



Dirac-3S Technical Reference: Continuous Solver Benchmarking and Performance Characterization

September 2026, Revision A

Quantum Computing Inc
quantumcomputinginc.com
(703) 436-2161

Contents

1	Executive Summary	5
2	Solvers Under Test	7
2.1	Dirac-3S	7
2.2	Hexaly	7
2.3	Projected gradient descent	8
3	Test Instances	8
3.1	QPLIB instances	9
3.2	Planted quadratic Instances	9
3.3	Maximum clique via Motzkin-Straus	10
4	Results	10
4.1	QPLIB	10
4.2	Planted non-convex StQP	11
4.3	Maximum clique	15
5	Discussion	18
5.1	QPLIB	18
5.2	Planted instances	18
5.3	Maximum clique	19
6	Conclusion	20
7	Outlook	21
A	Instance formats	22
A.1	Planted StQP (.npy)	22
B	Related Technical References	22
C	Contacting Support	22

Document Revision History

Version	Date	Changes
0.0.1	September, 2026	N/A

Notice and Disclaimer

This Technical Reference, including all text, images, diagrams, and other content contained herein, is the intellectual property of Quantum Computing Inc. All rights, including copyright, are reserved. No part of this User Guide may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language without the prior written permission of Quantum Computing Inc.

Any unauthorized use or reproduction of this Technical Reference violate copyright laws and international treaties. The contents of this Technical Reference are provided for informational purposes only and may be subject to change without notice. Quantum Computing Inc. assumes no responsibility for errors or omissions in this document or for any damages, whether direct, indirect, incidental, consequential, or otherwise, arising from the use or inability to use the information contained herein.

All trademarks, trade names, logos, and service marks mentioned in this User Guide belong to their respective owners and are hereby acknowledged.

The use of this Technical Reference is limited to the intended purpose of assisting users in the proper usage of the associated hardware and software. Any other use, including but not limited to commercial exploitation, distribution, or modification of the content, is strictly prohibited.

By accessing and using this Technical Reference, you acknowledge and agree to respect the copyright and intellectual property rights of Quantum Computing Inc. If you have any questions or requests regarding the use of this User Guide, please contact support@quantumcomputinginc.com

1 Executive Summary

Dirac-3S is a hybrid entropy quantum optimization machine combining quantum optics and digital electronics. Optimization problems are encoded into the photonic system, which evolves toward low-energy states representing candidate solutions [14]. Leveraging quantum fluctuations obtained through single-photon detection, Dirac-3S natively solves continuous-variable optimization problems with decision variables encoded in photon amplitudes. As a result, Dirac-3S natively operating as an optimizer for a challenging class of continuous non-convex optimization problems known as the *Standard Quadratic Problem* (StQP), which consists of a smooth quadratic objective over a simplex or scaled-simplex constraint set. Numerous industry applications related to selection, allocation and clustering use StQP framework for optimization [4]. A few of such examples are pooling and blending where where mixing streams with varying concentrations creates non-convex constraints [12], continuous trajectory planning for drones or autonomous vehicles navigating non-convex obstacle fields [7], sparse source localization and sensor network node positioning under Euclidean distance bounds in signal processing [1], portfolio selection without short positions, resource allocation with a fixed budget, evolutionary game theory (replicator dynamics), and a range of relaxations arising in machine learning all reduce to or pass through a standard quadratic program.

This Technical Reference characterizes the performance of Dirac-3S across three benchmark suites:

Benchmark Suite	Purpose	What We Evaluate
QPLIB	Published continuous quadratic optimization benchmarks with established reference solutions.	Solution quality, optimality gap, and ground-state probability.
Planted StQP	Synthetic benchmark instances with known global optima, allowing controlled variation of problem size and coefficient dynamic range.	Solution quality, success probability, scalability, and computation time as a function of problem size.
DIMACS Maximum Clique	Classical combinatorial optimization benchmarks reformulated as StQP through the Motzkin-Straus theorem.	Solution quality, clique recovery performance, and sensitivity to problem structure and graph size.

Table 1: Benchmark suites used to evaluate Dirac-3S performance.

The Technical Reference benchmarks Dirac-3S against two complementary classical baselines. Projected Gradient Descent (PGD) serves as an algorithmic reference, providing a widely adopted gradient-based optimization approach that operates on the same scaled-simplex feasible geometry as Dirac-3S. Hexaly serves as a commercial benchmark, representing a state-of-the-art optimization platform actively deployed to solve real-world planning and operations problems, including scheduling, routing, logistics, and supply chain optimization. Together, these baselines

provide perspective on Dirac-3S performance relative to both established optimization methodologies and commercially deployed optimization solutions.

Key Findings

- Dirac-3S demonstrated competitive or superior solution quality across synthetic, graph-based, and industry-standard benchmark problems.
- Dirac-3S more frequently identified the highest-quality maximum clique solutions on DIMACS benchmark instances than any other evaluated method.
- Dirac-3S was the only solver among the evaluated methods to achieve zero optimality gap across all synthetic benchmark instances with known global optima, spanning problem sizes from 2,000 to 9,980 variables.
- Dirac-3S exhibited favorable computational scaling characteristics, with solution times increasing only weakly as problem size grew to nearly 10,000 variables.

Many important NP-hard optimization problems can be reformulated, relaxed, or mapped into simplex-constrained standard quadratic programming (StQP) formulations. This class extends beyond native continuous optimization and encompasses a broad range of discrete, binary, and combinatorial optimization problems such as Maximum clique problem via Motzkin-Straus theorem [13, 2], Binary and graph partitioning problems [3], and general discrete and fractional problems [11]. Consequently, the benchmarking results presented in this Technical Reference may provide useful insight even for users whose original applications are formulated as binary optimization, QUBO, mixed-integer, or other constrained optimization problems. The Maximum Clique benchmarks included in this study illustrate this relationship. Although Maximum Clique is inherently a discrete graph optimization problem, it can be reformulated as a continuous optimization problem through the Motzkin-Straus theorem, enabling direct evaluation on the Dirac-3S platform. Similar reformulations exist for many other optimization problems. Depending on the problem structure and mapping approach, such reformulations may introduce trade-offs related to solution quality, problem size, runtime, or interpretability of the resulting solutions.

This benchmarking study focuses specifically on simplex-constrained continuous quadratic optimization and should not be interpreted as a comprehensive evaluation of all optimization problem classes. QCi will continue to publish additional Technical References covering other problem formulations, including QUBO, binary optimization, mixed-integer optimization, integer programming, and other application-specific workloads.

Interpretation of Results

The objective of this Technical Reference is not to establish a universal ranking among optimization solvers, but rather to characterize how the evaluated approaches behave across different problem structures, scales, and levels of difficulty. The results are intended to help users understand the performance characteristics of Dirac-3S and assess its suitability for their specific applications.

2 Solvers Under Test

2.1 Dirac-3S

Dirac-3S is a commercial optimization solver based on Quantum Computing Inc.'s entropy computing paradigm, a measurement-feedback optimization framework that leverages stochastic photonic processes and electronic control to search complex solution spaces. The solver represents optimization variables as high-dimensional temporal modes and iteratively updates candidate solutions through a closed-loop process involving probabilistic sampling, evaluation, and feedback.

Unlike conventional Ising-based optimization systems that operate primarily on binary variables and pairwise interactions, Dirac-3S natively supports discrete and continuous optimization formulations with higher-order interaction terms. The underlying architecture is capable of representing objective functions containing linear, quadratic, cubic, quartic, and quintic terms without requiring problem quadratization or large numbers of auxiliary variables. The solver natively supports binary, integer, and continuous decision variables and directly optimizes polynomial objective functions containing first- through fifth-order interactions:

$$E(v) = \sum_i C_i v_i + \sum_{i,j} J_{ij} v_i v_j + \sum_{i,j,k} T_{ijk} v_i v_j v_k + \sum_{i,j,k,l} Q_{ijkl} v_i v_j v_k v_l + \sum_{i,j,k,l,m} P_{ijklm} v_i v_j v_k v_l v_m. \quad (1)$$

The solver performs optimization under configurable constraints and employs quantum fluctuations originating from single photon counting to improve exploration of non-convex landscapes and reduce susceptibility to local minima.

