

A Width-Range Boundary Atlas for Fixed-Point Quantum LDPC Min-Sum Decoding

Hung Ngoc Dang

hungdn@ptit.edu.vn

Posts and Telecommunications Institute of Technology

Research Article

Keywords: quantum error correction, quantum LDPC codes, min-sum decoding, fixed-point arithmetic, quantisation, non-inferiority

Posted Date: August 13th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10671682/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

A Width–Range Boundary Atlas for Fixed-Point Quantum LDPC Min-Sum Decoding

Hung Ngoc Dang^{1*}

^{1*}Faculty of Information Technology 1, Posts and Telecommunications
Institute of Technology, Km 10, Nguyen Trai Street, Ha Dong, Hanoi,
Vietnam.

Corresponding author(s). E-mail(s): hungdn@ptit.edu.vn;

Abstract

Fixed-point quantum low-density parity-check decoders are often described by word width, although width does not determine dynamic range. We report a pre-registered descriptive study of 19 width–range formats across 48 cells covering four codes, both Calderbank–Shor–Steane sectors, three physical error rates and two normalisation recipes. No tested format was jointly resource-qualified and non-inferior in every cell; eight-bit behaviour changed with clip level and normalisation. Reproducible finite-precision reports must therefore include width, the normalisation recipe and the clip level. Evidence is limited to the tested code-capacity bank, with many unresolved cells and no held-out confirmation.

Keywords: quantum error correction, quantum LDPC codes, min-sum decoding, fixed-point arithmetic, quantisation, non-inferiority

1 Introduction

Practical quantum low-density parity-check (qLDPC) decoders must run in finite precision, and the reduced-precision literature commonly describes a decoder as “ w -bit”.

047 That shorthand implicitly assumes that the number of bits determines whether the
048 format works.
049

050 A fixed-point format is specified by two quantities, not one: a clip level L that
051 bounds representable magnitude, and a step $\Delta = L/2^{w-1}$ that bounds resolution.
052 At fixed width these trade directly against each other. Widening the range reduces
053 clipping and coarsens resolution; narrowing it does the reverse. Which side of that
054 trade a decoder lands on depends on where its message-magnitude distribution sits
055 relative to L . For normalised min-sum, the normalisation gain α multiplies exactly
056 that distribution against a fixed grid, so gain and range act on the same operating
057 point.
058

059 This paper asks whether the LER classifications and traffic savings of a fixed
060 width-range format remain consistent across codes, Calderbank–Shor–Steane (CSS)
061 sectors, physical error rates and normalisation recipes. We hold the decoder kernel,
062 budgets and statistical rule fixed, and vary width and range factorially across that
063 frozen bank. The study was prospectively staged: the design, thresholds and stopping
064 rule for each stage were fixed before the evidence for that stage was generated.
065

066 Our contributions are:

- 067 1. A preregistered factorial separation of word width and dynamic range for qLDPC
068 min-sum decoding, evaluated at a code-capacity logical endpoint rather than a
069 proxy metric, over both CSS sectors and two normalisation recipes.
070
- 071 2. A 48-cell $\times$ 19-format boundary atlas over the tested grid, reported without omit-
072 ting unresolved or unfavourable cells. It shows that no format in the frozen bank is
073 jointly resource-qualified and non-inferior everywhere, and separates format failures
074 from limits of statistical power.
075
- 076 3. Direct evidence, independent of the resource condition, that width is not a sufficient
077 specification: holding the width fixed at eight bits, clip level 32 is non-inferior in
078 16 of 48 cells while clip level 2 is inferior in all 48.
079
080
081
082
083
084
085
086
087
088
089
090
091
092

4. A transparent account of the study’s evidential limits: a preregistered held-out confirmation was declined before execution, and a preregistered mechanism figure could not be produced.

2 Background and related work

Quantised message passing is well studied for classical LDPC codes [1]. Information-optimum and mutual-information-maximising designs construct quantiser maps that preserve information at very low width [2, 3], and the reconstruction–computation–quantisation approach demonstrates low-bit-width decoding with hardware evidence [4]. Saturation and quantisation are known drivers of classical error-floor behaviour [5, 6], noisy and finite-precision min-sum has been analysed by density evolution [7], and code-aware quantiser design is an established idea [8]. Field-programmable gate array (FPGA) implementations of LDPC decoders are surveyed in [9]. Those established mechanisms are not the focus here.

A related but distinct application of quantisation is at the receiver front end, where low-resolution converters coarsen the observations before the decoder sees them. Protograph LDPC codes have been designed for one-bit and mixed-resolution large-scale multiple-input multiple-output (MIMO) systems under exactly that constraint, jointly with the detector [10, 11]. That work quantises the decoder’s *input*; the present paper leaves the input in floating point and quantises the messages exchanged *inside* the decoder, so the two are complementary rather than overlapping.

In the quantum setting, syndrome-based min-sum and ordered-statistics-decoding (OSD) architectures have been implemented and analysed on FPGAs [12]. Soft-syndrome architectures have been proposed at 6- and 8-bit precision [13]. The design space for belief-propagation and OSD implementations on FPGAs and application-specific integrated circuits (ASICs) has also been explored [14]. Hardware emulation has been used to evaluate several finite-precision formats and to build a diversity

decoder from multiple quantisers, reaching very low logical error rates [15]. Circuit-level hardware evidence at 12-bit normalised min-sum has been reported for correlated errors [16]. Decoder behaviour across a range of qLDPC codes [17], the finite-length codes used throughout this literature [18], and quantum trapping-set mechanisms [19] provide the population and mechanism context.

Prior work has therefore established that quantisation can help qLDPC belief propagation (BP), that different quantisers can correct different error patterns, and that multi-precision diversity can be useful [15]. The narrower question addressed here is a controlled *factorial separation* of width and range on one fixed decoder, evaluated at a logical endpoint in both CSS sectors under matched normalisation, with the resource accounting stated in its own units.

3 Methods

The study ran in three prospectively staged phases. The design, thresholds and stopping rule for each phase were fixed before its evidence was generated. The endpoint throughout is logical error rate (LER). Fig. 1 sets out the order. The second normalisation recipe was added after the first-stage result exposed a confound and before the atlas was generated. The held-out confirmation was declined at its preregistered preflight point, before execution, using the projections shown in the figure.

3.1 Codes and channel

The bank comprises four public codes, frozen in the sense used throughout this paper: fixed before any of the evidence reported here was generated. The hypergraph-product (HGP) code [20] is built from a $(3, 4)$ -regular $[20, 5]$ classical code with a 15×20 parity-check matrix generated at seed 7, taken as both factors, giving $[[625, 25]]$. The three bivariate-bicycle (BB) codes follow the construction of [21]. They use $A = x^{a_1} + y^{a_2} + y^{a_3}$ and $B = y^{b_1} + x^{b_2} + x^{b_3}$ over $\ell \times m$ circulants. The parameter triples

