The Two-Dimensional Overlapping-Window Strengthening of the Normal PEPS Fundamental Theorem

2026-06-13

Verdict: open-gap (open).

1 The source statement

The final unformalized statement of [MGRPG+18] is the two-dimensional strengthening of the torus Fundamental Theorem announced at the end of the overlapping-window section.[1] The source states it as a corollary:

> Let A and B be two normal 2D PEPS tensors such that every $L \times K$ region is injective. Suppose they generate the same state on some region $n \times m$ with $n \geq 2L+1$ and $m \geq 2K+1$. Then A and B are related to each other with a gauge:
>
> $$B^i \;=\; \lambda \cdot (X \otimes Y)\, A^i\, (X^{-1} \otimes Y^{-1}),$$
>
> with $\lambda^{n \cdot m} = 1$ and X, Y invertible matrices. Moreover these matrices X, Y are unique up to a multiplicative constant.

Here the displayed gauge relation abbreviates the source's diagram: X acts on the two horizontal virtual legs of the tensor and Y on the two vertical legs, with inverses on the outgoing sides. The conclusion is identical to that of the source's Theorem 3;[2] only the hypotheses differ. Theorem 3 assumes that every 2×3 *and* every 3×2 region is injective and that $n, m \geq 7$. The corollary assumes injectivity of a single rectangle shape $L \times K$ — one orientation only — and the sizes $n \geq 2L+1$, $m \geq 2K+1$. For $L = K = 2$ this reads: injectivity of every 2×2 region and sizes $n, m \geq 5$, strictly weaker hypotheses than the formalized theorem in both the region shapes and the sizes. The strengthening is therefore real even in the smallest case, and grows with L, K (for $L = 2$, $K = 3$ it gives 5×7 in place of 7×7).

[1] `Papers/1804.04964/paper_normal.tex:2296--2318`.

[2] Theorem 3 is at `Papers/1804.04964/paper_normal.tex:1451--1472`; its torus form is formalized as `TNLean.PEPS.fundamentalTheorem_normalTorusPEPS_unconditional` in `TNLean/PEPS/TorusFundamentalTheorem2.lean`.

2 The source proof sketch

The source gives only a proof sketch.[3] Its full text, with the diagrams described in words:

> We only need to prove a statement similar to Lemma 5. For that, notice that a virtual operation on a given bond can be interpreted as a physical operation on any of the following four regions (in the case of $K = L = 2$): [four 2×2 windows around a horizontal edge: the window right of the edge, then sliding left across the edge in two steps, then sliding down one step]. We need to prove that conversely, any four physical operators on the above regions that transforms the PEPS into the same state means that the transformation can equally be done with a virtual operation on the highlighted edge. The system size required to compare any two consecutive regions is only 5×5 (and in general, $(2L+1) \times (2K+1)$). Therefore, similar to Lemma 5, [the four operators applied to the patch around the edge agree] with open boundaries. Compare now the first and the last expression in the above equation. One can add two-two tensors in the upper left and lower right corner and invert the resulting regions, leading to [the equality restricted to the staircase-shaped union of the first and last windows]. This results in the desired virtual operation on the highlighted edge. The rest of the proof is the same as the proof of Theorem 3.

Lemma 5 is the one-dimensional overlapping-window bond-operator extraction,[4] which delivers the one-dimensional Fundamental Theorem at $n \geq 2L + 1$ in place of the three-block bound $n \geq 3L$.

2.1 What the sketch asserts, and where the Lemma 5 analogy fails

The sketch is explicitly labelled "Sketch of proof"[5] and its single load-bearing claim is asserted by analogy to Lemma 5 rather than derived. The claim is the converse correspondence between virtual and physical operations: "a virtual operation on a given bond can be interpreted as a physical operation on any of the $[\, L + K \,]$ regions," and conversely "any $[\, L + K \,]$ physical operators on the above regions that transforms the PEPS into the same state means that the transformation can equally be done with a virtual operation on the highlighted edge."[6] This converse is exactly the cross-bond-pushing the formalization iso-

[3] `Papers/1804.04964/paper_normal.tex:2320--2445`.