2.2 Hexaly

Hexaly is a commercial general-purpose mathematical programming solver built on a portfolio of local-search and direct-search heuristics with exact components. We use Hexaly15.0 [9] to solve the problem instances. We declare n continuous decision variables $x \geq 0$, one linear equality constraint $\sum_i x_i = R$, and minimise the objective. The quadratic form is supplied as a black-box external function together with its analytic gradient:

$$f_{\text{ext}}(x) = x^T Q x, \quad \nabla f_{\text{ext}}(x) = 2Qx. \quad (2)$$

Supplying the gradient rather than letting Hexaly finite-difference helps it scale well at large number of variables.

Hexaly solver has a user defined `time_limit` parameter. For non-convex problems, Hexaly tends to keep running until the limit expires, irrespective of whether objective function is saturated to a value or not. We put a large time limit to the solver. The definition of "large" varies for different experiments. We specify the limit that we chose for each kind of data suits in the results section.

2.3 Projected gradient descent

Projected gradient descent (PGD) is the simplest method that correctly respects the geometry of StQP (4). We project the solution space onto Δ_R using sort-and-threshold algorithm [8, 5]. This algorithm sorts v descending as u , finds $\rho = \max\{j : u_j - (\sum_{r \leq j} u_r - R)/j > 0\}$, and returns $\max(v - \theta, 0)$ with $\theta = (\sum_{r \leq \rho} u_r - R)/\rho$. With a fixed step size η ,

$$x^{(t+1)} = \Pi_{\Delta_R}(x^{(t)} - \eta(2Qx^{(t)} + c)), \quad (3)$$

terminated when $\|x^{(t+1)} - x^{(t)}\|_2 < \text{tol}$ or after max_iter iterations. Here Π_{Δ_R} is the exact Euclidean projection onto Δ_R .

A single PGD run is a descent method and returns a local minimum. We therefore run N_{restarts} independent runs. The first run is from the simplex centre $x_i^{(0)} = R/n$, the remainder are from $x^{(0)} \sim \text{Unif}[0, 1]^n$ rescaled to sum to R , with a fixed seed. We report the best objective over all restarts. We have set $N_{\text{restarts}} = 100$ and $\text{max_iter} = 2 \times 10^4$ throughout all our experiments.

Table 2: Solver configurations.

	Dirac-3S	Hexaly	PGD (multi-start)
Type	sampler (hardware)	commercial local search	reference baseline
Interface	cloud API	Python API	NumPy
Constraint	native sum constraint	linear equality	exact simplex projection
Objective	sparse polynomial	black-box + gradient	explicit Q, c
Stochastic	yes	yes	yes (random restarts)
Key parameters	<code>relaxation_schedule</code> , <code>num_samples</code>	<code>time_limit</code> , stagnation patience	<code>restarts</code> , <code>lr</code> , <code>max_iter</code>

3 Test Instances

Throughout, n is the number of decision variables, $Q \in \mathbb{R}^{n \times n}$ is a symmetric matrix, $c \in \mathbb{R}^n$, and $R > 0$ is the sum constraint. We consider

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) = x^T Q x + c^T x \\ \text{s.t.} \quad & \sum_{i=1}^n x_i = R, \quad x_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (4)$$

We write the feasible set as the scaled simplex

$$\Delta_R = \left\{ x \in \mathbb{R}_{\geq 0}^n : \sum_i x_i = R \right\}, \quad (5)$$

so Δ_1 is the standard unit simplex. When $c = 0$ the problem is a standard quadratic program (StQP) [3]. Q is not assumed positive semidefinite. When Q is indefinite, Eqn. (4) is non-convex and generally has many local minima. We choose three different types of problem instances with the above general structure. Those are the following.

3.1 QPLIB instances

QPLIB [6] is the standard library of quadratic programming instances. We take three instances of type QCL with continuous variables, quadratic objective, linear constraints. Each of these instances is exactly a standard quadratic program with a single equality constraint $\sum_i x_i = 1$ with $x \geq 0$.

Table 3: QPLIB instances. All are minimisation problems of type QCL with one constraint $\sum_i x_i = 1$, $x \geq 0$.

Instance	n	quadratic terms	R	best known objective
QPLIB_0018	50	1 275	1	-6.38601498
QPLIB_2712	200	20 100	1	0.01286883
QPLIB_2761	500	125 250	1	0.00104853

3.2 Planted quadratic Instances

We generate Q so that a planted solution x^* is the known global minimiser, with a known objective value f^* and a known support size k . This gives us control of sparsity and scaling, i.e, we can generate instances with desired values of k and n . Since we know the exact solution, it is easy to verify solvers' solution quality.

The instance family is parameterised by

n	number of variables
k	size of the planted support
R	sum constraint (fixed at 100)
ub	contrast parameter
s	random seed (fixed at 100).

The planted optimum is the uniform point on K ,

$$x_i^* = \begin{cases} R/k, & i \in K \\ 0, & i \notin K, \end{cases} \quad f^* = \frac{R^2}{k^2} \sum_{i,j \in K} Q_{ij} = \bar{q} R^2, \quad (6)$$

which for $R = 100$ and $\bar{q} = 1.2$ gives $f^* = 12000$ for every instance in the family, independent of n and k . Holding f^* fixed across the family is convenient: relative gaps are directly comparable across instance sizes without renormalisation.

The contrast parameter ub controls dynamic range of the instances. The lower bound of the absolute value of the matrix elements is set to 1. Upper bound of the absolute value is ub . When ub is 10, the dynamic range is ~10 dB, and for $ub = 100$, the dynamic range is ~20 dB.

Instances are stored as NumPy `.npy` arrays holding the flattened upper triangle of Q (including the diagonal), $n(n+1)/2$ entries in `float32`. The full symmetric matrix is reconstructed by mirroring.

3.3 Maximum clique via Motzkin-Straus

We test max-clique benchmark problems from DIMACS second-challenge maximum-clique benchmark [10]. Max-clique problems can be formulated as StQP via Motzkin-Straus which can be stated as follows.

Theorem 1 (Motzkin–Straus [13]). *Let G be a simple undirected graph on n vertices with adjacency matrix A (symmetric, zero diagonal, 0/1 entries), and let $\omega(G)$ be its clique number. Then*

$$\max_{y \in \Delta_1} y^T A y = 1 - \frac{1}{\omega(G)}. \quad (7)$$

For a scaled simple $x \in \Delta_R$, the solution becomes

$$\max_{x \in \Delta_R} x^T A x = R^2 \left(1 - \frac{1}{\omega(G)} \right). \quad (8)$$

Given a graph $G = (V, E)$ with adjacency matrix A , we set $Q = -A$, $c = 0$ and solve (4) over Δ_R . By Theorem 1 and the scaling relation (8), the optimal energy is

$$f^* = -R^2 \left(1 - \frac{1}{\omega(G)} \right), \quad \text{equivalently} \quad \omega(G) = \frac{1}{1 + f^*/R^2}. \quad (9)$$

Any feasible x therefore yields a continuous estimate $\hat{\omega}(x) = (1 + f(x)/R^2)^{-1}$, which is a lower bound on $\omega(G)$ up to the solver's optimality gap and takes non-integer values in general. Because the formula divides the energy by R^2 it is invariant to the choice of R , which is why runs at $R = 1$ and $R = 100$ are directly comparable.

The continuous estimate of ω is not guaranteed to be a maximum clique. Solution found using stochastic solver might not be a clique at all as there are spurious maximisers supported on non-cliques [15]. We therefore apply two distinct operations: (i) Extract $S = \{v : x_v > \tau \max_u x_u\}$ with a relative threshold τ , and test whether $A[S, S]$ has exactly $|S|(|S| - 1)$ ones, i.e., whether S induces a clique; and (ii) Grow a clique by repeatedly taking the highest-weight remaining candidate and intersecting the candidate set with its neighbourhood. The result is always a clique, so its size is a valid lower bound on $\omega(G)$.