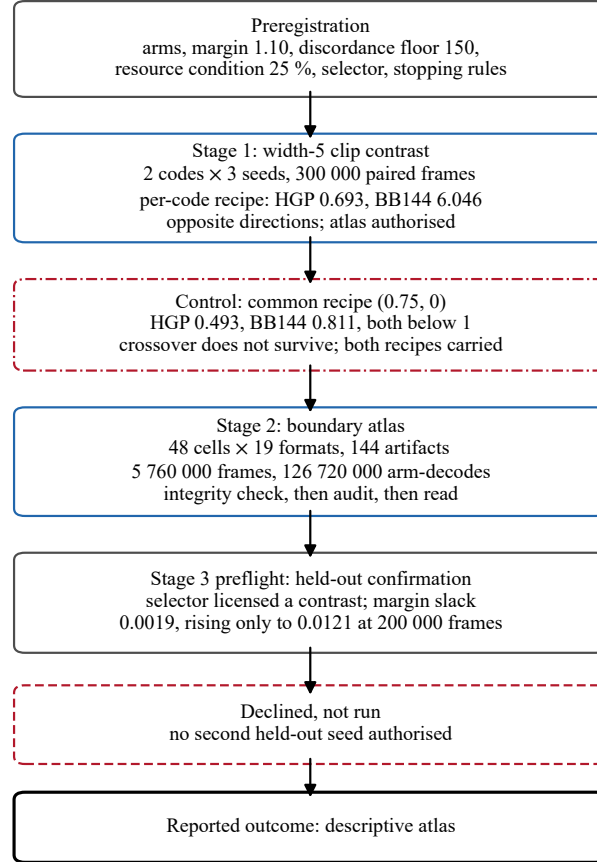


Fig. 1 Study design. Solid boxes mark executed stages, the dash-dot box marks the control, and the dashed box marks a preregistered branch that was not run. Ratios are paired LER against a matched floating-point reference.

$((\ell, m); (a_1, a_2, a_3); (b_1, b_2, b_3))$ are $((6, 6); (3, 1, 2); (3, 1, 2))$, $((15, 3); (9, 1, 2); (0, 2, 7))$ and $((12, 6); (3, 1, 2); (3, 1, 2))$. They give $[[72, 12]]$, $[[90, 8]]$ and $[[144, 12]]$, respectively.

Each is evaluated in both sectors of the CSS construction [22, 23] (X and Z) at three physical error rates, $p \in \{0.015, 0.020, 0.025\}$, under a depolarising code-capacity model. The prior handed to the decoder is the true component marginal $2p/3$, so the channel log-likelihood-ratio (LLR) magnitude is $\lambda = \ln((1 - 2p/3)/(2p/3)) = 4.60, 4.30$ and 4.08 at the three rates. This matters for reading the clip level: λ is the natural unit of the range axis, and clip levels 2 and 4 lie *below* the channel input at every

Table 1 Reference error-rate range. The floating-point logical error rate is shown by code over both sectors, all three error rates and both recipes.

code	min reference LER	max reference LER
HGP	3.94×10^{-3}	2.19×10^{-2}
BB72	2.00×10^{-3}	8.73×10^{-3}
BB90	6.00×10^{-4}	3.63×10^{-3}
BB144	7.17×10^{-4}	4.15×10^{-3}

rate. The resulting $4 \times 2 \times 3 = 24$ populations are evaluated under two normalisation recipes, giving 48 frozen cells.

Reference logical error rates span 6.0×10^{-4} to 2.2×10^{-2} (Table 1). This is a measurable-difference regime, well above the 10^{-4} band in which [15] reports precision behaviour changing; Section 5.3 bounds the claims accordingly.

The endpoint is a *single-sector* logical error rate under a component code-capacity model. It is neither a full CSS logical error rate nor a circuit-level one.

3.2 Decoder and formats

The decoder is min-sum [24] in its normalised-offset form [25], with at most 32 iterations. Within any given contrast every decoder shares the same (α, β) , so that precision is the only quantity varying. Two normalisation recipes are used: a per-code recipe fitted in earlier work, and a single common recipe $(\alpha, \beta) = (0.75, 0)$ applied to all four codes. The second exists because a per-code recipe confounds code identity with normalisation; the common recipe removes that particular confound.

Note that the two recipes differ in *both* parameters: the per-code recipe carries $\beta \in \{0.1, 0.2\}$ and the common recipe carries $\beta = 0$ (Table 12). Every contrast between them is therefore a contrast of recipes, not of the gain alone.

Formats are indexed as `ww_rr` for word width $w \in \{5, 6, 7, 8\}$ and clip level $r \in \{2, 4, 8, 16, 32\}$. A format is a signed two's-complement pair (w, f) in which the stored integer q represents $q/2^f$, so that the clip level and the fractional width are tied by

$f = w - 1 - \log_2 r$. One combination, `w5_r32`, requires $f = -1$ and is therefore not representable: the widest range a signed w -bit word can carry is 2^{w-1} , so five bits cannot span ± 32 at all. That leaves 19 feasible formats. Each cell additionally carries a matched floating-point reference and two plain-min-sum controls, the latter used only as normalisation diagnostics.

To make the format definition independent of language defaults, the kernel uses round-half-away-from-zero, saturating rather than wrapping overflow, and an explicitly represented scale. These choices define the formats evaluated here; the study does not compare alternative rounding or overflow rules.

3.3 Endpoint, resource accounting and decision rule

Logical error rate is compared to the matched floating-point reference on identical frame populations. Every decoder sees the same frames as the reference, so each frame yields a paired outcome. The resulting comparison is a 2×2 table of concordant and discordant decisions, not two independent rate estimates. Only the discordant cells carry information about the difference, which is why a discordance floor appears in the decision rule below. Intervals on the paired ratio come from 20 000 bootstrap resamples, and pooling across seeds is by addition of the sufficient statistics of those tables, never by averaging ratios.

Intervals are two-sided 95 % percentile bootstrap bounds, resampled from the multinomial over the four cells of the paired table. A cell is first required to carry at least 150 discordant pairs; this floor gates *both* verdicts, so a cell below it is unresolved whichever way its point estimate leans and cannot be declared inferior any more than it can be declared non-inferior. Above the floor, a format is *non-inferior* when the upper bound is at most 1.10, *inferior* when the lower bound exceeds 1.10, and *unresolved* otherwise. A point estimate below the margin with an interval crossing it is unresolved, never a pass.

323 The floor is a proxy for power rather than a measure of it. The width of an interval
 324 on a ratio of marginal rates depends on the concordant failure mass as well as on
 325 the discordant count. The floor is therefore necessary, but not sufficient, for a cell to
 326 be informative. It is applied symmetrically, and it is not free on either side. Before
 327 the floor is imposed, 327 cells have an upper bound at or below 1.10 and 482 have
 328 a lower bound above it. The floor demotes 236 of the former and 81 of the latter
 329 to unresolved, leaving the 91 non-inferior and 401 inferior cells reported here. The
 330 demoted inferior cells carry between 26 and 146 discordant pairs. Section 4.9 reports
 331 which of the two causes of “unresolved” dominates, because the answer changes what
 332 the atlas’s negative result means.

339 340 **3.3.1 Multiplicity**

342 The atlas contains $19 \times 48 = 912$ interval statements, and we apply no multiplicity
 343 correction to them. Each bound is a per-comparison 95 % bound, so under a boundary
 344 null in which the true ratio equals the margin, of order 23 of the 912 cells would be
 345 declared non-inferior by chance. Individual cell verdicts, and small differences between
 346 per-format counts (17 against 16 against 12), are therefore uncorrected and not sep-
 347 arable by this design. Two conclusions are unaffected. The principal negative result
 348 is an intersection–union statement — no format qualified in *every* cell — and multi-
 349 plicity makes such a statement harder to satisfy, not easier, so it stands uncorrected.
 350 The cells are also not independent replicates: within a cell all 19 formats share one
 351 reference realisation on the same frames, and the 48 cells are 4 codes $\times$ 2 sectors $\times$ 3
 352 rates $\times$ 2 recipes with an effective independent-code count of four. We therefore make
 353 no inferential use of the counts beyond ranking.

