reverse annealing. In this paper, we utilize reverse annealing to improve performance of the NBMF algorithm. Specifically, we use reverse annealing to explore local minima near an initial state defined by the results of the previous iteration of the algorithm. This significantly reduces the iteration-over-iteration change in the algorithm, allowing promising solutions to be refined rather than discarded.

Review of NBMF algorithm

The NBMF algorithm takes a real-valued $n \times m$ matrix A and finds B and C such that

$$A \approx BC \tag{1}$$

where B is a nonnegative $n \times k$ matrix and C is a binary $k \times m$ matrix. Generally, a small value of k is used, so that the factorization is low rank. The chief benefit of NBMF, as opposed to the more general nonnegative matrix factorization (NMF) algorithm (which only requires C be nonnegative, rather than binary), is that C tends to be sparse [10]. Additionally, the memory needed to store C is low due to its binary nature.

We will now give an outline of the NBMF algorithm and the implementation on the D-Wave; for full details of the algorithm see [10]. After randomly initiating a seed matrix $C^{(0)}$, each iteration follows an alternating least squares approach:

$$\text{find } B^{(i)} = \arg\min \|A - XC^{(i-1)}\|, \tag{2}$$

$$\text{find } C^{(i)} = \arg\min \|A - B^{(i)}X\| \tag{3}$$

An important feature of the NBMF algorithm is that Eq 3 can be efficiently implemented on a quantum annealer. We used the D-Wave 2000Q quantum annealer at Los Alamos National Laboratory, which is designed to solve quadratic unconstrained binary optimization (QUBO) problems, generically formulated as

$$f(\vec{q}) = \sum a_i q_i + \sum b_{ij} q_i q_j. \tag{4}$$

The D-Wave quantum annealer can be thought of as a sampler returning samples $\vec{q}$ drawn from a Boltzmann distribution where $f(\vec{q})$ is the energy. As we will discuss in the next section, the annealing process by which the D-Wave produces these samples is amenable to significant tuning by the user, with the goal of reducing the average energy of the samples.

Returning to our matrix factorization problem, the columns C_j in Eq 3 can be solved for independently,

$$C_j = \arg\min \|A_j - B \cdot \vec{q}\|. \tag{5}$$

This can be converted into the QUBO format, Eq 4, with the variable assignments

$$a_j = \sum_l B_{lj}(B_{lj} - 2A_{lj}), \tag{6}$$

$$b_{jk} = 2\sum_l B_{lj}B_{lk}. \tag{7}$$

In order to translate this QUBO onto the D-Wave hardware, we employ an embedding that chains multiple physical qubits into logical qubits. This is required as the D-Wave hardware graph features limited connectivity, while the QUBO under study here requires a complete graph with k vertices. The limited number of qubits and amount of connectivity on the physical hardware is in fact not a significant problem, as the goal of NBMF is to factor matrices in to a C with small rank.

For consistency, we use the same data set (2,429 facial images) and rank $k = 35$ as previous work [10, 11]. The embedding, found via D-Wave's heuristic embedder, utilizes 437 physical qubits, with at most 15 physical qubits combined in to a single logical qubit. The facial images are composed of $19 \times 19 = 361$ greyscale pixels. Our matrix A is therefore of size 2429×361, with each column composed of the greyscale values of the pixels for an individual face.

Reverse annealing: Method, calibration & timing

The original implementation of NBMF on the D-Wave used the standard forward anneal procedure, where the device starts in an equal superposition of all possible states. Our motivation for this study is to use a new feature of the D-Wave 2000Q, reverse annealing, in order to improve performance. Reverse annealing begins in a specified classical state, then explores solutions in the local vicinity of that initial state. This allows us to iteratively improve upon solutions from previous iterations of the algorithm, rather than conducting global searches at every step.

The NBMF algorithm begins with a random initialization of the B and C matrices. Reverse annealing from this random starting point is ineffective and requires many iterations to achieve results comparable to forward annealing. Conducting a single iteration with forward annealing and then switching to reverse annealing produces much better results (increasing the number of forward annealing iterations beyond this does not offer noticeable improvements). This is in line with the idea that forward annealing performs a global search and reverse annealing performs a local search. Intuitively, it is advantageous to start with a global search and then transition to a local search. In the rest of this section we calibrate the reverse anneal process after the initial round of forward anneals.

In the simplest terms, the reverse anneal process depends on two parameters:

- reversal distance $r \in (0, 1]$,
- reversal time t_r (in μs).

The (dimensionless) reversal distance r controls the scope of the local search. Setting $r = 1$ is equivalent to conducting a global search (i.e. losing all information about the initial classical state), while setting $r = 0$ will not conduct any search and instead return the initial state. The reversal time t_r controls how long the search is conducted. A longer search has a greater chance of returning a lower energy sample, but at the cost of slowing the algorithm down. For this analysis, we do not employ any other tuning of the D-Wave, e.g. spin reversal transforms.