4 Results

4.1 QPLIB

We run Dirac-3 on `relaxation_schedule = 1` with 100 samples. We also run PGD with 100 random initial states. We set `hexaly time_limit = 300` seconds. We show the best solution form each solver in Table 4. We also shown reported best known solution from QPLIB website [6]. We have also shown the energy distribution over 100 samples for Dirac-3S and PGD solver in Figure 1.

Table 4: QPLIB results. Best objective value found by each solver, against the best known objective reported by QPLIB. Dirac-3S and PGD values are the best over 100 samples and 100 random restarts respectively; Hexaly was given a 300 second time limit. Bold marks a value below the published best known.

Instance	n	best known	Dirac-3S	Hexaly	PGD
QPLIB_0018	50	-6.38601498	-6.3860168	-6.38601109	-6.38601495
QPLIB_2712	200	0.01286883	0.0128688	0.01286883	0.01286883
QPLIB_2761	500	0.00104853	0.0010485	0.00105576	0.00104853

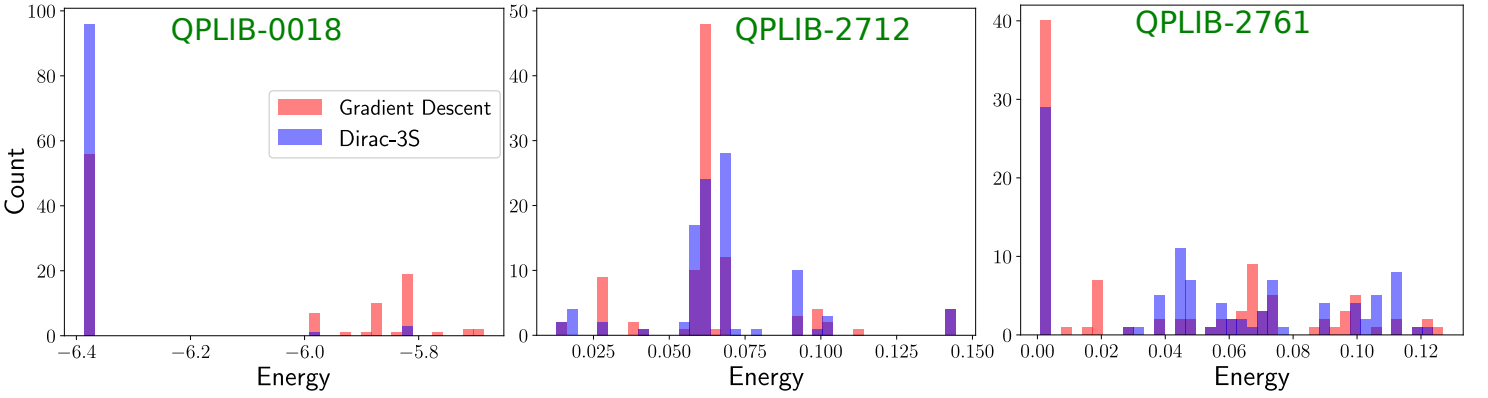


Figure 1: Energy distribution over 100 samples for Dirac-3S (blue) and over 100 random restarts for multi-start PGD (red) on the three QPLIB instances.

4.2 Planted non-convex StQP

We compare solution quality and time for the three solvers. Solution quality is determined by the relative percent gap defined as follows:

$$\text{gap(\%)} = \frac{\text{solver energy} - \text{optimal energy}}{\text{optimal energy}} \times 100 .$$

All time are in seconds. We ran Dirac-3S on `relaxation_schedule = 1`. Dirac-3S time includes pre-processing time, device time, and post-processing time. It should be noted that we ran every instance for 10 samples and report the best sample among them. We report time for the best sample as well as the total time for 10 samples. We ran PGD solver for two values of sample sizes, 10 and 100. We report relative gap and time for both of them. We show energy distribution for both Dirac-3S and PGD solver over 10 samples for a representative set of instances in Figure 4. We set Hexaly `time_limit = 3600`. We set a stagnation time limit 600 seconds, meaning if Hexaly objective does not improve for 600 seconds, we stop the solver. Solver time is reported as the first instance when the objective saturates. Figure 2 shows energy evolution of Hexaly solver with time for a representative instance. Results for contrast parameter $ub = 10$ is listed in table 5. Dirac-3S solver found optimal solution for all the test instances, when Hexaly struggles to close the gap with the optimal for almost 90 percent of the instances. Gradient descent performs similar to Hexaly with 10 restarts. Its performance improves when restarted with 100 random initial states.

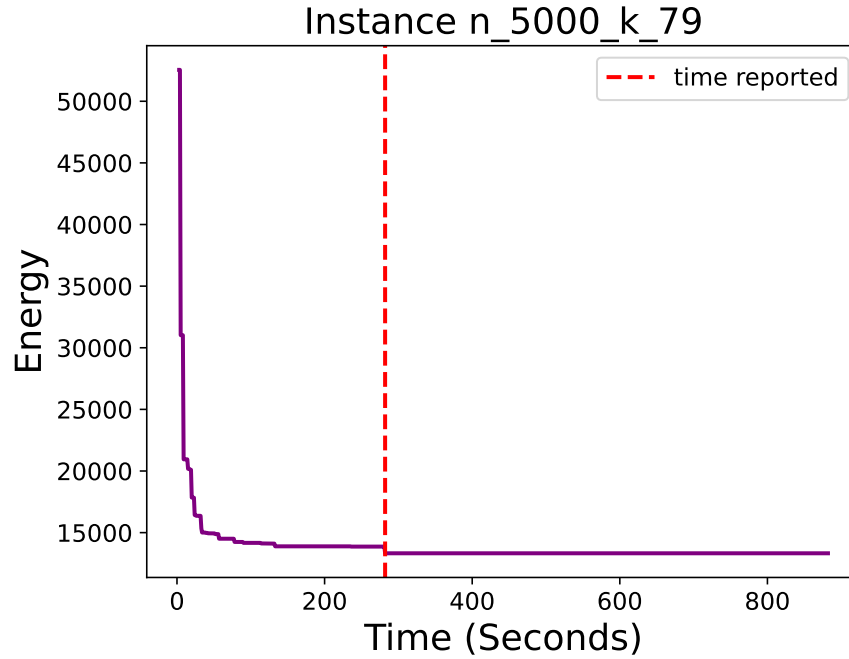


Figure 2: We report Hexaly time as the first instance when the energy starts to saturate. This plot shows evolution of Hexaly energy with time for Instance `n_5000_k_79` with $ub = 100$. Red line indicates the first instance when the energy saturates.

Table 5: Planted StQP, contrast $ub = 10$. Objective values against the planted optimum $f^* = 12000$. Gap is relative, in percent. Support is $|\text{supp}(x_{\text{best}})|$, to be compared with the planted k . Dirac-3S was run on `relaxation_schedule=1`. Time reported for Dirac-3S includes pre-process + device-time + post-process

Instance	n	k	Dirac-3S				Hexaly			PGD-10 sample			PGD-100 sample		
			gap	supp	time 1-sample	time 10-sample	gap	supp	time	gap	supp	time	gap	supp	time
n2000_k44	2000	44	0.0	44	12	166	0.0002	44	37	6.80	14	2.62	0.0	44	29
n4000_k55	4000	55	0.0	55	70	620	9.99	19	698	7.27	14	27	0.0	55	741
n5000_k71	5000	71	0.0	71	41	330	11.39	22	258	10.31	16	52	10.19	18	607
n6000_k78	6000	78	0.0	78	51	700	10.67	25	603	5.99	15	148	0.0	78	2511
n7000_k86	7000	86	0.0	86	93	574	12.59	31	962	0.0	86	83	0.0	86	3195
n8000_k95	8000	95	0.0	95	66	608	11.12	33	278	11.55	19	205	0.0	95	2218
n9000_k95	9000	95	0.0	95	187	1560	12.86	36	1078	13.52	20	205	11.57	20	1946
n9980_k97	9980	97	0.0	97	232	1691	14.65	59	364	13.03	19	250	12.49	19	2528