354 One property of the baseline should be stated before it is used. The `w8_r8` for-
 355 mat against which all traffic is measured is itself non-inferior in zero of the 48 cells
 356 (Table 10) and unresolved in 32 (Table 9), leaving 16 cells in which it is *inferior*.
 357
 358

It is an architectural reference point, chosen as the widest conventional format at a mid-range clip level, and it is not a quality reference.

Traffic is counted as the total fixed-point message bits moved per frame, $T = w \times E \times \bar{t}$ for word width w , edge count E and mean executed iterations $\bar{t}$, and is reported as the saving relative to `w8_r8`. Because $\bar{t}$ is an outcome rather than a constant, the resource axis is not independent of decoding quality; Section 5.3 returns to this. At equal iteration count the saving against an eight-bit baseline is bounded by $1 - w/8$ — 37.5 % at $w = 5$ down to 0 % at $w = 8$ — so a wide format can only clear a demanding resource condition by converging faster. The largest saving observed in any single cell exceeds those bounds: 38.1 %, 27.2 %, 16.0 % and 5.8 % at $w = 5, 6, 7, 8$. That excess is the iteration effect. It also lifts the mean of `w8_r16` to 1.1 %, above the width-only bound of zero; the other per-format means remain below their width-only bounds. The effect is not large enough to lift $w = 7$ or $w = 8$ over 25 %.

Resource use is reported as two quantities that are never combined. The first is architectural message traffic in bits, relative to the `w8_r8` fixed-point baseline; the *joint target* requires at least 25 % traffic saving *and* non-inferiority in the same cell. The second is fixed shift/add work per message, reported separately in Table 12. Bits moved and logic operations are distinct units and are not combined.

3.4 Preregistration and scale

Decoder configurations, thresholds, the discordance floor, the non-inferiority margin, the selection policy for a held-out confirmation and the terminal stopping rules were all registered before the corresponding evidence existed. The atlas comprises 144 result sets of 40 000 frames each: 5 760 000 population frames and 126 720 000 decodes at 22 decoder configurations per frame (the 19 feasible formats, the matched reference and the two controls), pooled by addition of sufficient statistics over three seeds per cell. No random-effects model or across-code independence is assumed or claimed.

Table 2 Evidence integrity. The panel records the checks applied to the frozen atlas.

quantity	value
interpreter	CPython 3.11.9
raw artifacts / empty stderr	144 / 144
pooled cells	48
normalisation recipes	2
seeds per cell	3
frames per artifact	40,000
distinct artifact digests	144
snapshot test suite	250 passed
audit status	complete
held-out confirmation	licensed, declined (not run)

3.5 Use of AI-assisted tools

During manuscript preparation, the author used Claude (Anthropic) and Codex (OpenAI) for LaTeX format conversion, language editing and consistency checking. No model output was used as experimental data or as a source of scientific evidence. The author checked the resulting manuscript against the preregistration, source code and frozen result artefacts and accepts full responsibility for its content.

4 Results

4.1 Evidence integrity

Table 2 summarises the conditions under which the atlas was produced. All 144 result sets come from one frozen revision executed on one machine. Each set was checked for a clean run, verified against a recorded digest and found distinct. The accompanying test suite was rerun against the same revision before any result was read. Every result below is reported conditional on those checks.

4.2 Preliminary width-5 clip-level contrast

The atlas was preceded by a narrower preregistered contrast. It moved one five-bit format from clip level 8 to clip level 16 on two codes at $p = 0.020$ in the Z sector.

Table 3 Width-5 clip-level contrast. Each code contributes 300 000 paired frames; a ratio below one means the wider clip level gave the lower LER.

profile	code	ratio	95 % CI	disc.
per-code	HGP	0.693	[0.672, 0.714]	3202
	BB144	6.046	[5.609, 6.546]	3456
common	HGP	0.493	[0.479, 0.508]	4613
	BB144	0.811	[0.781, 0.842]	182

Each code contributed 300 000 paired frames over three fresh seeds. Table 3 reports the contrast.

Under the per-code recipe the two codes moved in *opposite* directions. Widening the range lowered the HGP logical error rate (paired ratio 0.69) and raised that of BB144 (paired ratio 6.05); both intervals lie far from unity. The direction was the same at each of the three fixed seeds. In isolation, this pattern suggests an apparent code-dependent crossover.

Under the common recipe, where every code decodes at $(\alpha, \beta) = (0.75, 0)$, the ratios are instead below unity for both codes: 0.49 for HGP and 0.81 for BB144, with both intervals wholly below unity. Thus the code-dependent crossover seen under the per-code settings is not present when both codes use the same recipe. In the per-code contrast, code identity is confounded with normalisation because BB144 uses $\alpha = 1.0$ whereas HGP uses $\alpha = 0.875$.

Two cautions apply to this contrast. The BB144 common-recipe reversal is estimated from 182 pooled discordant pairs, about 19 times fewer than the 3456 pairs in its per-code contrast; its per-seed counts of 54, 63 and 65 are all below the 150-pair floor. With one code per family, the reversal cannot be attributed to a code *family*; it is a difference between these two codes at these two gains.

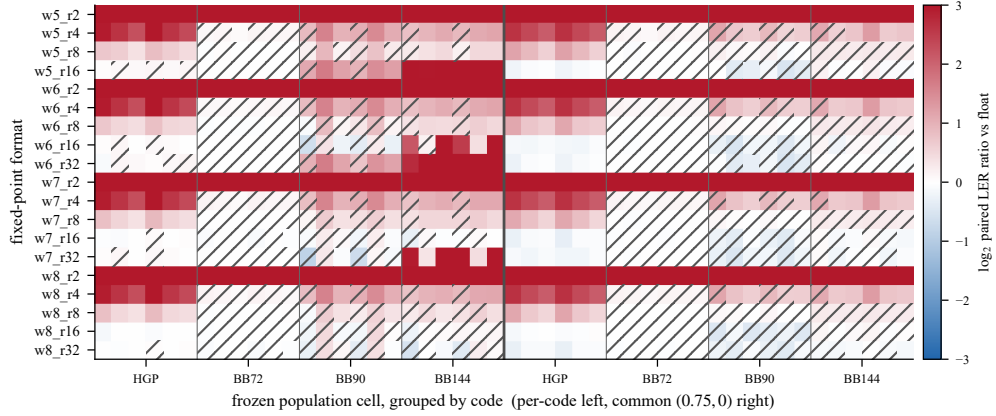


Fig. 2 Width-range boundary atlas. Paired logical-error-rate ratios are shown for all 19 feasible formats across the 48 frozen cells. Hatching marks cells whose interval leaves the outcome unresolved. The heavy vertical rule separates the two normalisation recipes; lighter rules separate codes.

4.3 The boundary atlas

Fig. 2 shows every feasible format in every frozen cell, with no cell omitted. Two features are visible without any statistics. Clip level 2 is inferior in all 48 cells at every width. The favourable region is confined to the wider clip levels, and is interrupted by large unresolved areas.

4.4 No format is non-inferior in all cells

Across the frozen bank the set of formats meeting the joint target in *every* cell is empty, and it is empty under both normalisation recipes separately. Of the 912 format-cell combinations, 91 are non-inferior, 401 are inferior, and 420 are unresolved.

Table 4 and Fig. 3 give the distribution by width and range. The largest joint-target count belongs to **w5_r16**, which reaches the target in 10 of 48 cells at a mean traffic saving of 30.6 %. No eight-bit format reaches it in any cell.