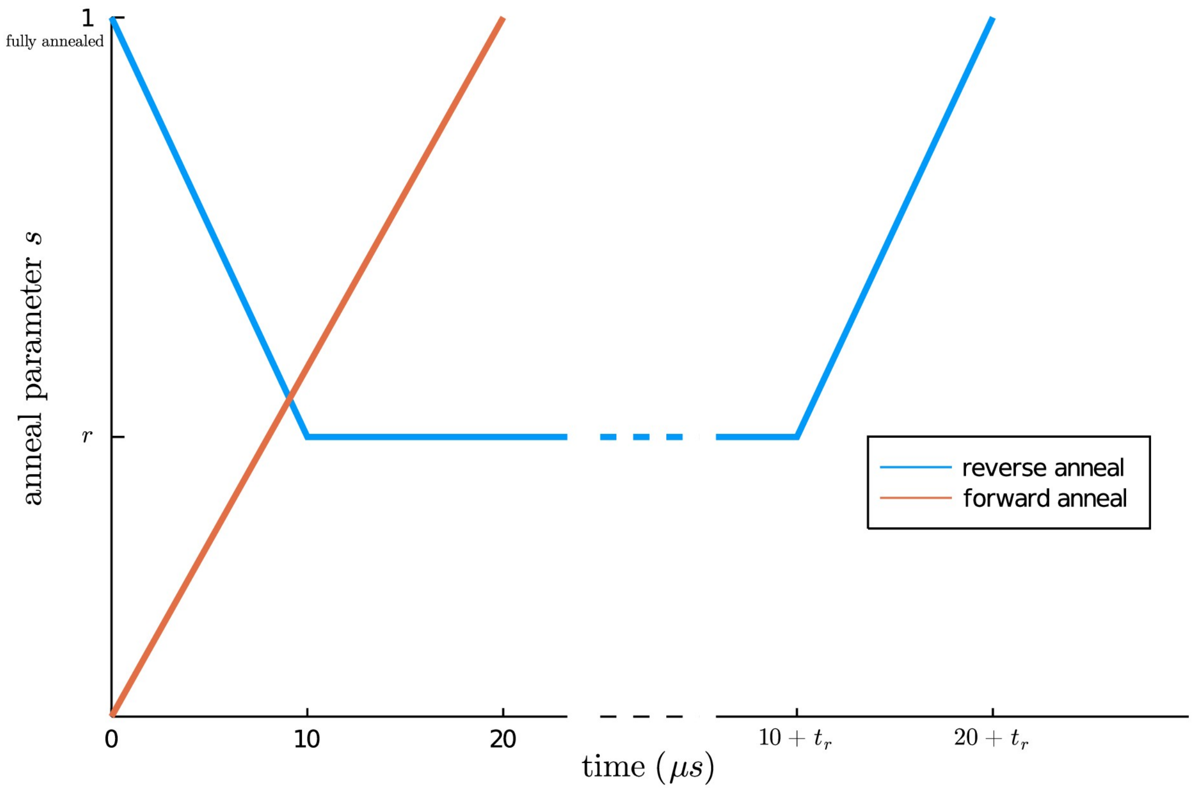

Fig 1. Represention of the anneal schedules used as a function of the dimensionless anneal parameter *s*. $s = 1$ indicates a fully annealed system. The reverse anneal schedule (blue) is parameterized by the reversal distance r and reversal time t_r, while default forward anneal schedule (orange) increases from $s = 0$ to $s = 1$ over 20 μs.

https://doi.org/10.1371/journal.pone.0244026.g001

For a given reversal distance r and time t_r, the anneal schedule we use is

$$[(0,1),\ (10,1-r),\ (10+t_r,1-r),\ (20+t_r,1)], \tag{8}$$

where the first entry in each tuple is the elapsed time (in microseconds) and the second entry is the dimensionless anneal parameter $s \in [0, 1]$, which controls the strength of a transverse magnetic field in the annealing device. As $s \rightarrow 1$, the transverse field strength decreases, encouraging quantum tunneling towards the ground state of the QUBO. The physical interpretation of Eq 8 is that we begin in a specific annealed state at $t = 0$, "warm" the system up to a certain temperature (parametrized by r), hold the system at that temperature for a time t_r, and then re-anneal the system. See Fig 1 for a diagrammatic representation of the reverse anneal schedule, as parameterized by r and t_r, and the D-Wave default forward anneal schedule.

In addition to specifying the reverse anneal schedule, we must also specify the initial state. As discussed in the introduction, the NBMF algorithm naturally provides an initial configuration based on the results of the previous iteration of the algorithm. Specifically, if we are solving for $C_j^{(i)}$, i.e. the ith iteration of the jth column of C, we can use $C_j^{(i-1)}$ as the beginning point of our reverse anneal process.