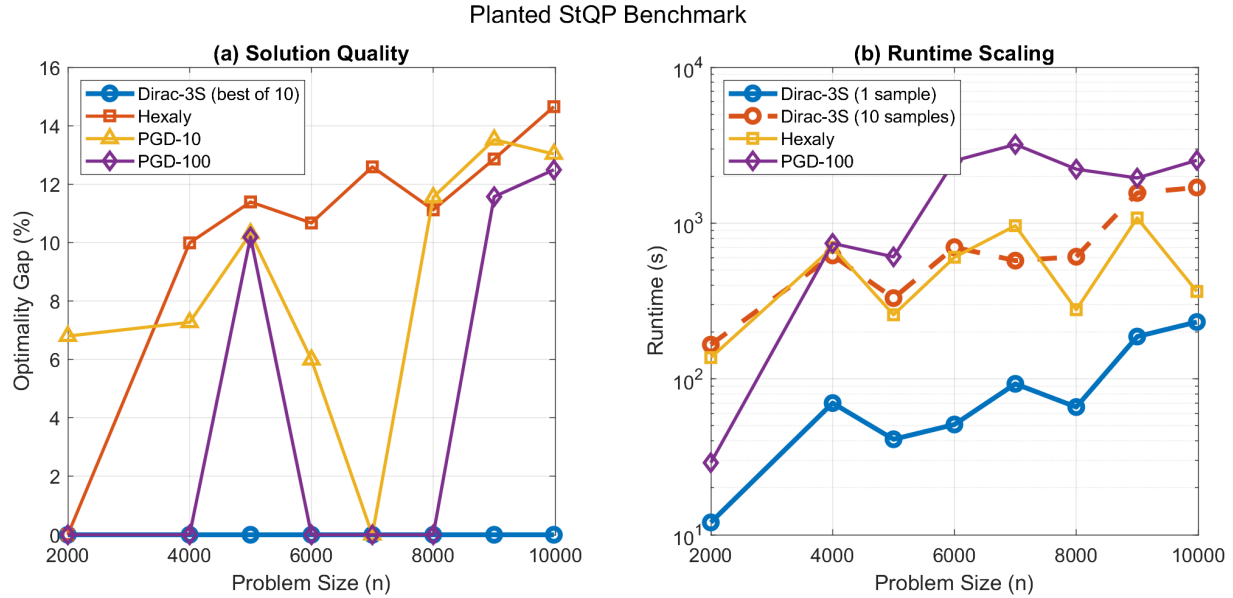


Figure 3: Performance on planted StQP benchmark instances with increasing problem size. **(a)** Optimality gap relative to the planted optimum. Dirac-3S recovered the planted optimum for all benchmark instances, maintaining a 0% optimality gap across problem sizes ranging from 2,000 to 9,980 variables. In contrast, both Hexaly and PGD exhibited increasing deviations from the planted optimum as problem size increased. **(b)** Runtime scaling comparison. Dirac-3S runtimes are reported for both a single sample execution and the full 10-sample execution used to obtain the reported best-of-10 solutions. Even when accounting for all 10 samples, runtime growth remains moderate while preserving perfect solution quality across all benchmark instances.

Results for the contrast parameter $ub = 100$ are given in Table 6. As discussed earlier, this parameter controls the dynamic range of the problem instances. While increasing the dynamic range does not inherently make the optimization problem more difficult, it provides an important stress test for physical optimization hardware. In Dirac-3S, the representable dynamic range is ultimately constrained by practical hardware considerations, including modulation fidelity, DAC and ADC resolution, and other analog system imperfections. Therefore, evaluating performance at higher contrast levels helps characterize solver robustness near the operational limits of the hardware platform. Despite operating at a significantly increased dynamic range, Dirac-3S continued to recover the optimal solution efficiently and consistently, outperforming both comparison solvers in terms of solution quality while maintaining competitive execution times. These results demonstrate that Dirac-3S remains effective even as problem coefficients approach the edge of the system's supported dynamic range. Figure 4 shows the distribution of objective values obtained over 10 samples for Dirac-3S and PGD. The distributions clearly demonstrate that Dirac-3S produces solutions tightly concentrated around the known optimum, whereas PGD exhibits a much broader distribution, with only a small fraction of runs achieving objective values close to the optimum. This behavior is expected because PGD is fundamentally a local optimization method. Depending on the initialization point, the solver converges to the nearest local minimum and only occasionally reaches the global optimum. In contrast, Dirac-3S is designed as a global optimization solver and

maintains strong performance even in the presence of increased coefficient contrast and dynamic range requirements.

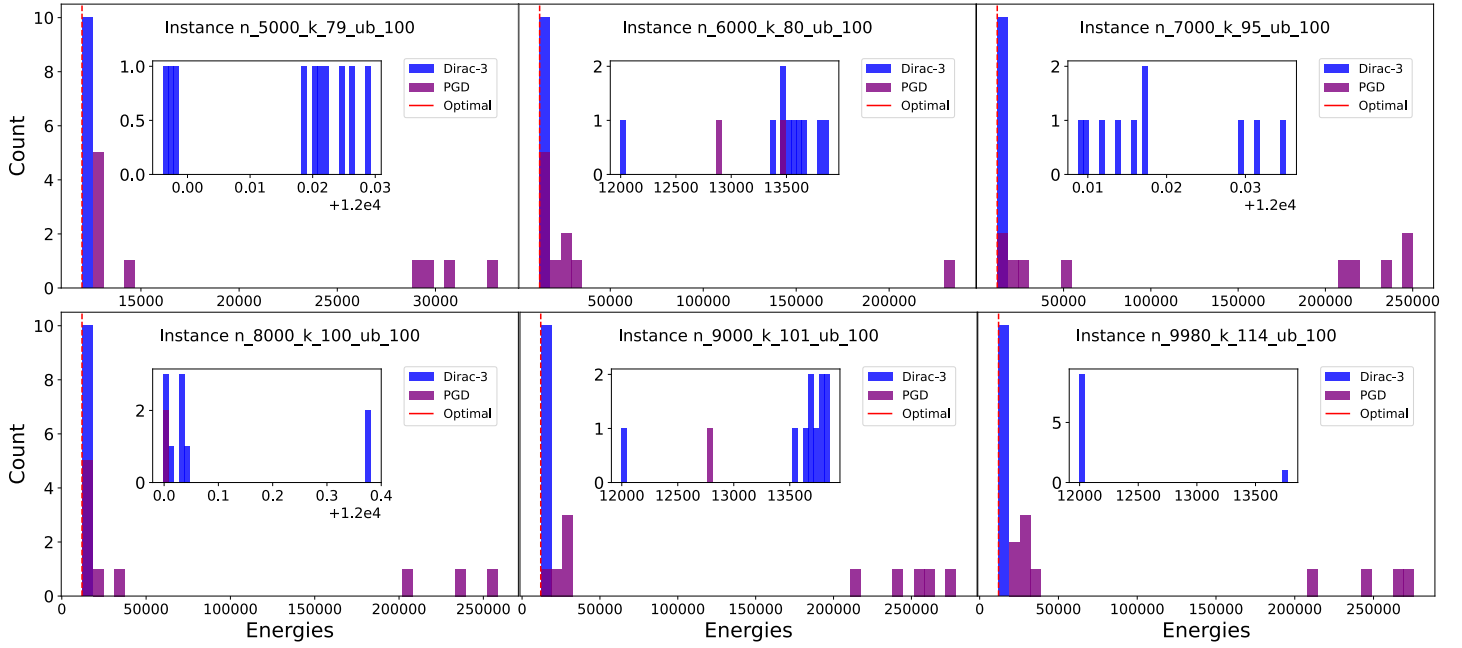


Figure 4: Energy distribution of Dirac-3S solver and PGD solver energy over 10 samples for representative instances of size 5000, 6000, 7000, 8000, 9000, and 9980 with $ub = 100$. Inset plots show energy distribution close to the known optimal. Due to the stochastic nature of the solver there are a certain probabilities of getting a sub-optimal solution for both PGD and Dirac-3S. But Dirac-3S has higher probability of escaping local minima and obtaining the optimal solution with relatively small sample size

Table 6: Planted StQP, contrast $ub = 100$. Columns as in Table 5.