4.5 Width is not a format specification

The most direct evidence for this claim needs no resource condition at all. Holding the word width fixed at eight bits and moving only the clip level, **w8_r32** is non-inferior

Table 4 Joint-target count and traffic saving. Results are grouped by word width and clip level. Traffic saving is relative to the **w8_r8** baseline; “_” marks the arithmetically impossible format.

width	dynamic range				
	2	4	8	16	32
<i>cells meeting the joint target, of 48</i>					
$w = 5$	0	2	0	10	—
$w = 6$	0	0	0	4	5
$w = 7$	0	0	0	0	0
$w = 8$	0	0	0	0	0
<i>mean traffic saving vs w8_r8 (%)</i>					
$w = 5$	-277.5	29.3	32.0	30.6	—
$w = 6$	-320.7	17.2	22.1	19.6	18.1
$w = 7$	-382.6	4.8	11.6	11.2	7.3
$w = 8$	-444.5	-7.6	0.0	1.1	-0.3

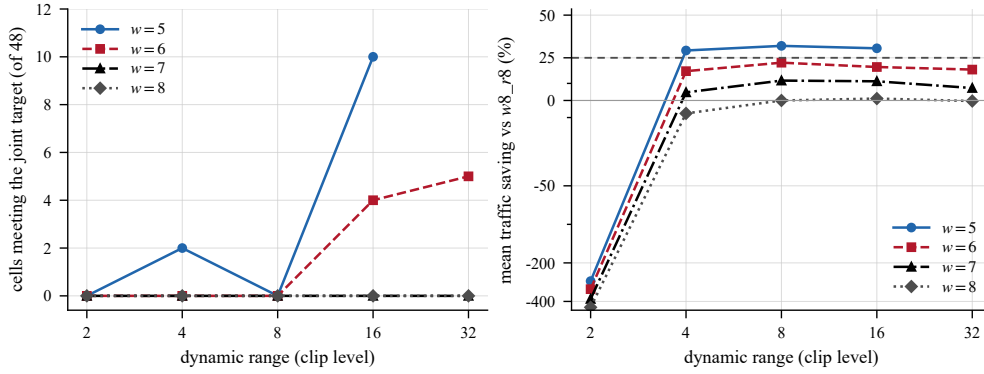


Fig. 3 Joint-target count and traffic saving. Cells meeting the joint target (left) and mean traffic saving relative to the **w8_r8** baseline (right) are shown by clip level. The right panel uses a symmetric log scale; the dashed line is the 25 % resource condition.

in 16 of 48 cells and inferior in none, while **w8_r2** is inferior in all 48. Two formats reported as “8-bit” sit at opposite ends of the atlas. The same contrast holds at five bits, where **w5_r16** is non-inferior in 10 cells and **w5_r2** in none.

The joint criterion gives a second and weaker result, reported here with its caveat. **w5_r16** meets the joint target in 10 cells and no eight-bit format meets it anywhere, but the eight-bit formats do not fail on decoding. **w8_r32** has more non-inferior verdicts

Table 5 Selected format specialists. The two formats discussed in the text and the two preregistered transfer specialists are shown over all 48 cells and within each normalisation recipe. “joint” counts cells meeting both the resource and non-inferiority conditions; “worst CI” is the largest confidence-interval (CI) upper bound on the paired ratio and “worst save” the smallest traffic saving, both over the cells in that scope.

format	scope	non-inf.	unres.	inferior	joint	worst CI	worst save
w5_r16	all 48	10	28	10	10	12.16	9.5
	per-code	3	11	10	3	12.16	9.5
	common	7	17	0	7	1.23	33.4
w6_r32	all 48	10	28	10	5	14.26	-10.7
	per-code	3	11	10	3	14.26	-10.7
	common	7	17	0	2	1.13	21.8
w5_r8	all 48	0	30	18	0	2.46	8.4
	per-code	0	12	12	0	2.29	8.4
	common	0	18	6	0	2.46	28.9
w8_r32	all 48	16	32	0	0	1.59	-3.7
	per-code	8	16	0	0	1.59	-3.7
	common	8	16	0	0	1.13	-1.1

than w5_r16, 16 against 10. It fails on resource, at a mean of -0.3% against a baseline of its own width, which as noted in Section 3.3 is close to structurally unavoidable.

This is the general pattern, not an isolated pair. Five formats are non-inferior in 12 or more of the 48 cells: w7_r16 with 17, w8_r32 with 16, and w6_r16, w7_r32 and w8_r16 with 12 each. Of those five, only w6_r16 reaches the joint target anywhere, and then in 4 cells. The other four save too little traffic to qualify against a 25% condition: w7_r16 at 11.2%, w8_r32 at -0.3% , w7_r32 at 7.3% and w8_r16 at 1.1%.

The joint-target leader therefore does not lead on non-inferiority count: w5_r16 is non-inferior in 10 cells, fewer than each of those five. Non-inferiority and traffic saving favour different regions of the format space; w5_r16 satisfies both conditions in the largest number of cells. It should be read only as the joint-target leader, not as the format with the most favourable LER verdict profile.

Table 5 reports these two formats over all 48 cells and within each normalisation recipe, together with the two formats frozen as transfer specialists before the atlas was run, w6_r32 and w5_r8.

Table 6 Inferior cells by code and recipe. Counts for `w5_r16` are shown against the per-code recipe parameters.

code	(α, β)	per-code	per-code	common
BB72	(0.8125, 0.1)	0	0	0
HGP	(0.8750, 0.2)	0	0	0
BB90	(0.9375, 0.2)	4	0	0
BB144	(1.0000, 0.2)	6	0	0

4.6 Where `w5_r16` fails, and what moves it

Counting cells hides the size of a failure, so Table 5 also reports the worst case in each scope. `w5_r16` does not behave uniformly across cells, and where it fails is informative.

`w5_r16` is inferior in 10 of 48 cells, and *every one of those ten lies in the per-code recipe*. Its worst upper confidence bound there is 12.16, so that interval permits more than an order-of-magnitude degradation relative to the floating-point reference. Under the common recipe the same format has no inferior cell at all, a worst bound of 1.23, and a worst traffic saving of 33.4%.

The ten failures are also not spread evenly across codes. They fall on the two codes with the highest per-code gains, and their count increases with that gain (Table 6) — though β is not constant down that column either, so even the ordering is not an α effect.

Holding the code, sector, error rate and format fixed and changing the normalisation recipe to (0.75, 0) leaves none of those ten cells inferior. That is a statement about verdicts, and the accounting underneath it matters: of the ten, exactly one becomes demonstrably non-inferior, and the other nine fall below the discordance floor, carrying between 22 and 144 discordant pairs.

The recipe change therefore removes the inferior verdicts but establishes non-inferiority in only one of ten cases. The other nine become unresolved because too few discordances remain, not because safety has been demonstrated. The distinction

drawn in Section 4.9 therefore bounds this claim: the per-code recipe makes this format measurably worse, and the common recipe makes it mostly unmeasurable.

The comparison isolates less than it appears to. The manipulation is within-code, since code, sector, error rate and format are all held, so it is not confounded with code identity. But it moves *both* recipe parameters at once: α falls and β goes to zero (Table 12). This study therefore cannot attribute the effect to the gain alone. The claim is that the (α, β) recipe decides usability at fixed format; separating the two parameters would need a third recipe, which the terminal stopping rule of our preregistration does not authorise.