We characterize the efficacy of a reverse anneal sample by seeing if it:

- is the same as the initial state,
- has a lower energy than the initial state (good),

- has a higher energy than the initial state (bad).

The frequency with which samples fall in to each category gives us an idea of how effective the reverse anneal is at finding improved solutions.

We studied the effects of reversal time t_r and reversal distance r on 100 randomly selected QUBOs generated during an evaluation of the NBMF algorithm (after the first round of forward anneals). The overall effectiveness of reverse annealing on an individual QUBO was highly instance-specific; Fig 2 shows the average of our results (the standard deviation for each point was generally near 100% of the mean). However, the impact of t_r and r remained consistent across instances. First, in increasing t_r does not significantly increase the likelihood of discovering better states. The peak probability of discovering a lower-energy sample was 13.5% ± 9.1% for $t_r = 10\mu s$ and 13.52% ± 9.8% for $t_r = 100\mu s$. Second, the peak reversal distance remained constant across samples ($r = 0.45$ for $t_r = 10\mu s$ and $r = 0.4$ for $t_r = 100\mu s$). In order to minimize QPU access time, we therefore adopt a reverse anneal schedule for the NBMF algorithm with $r = 0.45$ and $t_r = 10\mu s$.

We note that the choice of these parameters has some dependence on the matrix that is being factored. For example, the same calibration procedure evaluated on a matrix with random values (as opposed to the highly structured facial imagery data) revealed an optimal reversal distance of $r = 0.2$.

There is additional computational overhead related to the reverse anneal process, such as configuring the hardware in to the chosen initial state before each anneal. Therefore, for the purposes of comparing forward and reverse anneal efficacy we will look at quality of solution vs. total QPU access time (as opposed to (annealing time × number of anneals), as was done in [10]). The total QPU access time is calculated via

$$\begin{aligned} \text{total QPU access time} \; &= \; (\text{anneal} \; + \; \text{readout} \; + \; \text{delay}) \times \text{number of samples} \\ &\quad + \text{QPU programming time} \end{aligned} \tag{9}$$

Forward and reverse anneals share identical readout and QPU programming times (123μs and 8001μs, respectively). As previously discussed, the forward anneal takes 20μs while the reverse anneal takes 30μs. The major difference is in the 'delay' time, as this is the period when the

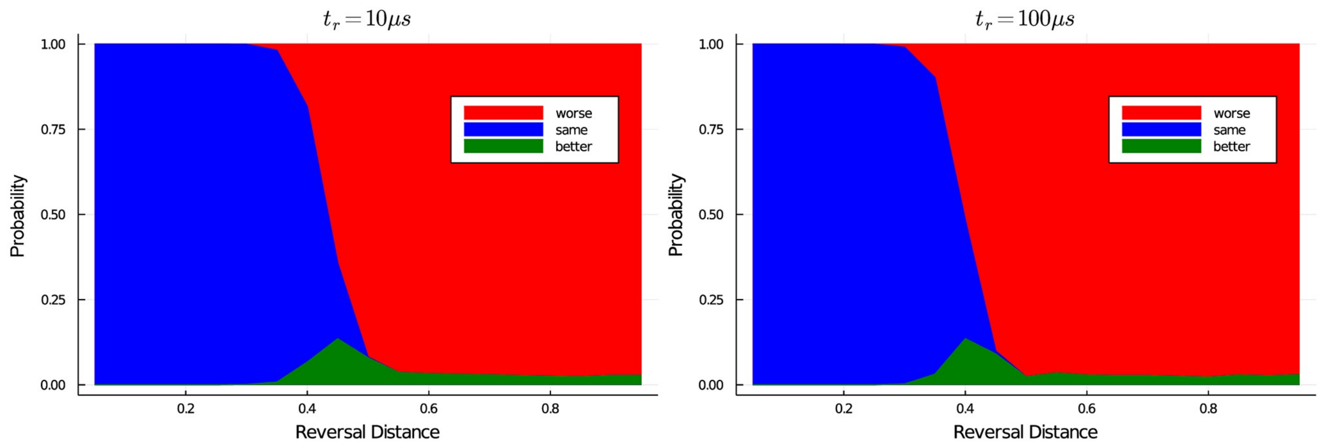

Fig 2. Calibration of the reverse anneal process. Evaluated for 100 randomly chosen QUBOs appearing during an evaluation of the NBMF algorithm for with $t_r = 10\mu s$ and $t_r = 100\mu s$. For a given reversal distance, the height of the green area indicates the mean probability that a reverse anneal sample will have a lower energy than the initial state. The heights of the red and blue areas indicate the mean probability that a given sample will be worse or the same, respectively, as the initial state. Standard deviations, not shown here, were upwards of 100% of the mean, however $t_r = 10\mu s$ and $t_r = 100\mu s$ always behaved similarly, and the peak reversal distance varied little from sample to sample.

https://doi.org/10.1371/journal.pone.0244026.g002 quantum annealer is reset to the initial state between anneals. For the D-Wave 2000Q used for this study, located at Los Alamos National Laboratory, the delay time per sample in the forward anneal case is 21μs, while the delay time per reverse anneal sample is 520μs. So we see that the biggest time commitment in doing reverse anneals comes not from the longer anneal schedule but from the repeated state preparation.