Instance	n	k	Dirac-3S				Hexaly			PGD-10 sample			PGD-100 sample		
			gap	supp	time 1-sample	time 10-sample	gap	supp	time	gap	supp	time	gap	supp	time
n2000_k43	2000	43	7.56	13	50	531	0.0015	43	200	7.31	13	8	7.56	14	1447
n4000_k55	4000	55	9.59	14	111	783	10.38	15	93	10.23	14	202	10.08	15	1902
n5000_k79	5000	79	0.0	79	76	1015	11.03	19	281	11.29	14	331	0.0	79	5140
n6000_k80	6000	80	0.0	80	52	431	11.82	19	793	12.02	14	624	10.33	16	23382
n7000_k95	7000	95	0.0	95	83	718	0.018	95	479	11.80	14	2361	0.0	95	13225
n8000_k100	8000	100	0.0	100	78	989	0.024	100	513	12.81	14	1486	0.0	100	20495
n9000_k101	9000	101	0.0	101	76	376	20.86	83	832	15.54	14	1975	14.01	15	18647
n9980_k114	9980	114	0.0	114	144	844	0.04	114	423	66.69	29	5418	14.5	15	52750

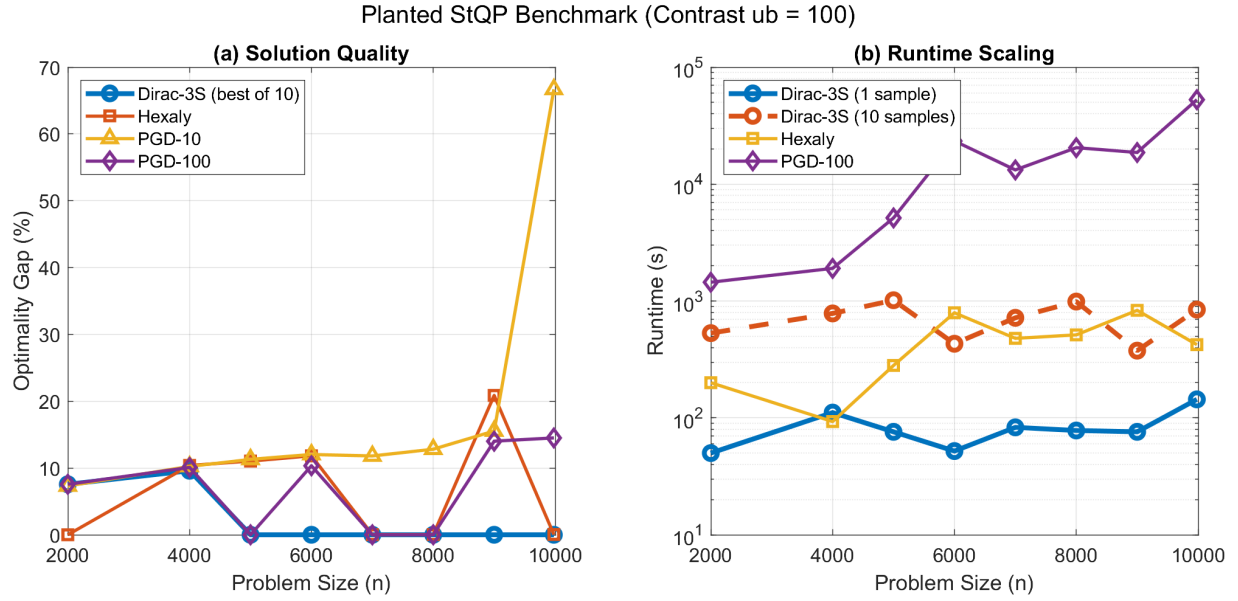


Figure 5: Performance on planted StQP instances with increased coefficient contrast ($ub = 100$). Although increasing dynamic range does not inherently increase optimization difficulty, it serves as a stress test for physical optimization hardware. Dirac-3S maintains near-optimal solution quality while operating closer to the representable limits imposed by analog hardware components such as modulators, DACs, and ADCs. (Read more explanation in "Dynamic Range" section of Dirac-3S User Guide) **(a)** Optimality gap relative to the planted optimum. **(b)** Runtime scaling comparison.

4.3 Maximum clique

We run Dirac-3S on four available `relaxation_schedule` with 100 samples. We report the best among those 100 samples for each relaxation schedule. For PGD, we ran with 100 random initial states and report the best. For Hexaly, we set the time limit to be 300 seconds. When a solver found the best solution among three solver, we marked that in bold. We also report the best known solution form DIMACS website [10] for comparison.

Table 7: Results for DIMACS second-challenge maximum-clique benchmark instances. The table shown instance name, number of variables (nodes), best known solution, and best solutions from out test solvers. Rows are shaded gray when all solvers found the optimal. Bold-faced numbers used when one solver found a solution that is superior than the rest.

Instance	Nodes	Best known	Dirac-3S				Hexaly	PGD
			RS= 1	RS= 2	RS= 3	RS= 4		
brock200_1	200	21	20	20	20	20	20	20
brock200_2	200	12	10	10	10	10	10	10
brock200_3	200	15	13	13	13	13	13	13
brock200_4	200	17	16	16	15	15	16	16
brock400_1	400	27	24	24	24	24	24	24
brock400_2	400	29	24	24	24	23	23	23
brock400_3	400	31	25	23	24	23	24	24
brock400_4	400	33	23	23	24	24	24	23

brock800_1	800	23	19	19	19	19	19	19
brock800_2	800	24	20	19	20	19	20	19
brock800_3	800	25	19	21	19	21	19	20
brock800_4	800	26	19	19	19	20	19	20
C125.9	125	34	34	34	34	34	34	34
C250.9	250	44	43	43	42	44	42	42
C500.9	500	≥ 57	52	53	51	52	52	52
C1000.9	1000	≥ 68	61	63	63	61	62	62
C2000.5	2000	16	15	14	14	15	14	14
C2000.9	2000	≥ 80	68	70	67	69	67	68
C4000.5	4000	18	15	15	15	15	12	15
DSJC500.5	500	13	12	12	12	12	12	12
DSJC1000.5	1000	15	14	13	13	13	14	14
keller4	171	11	9	9	9	9	9	9
keller5	776	27	19	17	18	18	16	16
keller6	3361	59	35	33	34	33	31	36
hamming6-2	64	32	32	32	32	32	32	32
hamming6-4	64	4	4	4	4	4	4	4
hamming8-2	256	128	128	128	128	128	128	128
hamming8-4	256	16	16	16	16	16	16	16
hamming10-2	1024	512	512	512	512	512	512	512
hamming10-4	1024	≥ 40	36	36	35	35	33	34
gen200-p0.9-44	200	44	38	38	38	38	38	38
gen200-p0.9-55	200	55	42	40	40	41	40	40
gen400-p0.9-55	400	55	50	50	50	51	50	51
gen400-p0.9-65	400	65	49	49	54	49	51	49
gen400-p0.9-75	400	75	49	50	49	52	44	52
c-fat200-1	200	12	12	12	12	12	12	12
c-fat200-2	200	24	24	24	24	24	24	24
c-fat200-5	200	58	58	58	58	58	58	58
c-fat500-1	500	14	14	14	14	14	14	14
c-fat500-2	500	26	26	26	26	26	26	26
c-fat500-5	500	64	64	64	64	64	64	64
c-fat500-10	500	126	126	126	126	126	126	126
johnson8-2-4	28	4	4	4	4	4	4	4
johnson8-4-4	70	14	14	14	14	14	14	14
johnson16-2-4	120	8	8	8	8	8	8	8
johnson32-2-4	496	≥ 16	16	16	16	16	16	16
p_hat300-1	300	8	8	8	8	8	8	8
p_hat300-2	300	25	25	25	25	25	25	25
p_hat300-3	300	36	36	36	35	34	35	36
p_hat500-1	500	9	9	9	9	9	9	9
p_hat500-2	500	36	36	36	36	36	36	36
p_hat500-3	500	50	48	49	49	49	49	49
p_hat700-1	700	11	10	10	9	9	9	10
p_hat700-2	700	44	44	44	44	44	44	44
p_hat700-3	700	62	61	62	61	60	62	59
p_hat1000-1	1000	10	10	10	10	10	9	9
p_hat1000-2	1000	46	45	46	45	45	45	44
p_hat1000-3	1000	68	64	64	64	64	64	64
p_hat1500-1	1500	12	11	11	10	11	10	10
p_hat1500-2	1500	65	64	64	65	64	63	65
p_hat1500-3	1500	94	92	92	92	91	92	91
san200_0.7_1	200	30	16	16	15	15	16	15
san200_0.7_2	200	18	12	12	12	12	12	13
san200_0.9_1	200	70	47	45	45	46	47	46
san200_0.9_2	200	60	39	41	40	39	60	40
san200_0.9_3	200	44	36	36	36	36	35	37
san400_0.5_1	400	13	7	7	7	7	7	7
san400_0.7_1	400	40	20	20	20	20	20	20
san400_0.7_2	400	30	15	15	15	15	15	16
san400_0.7_3	400	22	13	12	14	13	12	13
san400_0.9_1	400	100	55	100	100	100	100	88
san1000	1000	15	8	8	8	8	8	8
sanr200_0.7	200	18	18	18	18	18	18	18
sanr200_0.9	200	42	41	41	41	41	41	41
sanr400_0.5	400	13	13	12	13	12	12	12
sanr400_0.7	400	21	20	20	21	20	20	21