The across-code ordering in the table is weaker still: each recipe belongs to exactly one code, so that column is confounded with code identity and rests on four points. A further caution applies: BB72 and HGP show “0 inferior” partly because most of their cells do not resolve at all (Section 4.9), not because the format is demonstrably safe on them.

4.7 The ranking depends on the resource threshold

The 25 % resource condition is preregistered, but it is one point on a curve, and the format it selects is not the same at every point. Table 7 recounts the joint target at thresholds from 10 % to 30 %.

At 10 %, 15 % and 20 % the leader is **w6_r16**, at 12, 11 and 11 cells. Only at 25 % and above does **w5_r16** lead, and it does so partly because the wider formats have been excluded by the bound of Section 3.3 rather than outperformed. At a looser resource condition it is **w6_r16**, not **w5_r16**, that leads on this bank. The preregistered threshold was not derived from any deployment requirement.

Table 7 Resource-threshold sensitivity. Cells meeting the joint target are shown as the resource condition is varied. Formats reaching zero at every threshold are omitted; the final row names the leading format at each threshold.

format	resource threshold				
	10 %	15 %	20 %	25 %	30 %
w5_r4	2	2	2	2	2
w5_r16	10	10	10	10	10
w6_r16	12	11	11	4	0
w6_r32	10	10	10	5	0
w7_r16	10	1	0	0	0
w7_r32	6	4	0	0	0
best	w6_r16	w6_r16	w6_r16	w5_r16	w5_r16

4.8 Sensitivity to the margin and the floor

The non-inferiority margin 1.10 and the 150-pair floor are preregistered conventions, not quantities derived from a deployment requirement. We therefore report how much of the result they carry. Table 8 reclassifies the same intervals over a grid of both.

The absolute counts are sensitive to these choices, whereas the qualitative ordering is consistent across the tested grid. The total non-inferior count moves from 58 to 157 of 912, and w5_r16 moves from 6 to 19 of 48, so no absolute count in this paper is independent of the chosen margin and floor. What does not move is the ordering: at every one of the nine settings the same small group of formats leads, and the principal negative result survives throughout, since no format is non-inferior in all 48 cells at any margin or floor we tried.

4.9 Why cells are unresolved

“Unresolved” has two distinct causes with different evidential implications: too few disagreements to support any verdict, or enough disagreements but an interval that straddles the margin. Table 9 separates them; the two causes contribute very unequally.

For every format examined, most unresolved cells fall below the discordance floor rather than straddling: 25 of 28 for w5_r16, 29 of 32 for w8_r32, and 32 of 32 for the

Table 8 Decision-rule sensitivity. Non-inferior cells are counted under alternative non-inferiority margins and discordance floors. † marks the preregistered setting used throughout the paper.

floor	non-inferiority margin		
	1.05	1.10	1.20
<i>non-inferior cells, of 912</i>			
100	98	129	157
150†	74	91	111
250	58	70	84
<i>non-inferior cells for w5_r16, of 48</i>			
100	13	16	19
150†	7	10	13
250	6	8	11

w8_r8 baseline. Their unresolved status therefore arises from too few discordances, not from an interval that straddles the margin. This does not establish equivalence or safety.

This bounds how the atlas’s negative result should be read. “No format is non-inferior everywhere” is in large part a statement that our frames cannot certify formats that behave nearly identically to the floating-point reference. It is not evidence that those formats are unsafe. The formats that resolve are those that differ from the reference enough to be measured, so the atlas identifies formats that fail more readily than it certifies formats that pass.

4.10 Reading the atlas along constant quantisation step

Word width and clip level are treated as separate factors in the design, but together they determine the step through $\Delta = L/2^{w-1}$. Along a constant- Δ diagonal, resolution is fixed while word width and representable range increase together. For the LER endpoint, this changes the available range without changing the quantisation step; for resource use, word width still changes. Table 10 presents the atlas in that form.

Table 9 Sources of unresolved outcomes. Cells are separated into those below the 150-pair discordance floor and those whose interval straddles the margin.

format	unresolved	below floor	straddling
w5_r16	28	25	3
w6_r32	28	25	3
w7_r16	31	29	2
w8_r16	36	34	2
w8_r32	32	29	3
w5_r8	30	28	2
w8_r8	32	32	0

Two regularities appear that the (w, r) view obscures. First, the clip level separates the atlas far more sharply than the step does. All four formats at $L = 8$ are non-inferior in zero of 48 cells, at every width and on four different diagonals. Below $L = 16$, only w5_r4 is non-inferior anywhere, in 2 cells. Every format with $L \geq 16$ is non-inferior somewhere. The relation is not monotone in the clip level within a diagonal — on the $\Delta = 0.25$ diagonal the counts run 2, 0, 17, 16 as L goes 4, 8, 16, 32 — so we do not interpret it as a monotone trend.

Second, at $\Delta = 1$ the two members w5_r16 and w6_r32 return *identical* verdict counts in every scope: 10 / 28 / 10 overall, and the same 25/3 split of the unresolved cells. At cell level they agree in 46 of the 48 cells, and the two exceptions exchange a non-inferior verdict for an unresolved one on the same code and sector rather than crossing the margin. Doubling the range from 16 to 32 at fixed resolution therefore leaves the LER picture essentially unmoved. It does change the resource half substantially, and in the other direction: mean saving falls from 30.6 % to 18.1 % and the joint count from 10 to 5 (Tables 4 and 5).

Over the tested grid, these results resemble a threshold near $L = 16$: below it the clip level dominates, while above it additional range makes less difference. The grid does not bracket or identify an optimum. The pattern is also not separable from

Table 10 Constant-step diagonals. Formats are grouped by quantisation step $\Delta = L/2^{w-1}$; width and clip level increase together within each block while resolution remains fixed.

step Δ	format	clip L	non-inferior
0.015625	w8_r2	2	0
0.03125	w7_r2	2	0
	w8_r4	4	0
0.0625	w6_r2	2	0
	w7_r4	4	0
	w8_r8	8	0
0.125	w5_r2	2	0
	w6_r4	4	0
	w7_r8	8	0
	w8_r16	16	12
0.25	w5_r4	4	2
	w6_r8	8	0
	w7_r16	16	17
	w8_r32	32	16
0.5	w5_r8	8	0
	w6_r16	16	12
	w7_r32	32	12
1	w5_r16	16	10
	w6_r32	32	10

the analytic observation of Section 3.1 that $L \in \{2, 4\}$ lies below the channel input magnitude $\lambda \approx 4.1$ –4.6.

4.11 Restricting the range increases resource usage

Formats at clip level 2 are inferior in all 48 cells at every width, and their traffic saving is large and *negative*: from -277.5% at five bits to -444.5% at eight.

Fig. 4 shows the accompanying iteration count. At clip level 2 the decoder averages about 11 iterations; from clip level 4 upward it averages under 2.1. Because traffic is the product of per-message width and the number of messages moved, an iteration count five times higher overwhelms the saving from narrower messages.