When comparing the reverse anneal results against the original forward anneal version of the algorithm, we allot each method equal QPU access time. Given the timing values discussed above, we find that the ratio

$$\text{number of reverse anneals} = 0.24 * (\text{number of forward anneals}) \tag{10}$$

results in equivalent total QPU access time. The remaining important variables are the number of anneals per QUBO and the total number of iterations for the algorithm to run. We discuss these in the following section.

Results

In this section we use reverse annealing in the NBMF algorithm to factor the dataset of 2,429 facial images studied in [10] in to a 2429 × 35 non-negative matrix B and 35 × 2429 binary matrix C. In this application, the columns of the C matrix can be interpreted as decompositions of each face in to 35 component features. First, we will examine the differences between the two algorithms for a fixed number of anneals. We will then study the efficacy of the two algorithms as a function of total QPU access time.

Fig 3 shows the results of the two algorithms with 6182 seconds of total QPU access time (equivalent to 1000 forward anneals or 240 reverse anneals per QUBO). The reverse anneal algorithm shows consistent improvement for many more iterations, and produces a better result than forward anneal by the third iteration.

Recall that our hypothesis, outlined in the introduction, is that reverse annealing will outperform forward annealing due to more refinement of existing solutions as opposed to generation of entirely new solutions per QUBO. If we define

$$\%\ \text{Change in}\ B = \frac{\|B^{(i+1)} - B^{(i)}\|}{\|B^{(i)}\|} \tag{11}$$

and % change in C as the Hamming distance between $C^{(i+1)}$ and $C^{(i)}$ divided by the size of C, then we can look at the iteration-over-iteration change in the B and C matrices to see if this is indeed the case, see Fig 4. The iterative improvement in C is particularly striking and shows that the forward anneal is constantly changing the C matrix while reverse anneal pushes towards a local minimum.

In Fig 5 we compare the efficacy of the two algorithms over multiple values of QPU access time. Here we see that for very small values of total QPU access time, forward annealing results in superior performance. However, once the total QPU access time exceeds ≈210s, which corresponds to 7 reverse anneals per QUBO, reverse annealing overtakes forward annealing, eventually plateauing at approximately 12% improvement over the forward annealing algorithm.

The reason that forward annealing outperforms reverse annealing for small sample size is straightforward. For a single reverse anneal sample, the likelihood of finding a better state than the initial configuration is quite low ($\leq$ 25%, sometimes much lower, see Fig 2). Therefore, when the number of reverse anneals per QUBO is small, most iterations will result in no change. Forward annealing might be finding worse or different solutions for each QUBO, but the fact that they are new solutions for each iteration means that overall the algorithm can