MANN_a9	45	16	16	16	16	16	16	16
MANN_a27	378	126	117	117	117	117	125	125
MANN_a45	1035	345	330	330	330	330	330	342
MANN_a81	3321	≥ 1100	1080	1080	1080	1080	1096	1096

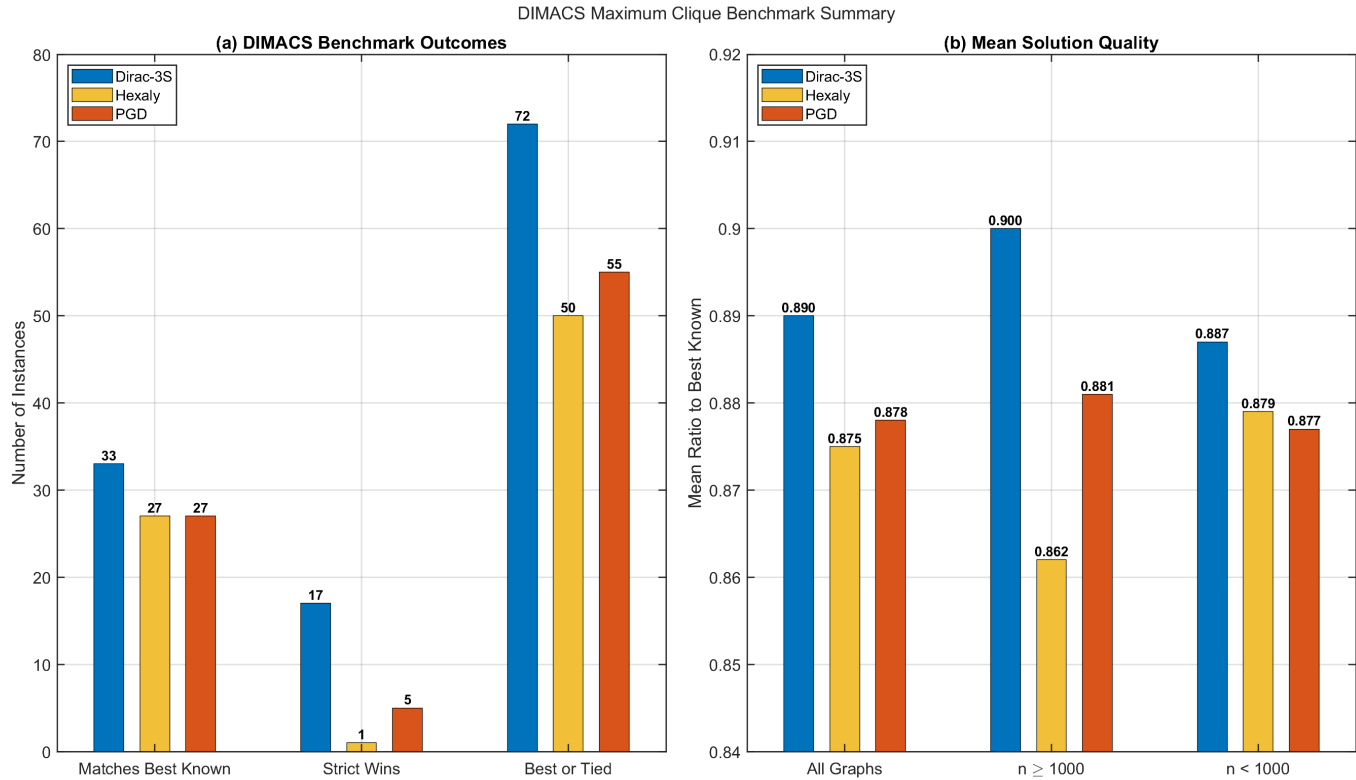


Figure 6: Aggregate DIMACS maximum-clique benchmark results over the 79 graph instances of Table 7 for which all three solvers reported results (MANN_a81 excluded). **(a)** Benchmark outcome statistics showing the number of instances on which each solver matched the best-known clique size, achieved a strict win over both competing solvers, or produced the best or a tied-best solution among the evaluated methods. A strict win indicates an instance where a solver returned a larger clique than both competing approaches. Best-or-tied counts instances where a solver attained the largest clique found by any of the three methods. **(b)** Mean solution quality, measured as the ratio between the clique size returned by a solver and the best-known clique size. Results are reported for the entire benchmark suite, large-scale instances ($n \geq 1000$), and smaller instances ($n < 1000$). Higher values indicate solutions closer to the best-known results.

Benchmark Highlight

Dirac-3S achieved the strongest overall performance across the DIMACS benchmark suite, matching the best-known clique size on 33 instances, producing the highest number of strict wins (17), and achieving the highest mean ratio to the best-known solution across both the full benchmark set and the large-scale instances ($n \geq 1000$).

5 Discussion

5.1 QPLIB

The three QPLIB instances do not meaningfully differentiate the solvers based on best objective value. Dirac-3S and PGD match the published best-known objectives to at least five significant figures on all three instances, while Hexaly achieves comparable performance on two of the three problems. The only notable deviation is on QPLIB_2761, where Hexaly returns 0.00105576 compared to the best-known value of 0.00104853, corresponding to a relative gap of approximately 0.7%. While the best-of-run objective values suggest near-equivalent performance, the sample distributions reveal a more meaningful distinction in solution quality and reliability.

Figure 1 shows that for QPLIB_0018, Dirac-3S places 96 of its 100 samples at the optimal objective value, whereas PGD reaches the optimum in only 56 samples and distributes the remaining solutions across three higher-energy local minima. Although both solvers ultimately report the same best objective, Dirac-3S exhibits substantially stronger concentration around the optimum. This observation parallels the behavior previously seen in Section 4.2. For problem classes with highly multi-modal objective landscapes, the distribution of returned solutions can be more informative than the best observed objective alone. A solver that consistently concentrates probability mass on optimal or near-optimal solutions is more likely to deliver high-quality results with fewer samples and lower computational effort than a solver whose solutions are dispersed across multiple local minima. Consequently, best-of-batch objective tables may understate practical performance differences between competing approaches.

5.2 Planted instances

On solution quality the planted family separates the three solvers cleanly, and it does so in the same direction at both dynamic ranges. At $ub = 10$ (Table 5) Dirac-3S reaches the planted optimum on all the instances, against five of eight for PGD at 100 restarts and one of eight for PGD at 10 restarts; Hexaly reaches it on none, though it lands within 2×10^{-4} percent on one. At $ub = 100$ (Table 6) Dirac-3S reaches it on six of eight, against three of eight for PGD-100 and none for PGD-10; Hexaly again reaches it on none, but lands within 0.04 percent on four of the eight. Two features of this are worth drawing out. The first is that the ranking does not depend on which contrast level is used. The second is that a ten-sample Dirac-3S batch is more reliable than a hundred-restart PGD run on both tables, which is the comparison that matters if what one wants is a single good answer rather than a characterisation of the landscape.