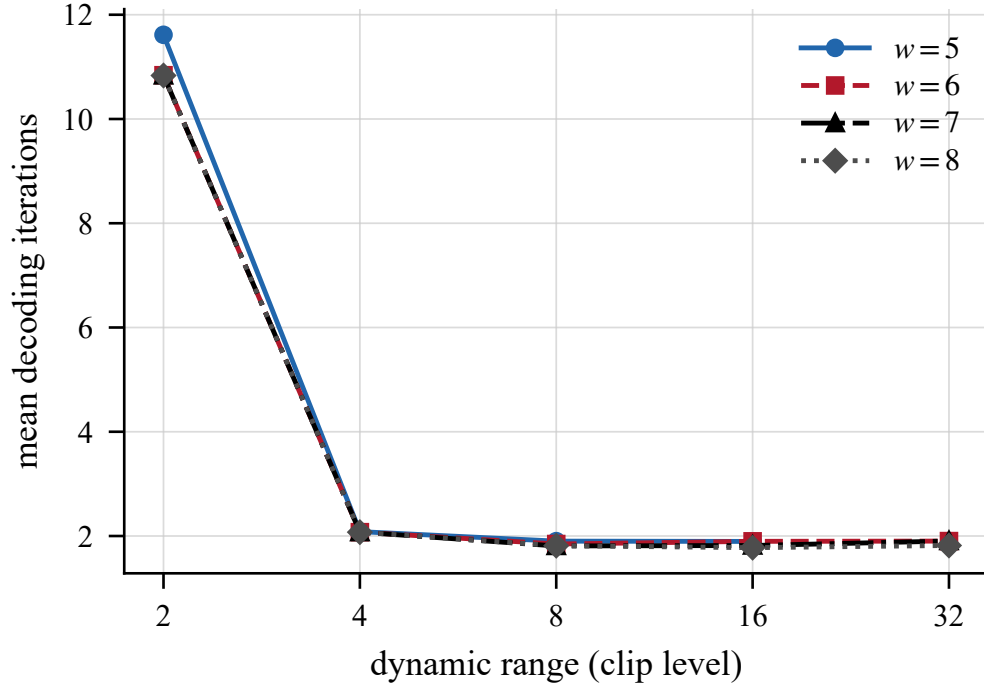


Fig. 4 Iterations by clip level. Mean decoding iterations are averaged over all frozen cells for each word width.

This behaviour is consistent with the saturation account of Section 5.1. It is not a measurement of saturation.

4.12 Seed-level diagnostic

To check whether any one of the three fixed seeds dominates the pooled result, Fig. 5 shows the paired ratio of `w5_r16` at the individual seed level for all 48 cells.

For each cell, dividing the across-seed range by the across-seed median gives a median spread of 0.057 over the 48 cells and a maximum of 0.443, against per-cell ratios that span roughly 0.6 to 12. This is a diagnostic over the three seeds used in the atlas, not an estimate of population-level seed variation or a held-out replication. At each fixed seed, the elevated cluster is confined to the per-code half shown on the left of the figure.

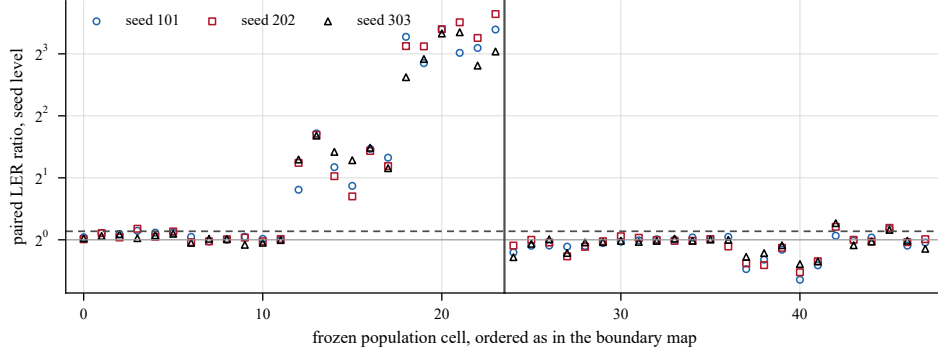


Fig. 5 Seed-level diagnostic. The paired LER ratio of `w5_r16` is shown for each seed separately in every frozen cell. The dashed line is the non-inferiority margin 1.10; the heavy vertical rule separates the two normalisation recipes.

Non-inferior counts are similar across CSS sectors (48 of 456 in X and 43 of 456 in Z) and rise with the physical error rate (23, 23 and 45 of 304 at $p = 0.015$, 0.020 and 0.025). The latter is consistent with higher error rates producing more disagreement and therefore more resolvable cells, which is the power effect of Section 4.9 rather than a statement about precision.

We pool by adding the sufficient statistics of the three paired tables. We do not fit a random-effects model and do not treat the three seeds as a sample from a population of seeds, so no across-seed variance component is claimed.

4.13 Controls

Two plain-min-sum controls are carried in every cell as diagnostics. They never enter the factorial comparison; Table 11 reports them only to show the effect of the normalisation.

`float_plain`, min-sum in floating point with no normalisation at all, is inferior in all 48 cells, with a median paired ratio of 6.047 and a range of 2.69 to 23.86. The fixed-point control, `w6_r8_plain`, is the same unnormalised decoder at six bits and clip level 8. It is inferior in 28 cells with a median ratio of 1.357.

Table 11 Plain min-sum controls over all 48 cells.

control	non-inf.	unres.	inferior	median ratio
<code>float_plain</code>	0	0	48	6.047
<code>w6_r8_plain</code>	7	13	28	1.357

Table 12 Fixed normalisation work per message, by recipe and code.

profile	code	(α, β)	ops/message
per-code	HGP	(0.875, 0.2)	3
per-code	BB72	(0.8125, 0.1)	5
per-code	BB90	(0.9375, 0.2)	3
per-code	BB144	(1, 0.2)	1
common	HGP	(0.75, 0)	2
common	BB72	(0.75, 0)	2
common	BB90	(0.75, 0)	2
common	BB144	(0.75, 0)	2

The diagnostic shows that the (α, β) recipe is not a tuning detail that may be omitted from a decoder specification. These controls are not part of the factorial comparison and are not used to rank normalisation against quantisation.

4.14 Normalisation work is not the explanation

Table 12 reports fixed shift/add work per message. It is a property of the (α, β) recipe alone: for a given recipe and code it is identical at every width and every clip level. Normalisation cost therefore cannot account for any difference reported above, which is precisely why the two resource quantities are kept apart.

5 Discussion

5.1 Interpretation of the width–range trade-off

At fixed width, raising the clip level trades less clipping for coarser resolution. A decoder whose message magnitudes are large relative to L is saturation-limited and

benefits from more range; one whose magnitudes are small relative to L is resolution-limited and is harmed by it. Because normalised min-sum rescales message magnitudes by α against a fixed grid, gain and range move the operating point together.

This qualitative account motivates the format definition but was not tested directly in this study.

5.2 Relation to the reduced-precision qLDPC literature

The closest comparator evaluates eight hypergraph-product and lifted-product codes under hardware emulation, reports several finite-precision formats, observes that precision behaviour changes below a logical error rate of 10^{-4} , and builds a diversity decoder from multiple quantisers [15]. Soft-syndrome architectures at 6- and 8-bit precision [13] and the broader FPGA/ASIC design space [14] occupy the implementation side.

The present study adds a factorial separation of width and clip level within one fixed decoder, which these comparator studies do not report. Holding width fixed and moving clip level shows directly that width alone does not define the operating point: `w8_r32` is non-inferior in 16 cells, whereas `w8_r2` is inferior in all 48.

The comparisons are not like-for-like. That work reports circuit- or emulation-level behaviour at much lower logical error rates; ours is code-capacity behaviour in a regime where paired differences are measurable at 120 000 frames per cell. The two are complementary, and our resource axis is counted message traffic, not a synthesised hardware cost.