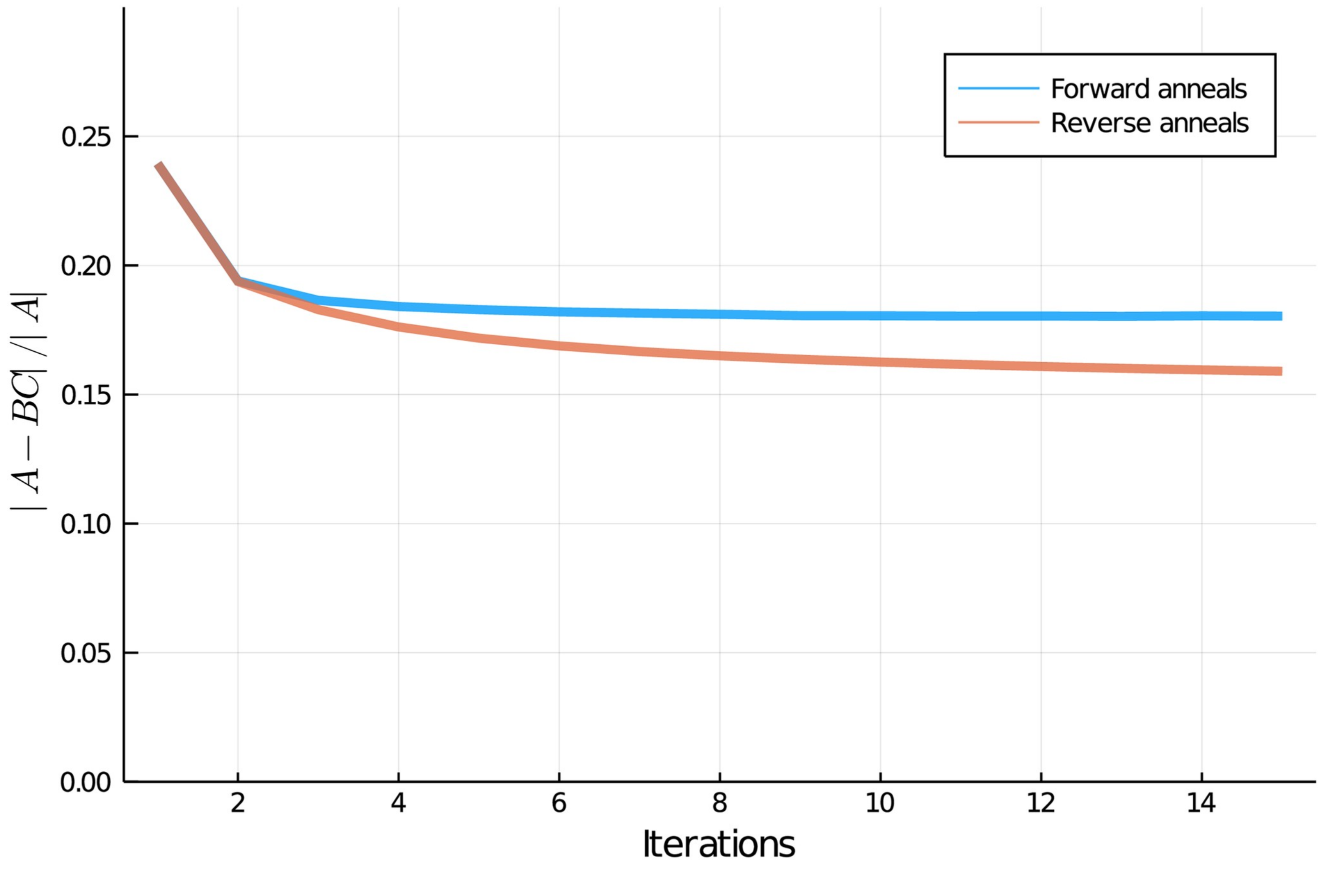

Fig 3. Comparison of forward and reverse anneal versions of the NBMF algorithm for fixed annealing time. Mean performance reported from five evaluations of the forward and reverse annealing versions of the NBMF algorithm, with 1000 forward anneals and 240 reverse anneals per QUBO, corresponding to a total QPU access time of 6182 seconds over the full evaluation of each algorithm. Standard deviation (not shown) was less than 1% of the mean.

https://doi.org/10.1371/journal.pone.0244026.g003

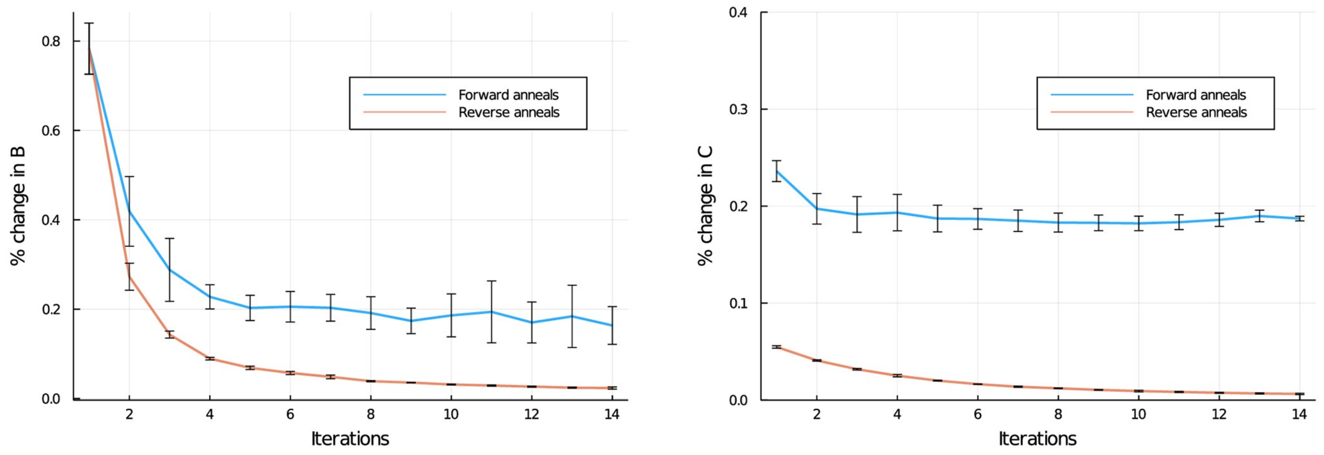

Fig 4. Iterative improvement of *B* and *C* matrices for forward and reverse anneal versions of the NBMF algorithm. Data taken during the evaluation of the algorithms as described in Fig 3.