Failures in this family are structural rather than marginal. When a solver misses the planted set it does not return a slightly worse point on the same support; it collapses onto a much smaller one. Hexaly's seven failures at $ub = 10$ carry supports of 19, 22, 25, 31, 33, 36 and 59 against planted sizes between 55 and 97, and its four failures at $ub = 100$ carry supports of 15, 19, 19 and 83 against planted sizes between 55 and 101. The same holds for PGD, whose failing runs sit at supports of 13 to 20 throughout, and for the three Dirac-3S failures, at supports 16, 13 and 14 against planted sizes 55, 43 and 55. Every solver that does find the planted optimum recovers a support of exactly k . The

gap is therefore close to binary: either the planted set is located, in which case the residual gap is numerical, or it is not, in which case the gap is typically 6 to 15 percent, with two outliers at 20.9 and 66.7 percent. Figure 2 shows what this looks like in time for Hexaly on `n5000_k79` at $ub = 100$: the objective falls to roughly 13324 by $t \approx 281$ seconds and then does not move for the remaining 600 seconds of the run, which is the 11.03 percent gap recorded in Table 6. The stagnation criterion is measuring a genuine plateau, not an impatient stopping rule.

The time columns support a narrower claim than the quality columns, and the two contrast levels point in opposite directions. Comparing at matched sample counts – the Dirac-3S 10-sample column against PGD-10 – PGD is between 3 and 63 times faster at $ub = 10$, but solves one instance to optimality where Dirac-3S solves seven. At $ub = 10$, then, Dirac-3S buys quality at a time premium rather than winning on time. At $ub = 100$ the picture inverts with size: PGD-10 is faster up to $n = 5000$, but from $n = 6000$ upward Dirac-3S is faster *and* better, by factors of 1.4, 3.3, 1.5, 7.1 and 6.4 at $n = 6000$ through 9980, while solving six of eight against none. The underlying reason is visible in how the two scale. Dirac-3S's 10-sample time at $ub = 100$ stays between 376 and 1015 seconds across a fivefold range of n , with no trend, whereas PGD-10 grows from 8 to 5418 seconds and PGD-100 from 1447 to 52750 seconds over the same range. Each PGD iteration costs $\Theta(n^2)$ through the gradient's matrix-vector product, and the iteration count needed to resolve the planted support does not fall as n grows.

Figure 4 shows that what degrades on the harder instances is not the device's best answer but the probability of obtaining it. All six panels are instances that Table 6 records as solved to optimality, yet the number of the ten samples that actually land at the planted optimum ranges from essentially all ten on `n5000_k79`, `n7000_k95` and `n8000_k100` down to roughly one on `n6000_k80` and `n9000_k101`, with the remaining samples an order of magnitude above the optimum. PGD's distribution is wider still and only a small fraction of its restarts come near the optimum. A best-of-batch table therefore reports the same value, 0.0, for a per-sample success probability of 1.0 and for one of 0.1, and the second of those would not survive a batch of two. This is the quantity to watch as instances get harder, and it is the reason the distribution figure carries information the tables cannot.

5.3 Maximum clique

On 40 of the 80 graphs in Table 7 all three solvers return the same clique size, and on 24 of those the common value equals the best known one. Half of the DIMACS suite therefore carries no information about the relative merits of these three solvers, which is worth stating plainly because aggregate means over the whole suite are correspondingly compressed: the mean ratios to best known in Figure 6 differ by only 0.015 across solvers, a difference dominated by the graphs on which they tie.

Where the suite does separate them, three patterns appear. First, all three solvers fail together on the same families. The mean ratio to the best known clique number is 0.65, 0.68 and 0.64 for Dirac-3S, Hexaly and PGD on the `san` family, 0.53 for all three on `san1000`, and 0.71, 0.65 and 0.67 on `keller`. These are graphs built to defeat exactly this kind of search: the `san` instances hide a large clique inside a structure engineered so that greedy and local methods are drawn elsewhere, and

the Motzkin-Straus formulation inherits the difficulty in the form of spurious local maxima supported on non-cliques [15]. A continuous relaxation solved to local optimality has no mechanism for escaping these, and the greedy repair step, which rounds a continuous point to a genuine clique, can only make the best of whatever basin the solver reached.

Second, the graphs on which all three succeed are the ones with strong local structure: `c-fat` and `johnson` at a mean ratio of 1.000, `hamming` at 0.97-0.98, `sanr` at 0.96-0.99, and `p_hat` at 0.96-0.98. On these families the continuous formulation is well behaved and the choice of solver does not matter much.

Third, Dirac-3S's advantage is concentrated on large, dense, unstructured graphs. It is the strict best of the three on five of the six `C` instances with $n \geq 250$ – `C250.9`, `C500.9`, `C1000.9`, `C2000.5` and `C2000.9`, the exception being `C4000.5`, where it ties PGD at 15 – giving a family mean of 0.929 against 0.880 for Hexaly and 0.905 for PGD. It is also strictly best on `keller5`, `hamming10-4`, `gen200-p0.9-55`, `gen400-p0.9-65`, `p_hat1000-1`, `p_hat1000-2` and `p_hat1500-1`. Splitting the suite by size makes the same point: on the 17 graphs with $n \geq 1000$ the mean ratios are 0.900, 0.862 and 0.881, a spread of 0.038, whereas on the 63 smaller graphs they are 0.887, 0.879 and 0.877, a spread of 0.010. Whatever advantage the device has grows with n and is close to invisible on small instances. The counterexample is `keller6`, the largest graph in the suite at $n = 3361$, where PGD returns 36 against Dirac-3S's 35; the MANN family similarly favours the baselines, at mean ratios of 0.995 for PGD and 0.986 for Hexaly against 0.967 for Dirac-3S.

It should be kept in mind throughout that these margins are one or two vertices on graphs whose best known cliques number in the tens, and that none of the three solvers is competitive with a dedicated combinatorial maximum-clique algorithm. The comparison here is between three general-purpose continuous solvers applied to the same relaxation, not between clique algorithms.

6 Conclusion

We have compared Dirac-3S against Hexaly and multi-start projected gradient descent on non-convex quadratic programs over a scaled simplex, using three benchmark families chosen such that solution quality can be assessed against an external reference: a planted family with known optima, DIMACS maximum-clique graphs with published clique numbers, and QPLIB instances with published best-known objectives. On the planted family, Dirac-3S is the most reliable of the three solvers at both dynamic ranges, reaching the planted optimum on seven of eight instances at $ub = 10$ and six of eight at $ub = 100$, compared to five and three instances for PGD with 100 restarts, and none for Hexaly. In this benchmark, failures are structural rather than marginal: solvers either recover the planted support exactly, yielding only numerical residual error, or converge to significantly smaller supports and incur objective gaps typically ranging from 6–15%. On the DIMACS maximum-clique suite, the three solvers return identical clique sizes on 40 of the 80 graphs, indicating that half of the benchmark does not meaningfully separate them. Across the remaining instances, however, Dirac-3S, taken as the best of its four relaxation schedules, is strictly best on 17 graphs, compared to 5 for PGD and 1 for Hexaly. This advantage is concentrated on large,