5.3 What this evidence does not support

The result is descriptive, and unresolved outcomes dominate. For every format with clip level 8 or wider, between 28 and 36 of the 48 cells are unresolved, and Section 4.9 shows that mass is almost entirely below the discordance floor. We make no confirmatory inference from any individual cell or from the atlas as a whole.

“No format is non-inferior in all cells” is a statement about the tested bank. It means no format met the joint rule in all 48 cells; in many of those cells the rule could not have been met by any format that behaves like the floating-point reference, because the frames do not generate enough disagreement to certify one. It does not establish that no such format exists, over this bank or any other.

The regime matters and bounds the reach of everything above. Reference error rates here run from 6.0×10^{-4} to 2.2×10^{-2} (Table 1), six to 220 times the 10^{-4} reference level below which [15] reports precision behaviour changing. Our rankings should not be extrapolated into that regime.

The resource axis is not independent of decoding quality: traffic includes the executed iteration count, so a format that converges faster scores better on both halves of the joint criterion. Word width is the only part of the resource axis that is orthogonal to convergence. Section 4.7 shows the consequence, that the ranking moves with the threshold, and we report the curve rather than a single point.

We also make no claim at the level of a code *family*. The bank holds one hypergraph-product code and three bivariate-bicycle codes, so any difference we observe is a difference between *these* codes. A single code cannot carry an attribution to the family it belongs to. Relatedly, the per-code recipe confounds code identity with normalisation by construction; the common recipe removes that confound but changes α and β together, so no single-parameter attribution is available either.

Two further scope limits apply. First, the clip grid stops at $L = 32$, while $L = 64$ at $w = 7$ and $L = 64, 128$ at $w = 8$ are representable and were not tested. At the preregistered 25% condition, `w5_r16` has the largest joint-target count and `w6_r32` ranks second; both sit at the largest clip level available at their width. The grid therefore does not bracket the optimum from above. Second, the decoder carries no post-processing. Deployed qLDPC decoders commonly pair belief propagation with ordered-statistics decoding [12, 17, 26]. That stage consumes the reliability *ordering*

1151 of the soft outputs. A format with $\Delta = 1$ offers few distinct magnitudes, and we have
1152 no evidence that this ranking transfers to a BP+OSD pipeline.

1154 The endpoint is single-sector code-capacity logical error rate. We make no area,
1155 energy, latency, throughput, real-time or circuit-level claim, and no error-floor claim.
1157 `w5_r16` has the largest joint-target count *in this frozen bank at a 25% resource*
1158 *condition*; it is not a recommendation to implementers.

1162 5.4 Deviations from preregistration

1164 The design reported here differs from the registered one in four respects, each recorded
1165 before, or independently of, the evidence it touches.

1167 First, the registered primary claim was a *code-family-dependent* crossover. It was
1168 withdrawn and restated before any atlas value was read, because code identity and
1169 normalisation gain are confounded by construction in the preliminary contrast and
1170 because that contrast contains one HGP and one BB code. The full atlas contains
1171 one HGP and three BB codes, which still does not support family attribution. What
1172 survives is the weaker claim this paper makes, that word width does not identify the
1173 format; Section 4.2 reports the evidence that forced the change.

1179 Second, the common normalisation recipe is an amendment. The registered design
1180 fixed one per-code recipe; the common recipe was added after the preliminary contrast
1181 reversed under it, and before any atlas result existed.

1184 Third, the preregistered held-out confirmation on an unseen seed was not run. At
1185 preflight, the two selected cells were projected to carry 887 and 478 discordant pairs,
1186 so the discordance floor was not limiting. The non-inferior cell had margin slack of
1187 only 0.0019 in the atlas and a projected 0.0121 at the fixed 200 000-frame budget, less
1188 than one standard error of the atlas estimate. The other selected cell was projected
1189 to remain inferior, but the preregistered intersection–union rule required both cells to
1190 satisfy their respective verdict criteria. Because the decision depended on a margin
1191
1192
1193
1194
1195
1196

smaller than the estimate’s sampling uncertainty and no second held-out seed was authorised, the run was declined before execution. The consequence is explicit: the atlas has no held-out confirmation, and the study remains descriptive.

Fourth, a preregistered figure plotting the range effect against measured saturation counters is not included. The instrumentation for it was mandated after the atlas had finished generating, so the audited result sets do not carry the counters, and the terminal stopping rule forbade both a re-run and an instrumentation-only pass to obtain them. The central comparisons therefore rest on audited result sets, while the mechanism described in Section 5.1 remains an interpretation rather than a measurement.

6 Conclusions

The frozen 48-cell atlas spans four public codes (one hypergraph-product and three bivariate-bicycle), both CSS sectors, three physical error rates and two normalisation recipes. No fixed-point format was jointly resource-qualified and non-inferior in logical error rate in every cell; `w5_r16` had the largest joint-target count, at 10 of 48. Formats of equal word width differed in both the direction and the magnitude of their effect according to their dynamic range: at eight bits, clip level 32 was non-inferior in 16 cells while clip level 2 was inferior in all 48. Word width is therefore not a sufficient format specification for a fixed-point qLDPC min-sum decoder; the operating point requires the word width together with the normalisation recipe (α, β) and the clip level.

The evidence is descriptive and code-capacity, at reference error rates of 6×10^{-4} to 2.2×10^{-2} on the named bank. For every format at clip level 8 or wider, 28 to 36 of the 48 cells are unresolved, mostly because the fixed-point and floating-point decoders rarely disagree; the format ranking moves with the resource threshold; and no held-out confirmation was run.

1243 **List of abbreviations**

1244

1245 ASIC: application-specific integrated circuit; BB: bivariate-bicycle; BP: belief
1246 propagation; CI: confidence interval; CSS: Calderbank–Shor–Steane; FPGA: field-
1247 programmable gate array; HGP: hypergraph-product; LDPC: low-density parity-
1248 check; LER: logical error rate; LLR: log-likelihood ratio; MIMO: multiple-input
1249 multiple-output; OSD: ordered-statistics decoding; qLDPC: quantum low-density
1250 parity-check.
1251

1252

1253

1254 **Statements and Declarations**

1255

1256

1257 **Availability of data and materials**

1258

1259 The data that support the findings of this study are available from the corresponding
1260 author on reasonable request.
1261

1262

1263

1264 **Competing interests**

1265

1266 The authors declare that they have no competing interests.
1267

1268

1269

1270 **Funding**

1271

1272 Not applicable.
1273

1274

1275

1276 **Authors' contributions**

1277

1278 H.N.D. conceived the study, developed the software, conducted and analysed the
1279 experiments, prepared the figures, and wrote and approved the manuscript.
1280