https://doi.org/10.1371/journal.pone.0244026.g004

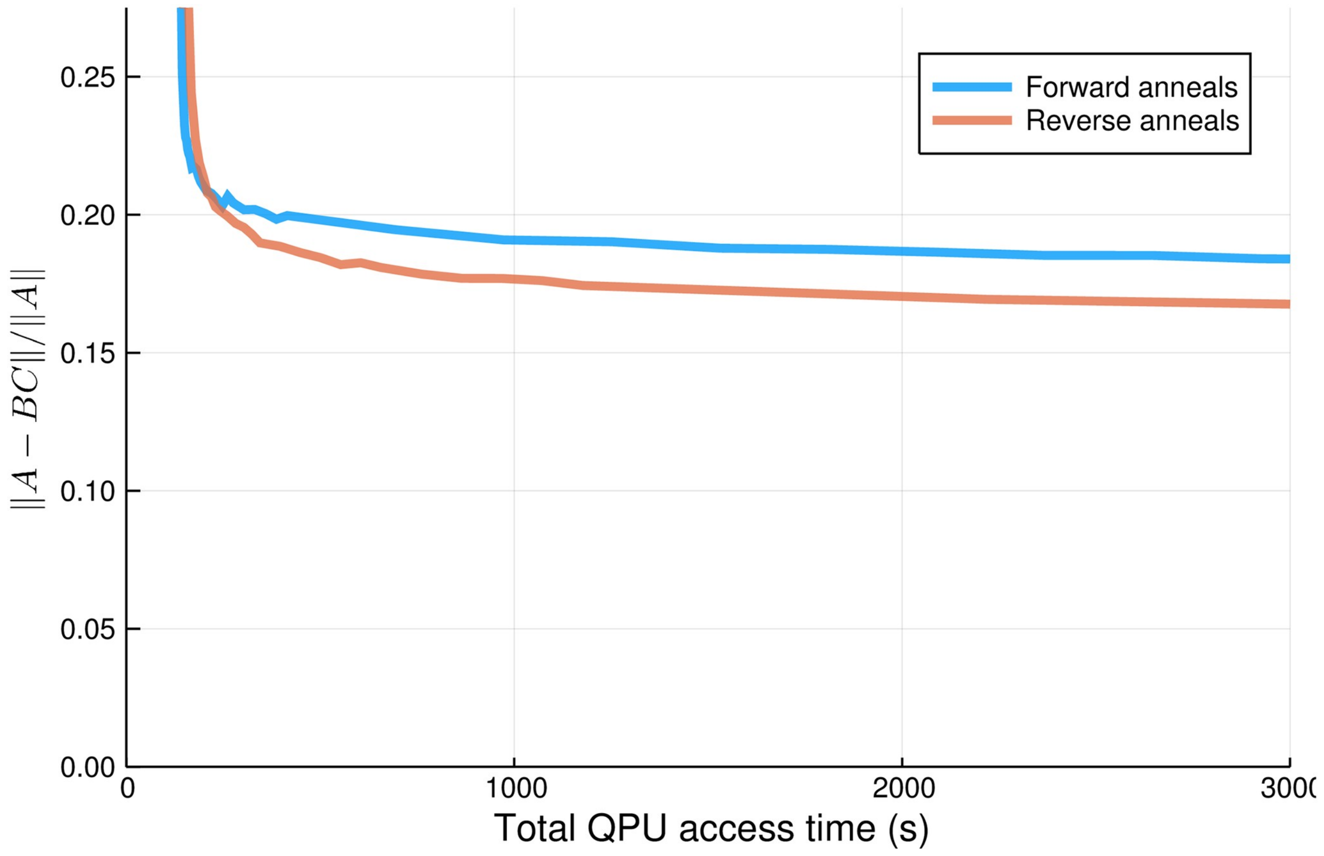

Fig 5. Comparison of forward and reverse anneal versions of NBMF algorithm after seven iterations for a range of annealing times. Data taken from 725 distinct evaluations of the NBMF algorithm, varying the number of forward and reverse anneals per evaluation. Reverse annealing results in up to a 12% increase in performance.

https://doi.org/10.1371/journal.pone.0244026.g005

improve. As the number of reverse annealing samples increases, the chance of finding a better/different solution increases significantly, resulting in improved performance.