dense, and unstructured graphs, where the difficulty of the optimization problem increases with problem size. All three approaches perform poorly on the `san` and `celler` families, suggesting limitations of the underlying relaxation formulation rather than differences in solver quality. For the QPLIB instances, all three solvers match the published best-known objectives to at least five significant figures. The primary distinction is not the best objective value obtained but the consistency with which high-quality solutions are produced. For QPLIB_0018, Dirac-3S places 96 of 100 samples at the optimum, while PGD reaches the optimum in only 56 samples and distributes the remaining solutions across several higher-energy local minima. This reinforces a theme seen throughout the benchmark: solution distributions contain information that is not captured by best-of-run objective values alone. Two qualifications bound these conclusions. First, the maximum-clique advantage is a best-of-four-relaxation-schedules advantage. When restricted to a single fixed schedule, the three solvers exhibit nearly identical aggregate performance. The benchmark therefore indicates that the relaxation schedule serves as an effective tuning axis whose optimal value depends on the graph instance. Second, runtime comparisons depend strongly on problem characteristics. At $ub = 10$, PGD is 3–63 times faster than Dirac-3S while solving fewer planted instances to optimality, implying a quality-versus-time tradeoff. At $ub = 100$ and larger problem sizes, Dirac-3S becomes both faster and more accurate, outperforming PGD by factors of 1.4–7.1 while solving six of eight instances to optimality compared to none for PGD. The underlying scaling trend is particularly notable: Dirac-3S's ten-sample runtime remains approximately constant across a fivefold increase in problem size, whereas PGD's hundred-restart runtime increases by more than an order of magnitude over the same range. Taken together, these results suggest that Dirac-3S is most differentiated not on small or easily solved instances, where many approaches converge to comparable solutions, but on larger and more challenging optimization problems where robustness, solution quality, and scaling behavior become increasingly important. Across all benchmark families considered, Dirac-3S consistently delivers competitive or superior solution quality while demonstrating favorable scaling characteristics as problem complexity grows.

7 Outlook

It is important to emphasize the scope of this benchmarking study. The objective is not to demonstrate superiority over every specialized algorithm developed for a particular problem class. Many NP-hard optimization problems admit highly tailored methods that can outperform general-purpose solvers when substantial effort is invested in problem-specific modeling, parameter tuning, and algorithm design. Instead, the purpose of this study is to evaluate each approach as a general NP-hard optimization solver under a common framework. This perspective reflects a practical reality faced by many organizations: real-world optimization portfolios often contain diverse problem types, evolving requirements, and limited opportunities to develop and maintain specialized solution methods for each application. In such settings, a solver that performs well across a broad range of problem classes can provide significant value by reducing development effort, simplifying deployment, and enabling a more consistent optimization workflow. Accordingly, the results presented here should not be interpreted as a universal ranking of optimization technologies, but rather as a characterization of the capabilities of Dirac-3S as a general-purpose optimization platform. The benchmarks demonstrate that Dirac-3S can serve as a

competitive alternative or complementary approach within existing optimization pipelines, particularly for challenging large-scale problems where robustness, scalability, and solution concentration are important considerations. QCi will continue expanding this benchmarking effort to additional problem classes, benchmark suites, and application domains. Future studies will provide a broader understanding of the strengths and limitations of Dirac-3S across diverse optimization workloads and help users identify opportunities to incorporate Dirac-3S into existing solver portfolios to improve solution quality, computational efficiency, or both.

A Instance formats

A.1 Planted StQP (.npz)

A one-dimensional `float32` array of length $n(n+1)/2$ holding the upper triangle of Q in row-major order including the diagonal. Reconstruction:

```
def loadQ(path, n):
    tri = np.load(path, mmap_mode="r")
    Q = np.empty((n, n), dtype=tri.dtype)
    iu = np.triu_indices(n)
    Q[iu] = tri
    Q.T[iu] = tri
    return Q
```

Filenames encode the parameters: STQP_n_{n}_k_{k}_R_{R}_seed_{seed}_ub_{u}.npz.

B Related Technical References

The following documents provide additional technical details that complement this User Guide:

- **Dirac-3S User Guide** Comprehensive guide to understand Dirac-3S computation, timing, how to install the hardware, and how to use the hardware.
- **Dirac-3S Continuous Solver Benchmarking** Benchmark examples, test methodologies, performance metrics, and results.

C Contacting Support

For assistance, please contact:

Email: support@quantumcomputinginc.com

Support Portal: <https://quantumcomputinginc.ladesk.com/>
 When contacting support, provide:

- Dirac-3S serial number
- Software version
- Problem description
- Error messages
- Relevant log files

References

- [1] Amir Beck and Dror Pan. "A branch and bound algorithm for nonconvex quadratic optimization with ball and linear constraints". In: 69.2 (Oct. 2017), pp. 309–342. ISSN: 0925-5001. DOI: [10.1007/s10898-017-0521-1](https://doi.org/10.1007/s10898-017-0521-1).
- [2] Guglielmo Beretta, Alessandro Torcinovich, and Marcello Pelillo. "On generalized KKT points for the Motzkin–Straus program". In: *Journal of Global Optimization* 91.3 (Dec. 2024), pp. 535–557. ISSN: 1573-2916. DOI: [10.1007/s10898-024-01457-2](https://doi.org/10.1007/s10898-024-01457-2). URL: <http://dx.doi.org/10.1007/s10898-024-01457-2>.
- [3] Immanuel Bomze. "On Standard Quadratic Optimization Problems". In: *Journal of Global Optimization* 4 (1998), pp. 369–87.
- [4] Immanuel M. Bomze. "Standard quadratic optimization problems: applicationsStandard Quadratic Optimization Problems: Applications". In: *Encyclopedia of Optimization*. Ed. by Christodoulos A. Floudas and Panos M. Pardalos. Boston, MA: Springer US, 2001, pp. 2442–2444. ISBN: 978-0-306-48332-5. DOI: [10.1007/0-306-48332-7_484](https://doi.org/10.1007/0-306-48332-7_484). URL: https://doi.org/10.1007/0-306-48332-7_484.
- [5] John Duchi et al. "Efficient projections onto the ℓ_1 -ball for learning in high dimensions". In: ICML '08. Helsinki, Finland: Association for Computing Machinery, 2008, pp. 272–279. ISBN: 9781605582054. DOI: [10.1145/1390156.1390191](https://doi.org/10.1145/1390156.1390191).
- [6] Fabio Furini et al. "QPLIB: A Library of Quadratic Programming Instances". In: *Mathematical Programming Computation* (2018). DOI: [10.1007/s12532-018-0147-4](https://doi.org/10.1007/s12532-018-0147-4). URL: <https://doi.org/10.1007/s12532-018-0147-4>.
- [7] He Hao, Daniel Silvestre, and Carlos Silvestre. "Continuous trajectory planning for non-convex utility functions using hybrid optimization". In: *European Journal of Control* 87 (2026), p. 101425. ISSN: 0947-3580. DOI: <https://doi.org/10.1016/j.ejcon.2025.101425>.
- [8] M Held, P Wolfe, and H P Crowder. "Validation of subgradient optimization". In: *Mathematical Programming* (1974), pp. 62–88.

- [9] Hexaly. *Hexaly Optimizer*. Version 15.0. 2026. URL: <https://www.hexaly.com>.
- [10] David J. Johnson and Michael A. Trick, eds. *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge Workshop, October 11-13, 1993*. Vol. 26. DIMACS Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 1996. ISBN: 978-0-8218-6609-2.
- [11] E. de Klerk and D.V. Pasechnik. "A linear programming reformulation of the standard quadratic optimization problem". In: *J Glob Optim* (2007), pp. 75–84. DOI: <https://doi.org/10.1007/s10898-006-9037-9>.
- [12] Asimina Marousi and Vassilis M. Charitopoulos. *Global and Robust Optimization for Non-Convex Quadratic Programs*. 2025. arXiv: 2503.07310 [math.OC]. URL: <https://arxiv.org/abs/2503.07310>.
- [13] TS Motzkin and EG Straus. "Maxima for Graphs and a New Proof of a Theorem of Turán". In: *Canadian Journal of Mathematics* (1965), pp. 533–540. DOI: [10.4153/CJM-1965-053-6](https://doi.org/10.4153/CJM-1965-053-6).
- [14] Lac Nguyen et al. *Entropy Computing: Entropy computing, a paradigm for optimization in open photonic systems*. 2025. URL: <https://doi.org/10.1038/s42005-025-02324-6>.
- [15] M Pelillo and A Jagota. "Feasible and infeasible maxima in a quadratic program for maximum clique". In: *Journal of Artificial Neural Networks* (1995), pp. 411–420.