1281

1282

1283 **Acknowledgements**

1284

1285 Not applicable.
1286

1287

1288

References

- [1] Gallager RG. Low-density parity-check codes. IRE Transactions on Information Theory. 1962;8(1):21–28. <https://doi.org/10.1109/TIT.1962.1057683>.
- [2] Lewandowsky J, Bauch G. Information-Optimum LDPC Decoders Based on the Information Bottleneck Method. IEEE Access. 2018;6:4054–4071. <https://doi.org/10.1109/ACCESS.2018.2797694>.
- [3] Kang P, Cai K, He X, Li S, Yuan J. Generalized Mutual Information-Maximizing Quantized Decoding of LDPC Codes With Layered Scheduling. IEEE Transactions on Vehicular Technology. 2022;71(7):7258–7273. <https://doi.org/10.1109/TVT.2022.3162579>.
- [4] Wang L, Terrill C, Stark M, Li Z, Chen S, Hulse C, et al. Reconstruction-Computation-Quantization (RCQ): A Paradigm for Low Bit Width LDPC Decoding. IEEE Transactions on Communications. 2022;70(4):2213–2226. <https://doi.org/10.1109/TCOMM.2022.3149913>.
- [5] Hatami H, Mitchell DGM, Costello DJ, Fuja TE. A Threshold-Based Min-Sum Algorithm to Lower the Error Floors of Quantized LDPC Decoders. IEEE Transactions on Communications. 2020;68(4):2005–2015. <https://doi.org/10.1109/TCOMM.2020.2969902>.
- [6] Zhang X, Siegel PH. Quantized Iterative Message Passing Decoders with Low Error Floor for LDPC Codes. IEEE Transactions on Communications. 2014;62(1):1–14. <https://doi.org/10.1109/TCOMM.2013.112313.120917>.
- [7] Ngassa CK, Savin V, Dupraz E, Declercq D. Density Evolution and Functional Threshold for the Noisy Min-Sum Decoder. IEEE Transactions on Communications. 2015;63(5):1497–1509. <https://doi.org/10.1109/TCOMM.2015>.

1335 2388472.

1336

1337 [8] Mheich Z, Nguyen-Ly TT, Savin V, Declercq D. Code-aware quantizer design

1338 for finite-precision min-sum decoders. In: 2016 IEEE International Black Sea

1339 Conference on Communications and Networking (BlackSeaCom); 2016. p. 1–5.

1340

1341

1342

1343 [9] Hailes P, Xu L, Maunder RG, Al-Hashimi BM, Hanzo L. A Survey of

1344 FPGA-Based LDPC Decoders. IEEE Communications Surveys & Tutorials.

1345 2016;18(2):1098–1122. <https://doi.org/10.1109/COMST.2015.2510381>.

1346

1347

1348

1349 [10] Dang HN, Nguyen TV. Protograph LDPC Code Design for LS-MIMO 1-bit ADC

1350 Systems. REV Journal on Electronics and Communications. 2021;11(1–2):1–6.

1351 <https://doi.org/10.21553/rev-jec.266>.

1352

1353

1354

1355 [11] Dang HN, Nguyen HT, Nguyen TV. Joint Detection and Decoding of Mixed-ADC

1356 Large-Scale MIMO Communication Systems With Protograph LDPC Codes.

1357 IEEE Access. 2021;9:101013–101029. [https://doi.org/10.1109/ACCESS.2021.](https://doi.org/10.1109/ACCESS.2021.3097444)

1358 [3097444](https://doi.org/10.1109/ACCESS.2021.3097444).

1359

1360

1361

1362 [12] Valls J, Garcia-Herrero F, Raveendran N, Vasic B. Syndrome-Based Min-Sum

1363 vs OSD-0 Decoders: FPGA Implementation and Analysis for Quantum LDPC

1364 Codes. IEEE Access. 2021;9:138734–138743. [https://doi.org/10.1109/ACCESS.](https://doi.org/10.1109/ACCESS.2021.3118544)

1365 [2021.3118544](https://doi.org/10.1109/ACCESS.2021.3118544).

1366

1367

1368

1369

1370 [13] Raveendran N, Valls J, Pradhan AK, Rengaswamy N, Garcia-Herrero F, Vasić

1371 B. Soft syndrome iterative decoding of quantum LDPC codes and hardware

1372 architectures. EPJ Quantum Technology. 2023;10:45. [https://doi.org/10.1140/](https://doi.org/10.1140/epjqt/s40507-023-00201-1)

1373 [epjqt/s40507-023-00201-1](https://doi.org/10.1140/epjqt/s40507-023-00201-1).

1374

1375

1376

1377 [14] Báscones D, Garcia-Herrero F, Valls J. Exploring the FPGA and ASIC design

1378 space of belief propagation and ordered statistics decoders for quantum error

1379

1380

- correction codes. EPJ Quantum Technology. 2025;12:140. <https://doi.org/10.1140/epjqt/s40507-025-00446-y>. 1381
1382
1383
1384
- [15] Garcia-Herrero F, Valls J, Vergara-Picazo L, Torres V. Diversity Methods for 1385
Improving Convergence and Accuracy of Quantum Error Correction Decoders 1386
Through Hardware Emulation. Quantum. 2026;10:2071. <https://doi.org/10.22331/q-2026-04-16-2071>. 1388
1389
1390
1391
1392
- [16] Maan AS, Garcia Herrero FM, Paler A, Savin V. Decoding correlated errors in 1393
quantum LDPC codes. Nature Communications. 2026;17:3965. <https://doi.org/10.1038/s41467-026-70556-3>. 1394
1395
1396
1397
1398
- [17] Roffe J, White DR, Burton S, Campbell E. Decoding across the quantum low- 1399
density parity-check code landscape. Physical Review Research. 2020;2:043423. 1400
<https://doi.org/10.1103/PhysRevResearch.2.043423>. 1401
1402
1403
- [18] Panteleev P, Kalachev G. Degenerate Quantum LDPC Codes With Good 1404
Finite Length Performance. Quantum. 2021;5:585. <https://doi.org/10.22331/q-2021-11-22-585>. 1405
1406
1407
1408
1409
- [19] Raveendran N, Vasić B. Trapping Sets of Quantum LDPC Codes. Quantum. 1410
2021;5:562. <https://doi.org/10.22331/q-2021-10-14-562>. 1411
1412
1413
- [20] Tillich JP, Zémor G. Quantum LDPC Codes With Positive Rate and Minimum 1414
Distance Proportional to the Square Root of the Blocklength. IEEE Trans- 1415
actions on Information Theory. 2014;60(2):1193–1202. <https://doi.org/10.1109/TIT.2013.2292061>. 1416
1417
1418
1419
1420
1421
- [21] Bravyi S, Cross AW, Gambetta JM, Maslov D, Rall P, Yoder TJ. High-threshold 1422
and low-overhead fault-tolerant quantum memory. Nature. 2024;627:778–782. 1423
1424
<https://doi.org/10.1038/s41586-024-07107-7>. 1425
1426

1427 [22] Calderbank AR, Shor PW. Good quantum error-correcting codes exist. Physical
1428 Review A. 1996;54:1098–1105. <https://doi.org/10.1103/PhysRevA.54.1098>.
1429
1430
1431 [23] Steane AM. Error Correcting Codes in Quantum Theory. Physical Review
1432 Letters. 1996;77:793–797. <https://doi.org/10.1103/PhysRevLett.77.793>.
1433
1434
1435 [24] Fossorier MPC, Mihaljević M, Imai H. Reduced complexity iterative decoding of
1436 low-density parity check codes based on belief propagation. IEEE Transactions
1437 on Communications. 1999;47(5):673–680. <https://doi.org/10.1109/26.768759>.
1438
1439
1440
1441 [25] Chen J, Dholakia A, Eleftheriou E, Fossorier MPC, Hu XY. Reduced-
1442 Complexity Decoding of LDPC Codes. IEEE Transactions on Communications.
1443 2005;53(8):1288–1299. <https://doi.org/10.1109/TCOMM.2005.852852>.
1444
1445
1446
1447 [26] Fossorier MPC, Lin S. Soft-decision decoding of linear block codes based on
1448 ordered statistics. IEEE Transactions on Information Theory. 1995;41(5):1379–
1449 1396. <https://doi.org/10.1109/18.412683>.
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472