We performed a benchmark using Gurobi that is similar to the benchmark performed previously for this problem [10] (Fig 6). The basic mechanism of the benchmark is to determine the time Gurobi requires to find a solution that is as good as or better than the solution found by reverse annealing. We call this time the "time to target." For this test we used data from one of the runs described in Fig 3, i.e. 240 reverse anneals per QUBO. The most notable change from the previous incarnation of this benchmark is that Gurobi uses the classical state that reverse annealing starts in as a starting point to perform its optimization, whereas it was not given a starting point in the previous benchmark. This was done to put it on more equal ground with the reverse annealing algorithm. The other changes are due to different classical hardware being used (a 2.4GHz 8-core Intel Core i9 processor) and an updated version of Gurobi (version 9.0.2). We ran Gurobi using 8 threads, whereas previously 1 thread was used. Fig 6 shows the result of the benchmark on each of the QUBOs where reverse annealing was performed in the course of a matrix factorization. Overall, the benchmark indicates that reverse annealing is performing well. Note that cases where reverse annealing did not improve on the previous solution were assigned a time to target of 0, and are not included in the plot. The sum of the individual time to targets was $\sim$68000 seconds. For reference, the total annealing time was 262 seconds and the total QPU access time was 6182 seconds.

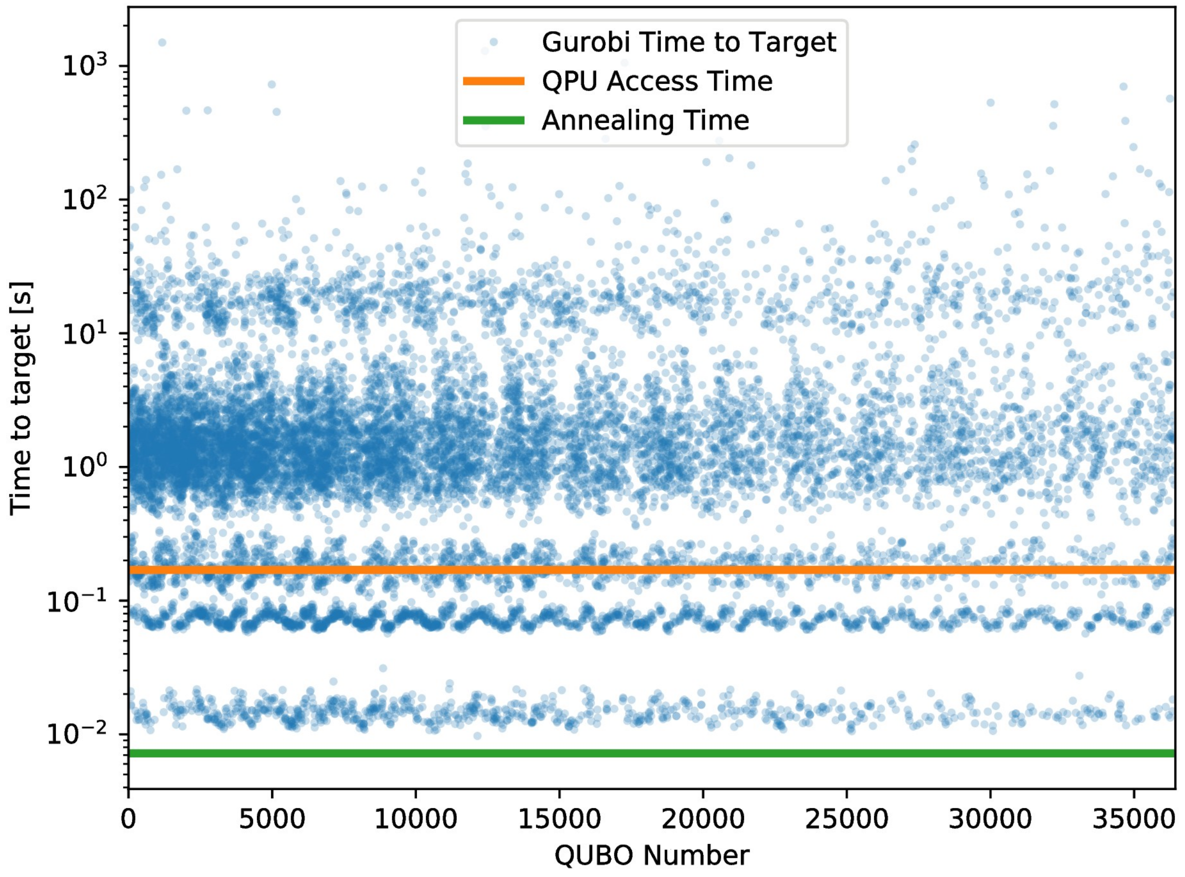

Fig 6. Comparison between Gurobi and reverse annealing. The time required for Gurobi to find a solution that is as good as or better than the solution found by reverse annealing for each QUBO in a factorization problem is shown in comparison to the QPU access time and the annealing time. Cases where reverse annealing failed to find a better solution are excluded from the plot. In many cases, the time required by Gurobi exceeds both the annealing time and the QPU access time.

https://doi.org/10.1371/journal.pone.0244026.g006

Conclusion

The results of this work suggest that reverse annealing improves the quality of the NBMF factorization by 12% for this application. This improvement is seen when the number of reverse anneals evaluated per QUBO is at least 7 (which is equivalent in QPU access time to 29 forward anneals). In [10], it was observed that quantum annealing had the largest performance gains relative to classical benchmarks in the short annealing timeframe, $\mathcal{O}(10)$ forward anneals per QUBO. Reverse annealing improves performance in the longer annealing timeframe, thus further establishing quantum annealing as a strong approach for non-negative binary matrix factorization.

In addition to characterizing the performance in terms of the quality of the factorization given a fixed time, it could be characterized in terms of how long it takes to obtain a factorization of a given quality. By this standard, reverse annealing would also perform well once the quality of the factorization is set sufficient low. Since NBFM with forward annealing tends to plateau at a worse factorization quality, the speed-up with reverse annealing would be very large once the factorization quality is set beyond this plateau.

Our results could be improved upon in several ways. First, it is possible that the optimal reverse anneal schedule could depend on how many iterations have already occured (i.e., as better solutions become harder to find). It is also our hope that future quantum annealing hardware will feature more rapid state initialization, as this accounts for over 98% of the additional time related to reverse annealing. This would improve the performance of NBMF with reverse annealing but leave the performance of NBMF with forward annealing unchanged. Lastly, the exact nature of the matrix being factorized appears to play a role in determining how effective the algorithm is, and this could be explored further.

Author Contributions

Conceptualization: John Golden.

Formal analysis: John Golden.

Funding acquisition: Daniel O'Malley.

Methodology: Daniel O'Malley.

Supervision: Daniel O'Malley.

Writing – original draft: John Golden.

Writing – review & editing: John Golden, Daniel O'Malley.

References

1. Danowitz Andrew, Kelley Kyle, Mao James, Stevenson John P, and Horowitz Mark. Cpu db: recording microprocessor history. *Communications of the ACM*, 55(4):55–63, 2012. https://doi.org/10.1145/2133806.2133822
2. Geer David. Chip makers turn to multicore processors. *Computer*, 38(5):11–13, 2005. https://doi.org/10.1109/MC.2005.160
3. Owens John D, Houston Mike, Luebke David, Green Simon, Stone John E, and Phillips James C. GPU computing. *Proceedings of the IEEE*, 96(5):879–899, 2008. https://doi.org/10.1109/JPROC.2008.917757
4. Monroe Don. Neuromorphic computing gets ready for the (really) big time. *Communications of the ACM*, 57(6):13–15, 2014.
5. Kadowaki Tadashi and Nishimori Hidetoshi. Quantum annealing in the transverse ising model. *Physical Review E*, 58(5):5355, 1998. https://doi.org/10.1103/PhysRevE.58.5355
6. Johnson Mark W, Amin Mohammad HS, Gildert Suzanne, Lanting Trevor, Hamze Firas, Dickson Neil, et al. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 2011. https://doi.org/10.1038/nature10012 PMID: 21562559
7. Gibney Elizabeth. D-wave upgrade: How scientists are using the world's most controversial quantum computer. *Nature*, 541(7638):447–448, 2017. https://doi.org/10.1038/541447b PMID: 28128267
8. Chancellor Nicholas. Modernizing quantum annealing using local searches. *New Journal of Physics*, 19(2):023024, 2017. https://doi.org/10.1088/1367-2630/aa59c4
9. Ohkuwa Masaki, Nishimori Hidetoshi, and Lidar Daniel A. Reverse annealing for the fully connected p-spin model. *Physical Review A*, 98(2):022314, 2018. https://doi.org/10.1103/PhysRevA.98.022314
10. O'Malley Daniel, Vesselinov Velimir V, Alexandrov Boian S, and Alexandrov Ludmil B. Nonnegative/binary matrix factorization with a d-wave quantum annealer. *PloS one*, 13(12):e0206653, 2018. https://doi.org/10.1371/journal.pone.0206653
11. Lee Daniel D and Seung H Sebastian. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. https://doi.org/10.1038/44565 PMID: 10548103
12. Pauca V. Paul, Shahnaz Farial, Berry Michael, and Plemmons Robert. Text Mining Using Non-Negative Matrix Factorizations. SIAM Proceedings Series. 2004.
13. Rajabi Roozbeh and Ghassemian Hassan. Spectral Unmixing of Hyperspectral Imagery Using Multi-layer NMF. *IEEE Geoscience and Remote Sensing Letters*, 12(1):38–42, 2015. https://doi.org/10.1109/LGRS.2014.2325874