[4] `Papers/1804.04964/paper_normal.tex:2045--2255`, formalized (site-dependent form) as `MPSChainTensor.overlapWindow_exists_bondOperator` in `TNLean/PEPS/CycleMPSChainOverlapWindow.lean`, with the closed-chain corollary `TNLean.PEPS.fundamentalTheorem_normalMPSChain_of_overlap` in `TNLean/PEPS/CycleMPSChainOverlapCapstone.lean`.

[5] `Papers/1804.04964/paper_normal.tex:2320`.

[6] `Papers/1804.04964/paper_normal.tex:2321` and `Papers/1804.04964/paper_normal.tex:2368`.

lates as its one residual: from a common state on the window chain, recover a single matrix X on the edge together with the two factorizations through the end windows.

The sketch supplies this converse by the reduction "we only need to prove a statement similar to Lemma 5"[7] and "therefore, similar to Lemma 5, [the four operators on the patch agree] with open boundaries,"[8] followed by the corner completion "one can add two-two tensors in the upper left and lower right corner and invert the resulting regions."[9] Two of these three steps transfer. The consecutive-window agreement with open boundaries, obtained by inverting the complement of each consecutive-window union, is rigorous and needs only the sizes $n \geq 2L+1$, $m \geq 2K+1$ (the sketch's "5×5" / "$(2L+1) \times (2K+1)$"); the corner completion is rigorous as a cancellation of one injective $L \times K$ rectangle. Both are formalized; see the filled-in derivation below.

The step that does *not* transfer is the final extraction of the single matrix X on the edge, which the sketch obtains by analogy to the closing move of Lemma 5. In Lemma 5 that move is the comparison of the two ends of the chain, "similar to" the injective read-off,[10] which there yields the bond matrix X and, by composition, the algebra homomorphisms $O_1 \mapsto X$ and $O_3^T \mapsto X$.[11] That read-off is valid in one dimension for a structural reason specific to a one-dimensional window: a length-L window has exactly two virtual legs, a left bond and a right bond, both of dimension D, so inverting the injective window tensors produces a genuine $D \times D$ *matrix* W on the bond,[12] and window injectivity is precisely the statement that these matrices span the full bond algebra, which is what makes the homomorphism extraction go through.

In two dimensions the window straddling a bond no longer has this structure. An $L \times K$ window touches the reference edge e with *one* endpoint, so the leg of e inside an end window is a single $\mathrm{Fin}(\mathrm{bondDim}\, e)$ index, not a pair; reading the window block at fixed external legs gives a *vector* on $\mathrm{Fin}(\mathrm{bondDim}\, e)$, never a square matrix on the bond, and the rest of the window's perimeter is open. The two indices of the matrix X the corollary wants are the e-legs of the two diagonally offset end windows, one from each; no single window supplies a square-matrix family, and folding a window's external perimeter into a second $\mathrm{Fin}(\mathrm{bondDim}\, e)$ index would require $\prod_{\text{external legs}} \mathrm{bondDim} = \mathrm{bondDim}\, e$, false in general. So the structural fact that powers the closing move of Lemma 5 — a window product is a square matrix on the pushed bond — is absent for the non-square $L \times K$ window, and the reduction "similar to Lemma 5" does not carry the extraction of X in two dimensions. The sketch glosses this dimensional gap.

[7] `Papers/1804.04964/paper_normal.tex:2321`.

[8] `Papers/1804.04964/paper_normal.tex:2368`.

[9] `Papers/1804.04964/paper_normal.tex:2415`.

[10] `Papers/1804.04964/paper_normal.tex:2213`, invoking `Papers/1804.04964/paper_normal.tex:377` (`eq:inj_O->X_argument`).

[11] `Papers/1804.04964/paper_normal.tex:2129` and `Papers/1804.04964/paper_normal.tex:2252`.

[12] The inverted-window matrix is the W of `Papers/1804.04964/paper_normal.tex:401–403`, an $D_{23} \times D_{23}$ object on the single bond.

Source-rigor verdict. The published corollary proof is a sketch. Its consecutive-window agreement and corner-completion steps are rigorous and transfer to two dimensions; its final extraction of the single edge matrix X is asserted by analogy to Lemma 5, and that analogy fails because the square-window geometry Lemma 5's closing read-off relies on is not present for a two-dimensional $L \times K$ window. The sketch therefore does not rigorously establish the corollary from $L \times K$-region injectivity at the minimal size: the cross-bond-pushing correspondence it asserts is the one step it does not derive. This is a faithfulness/correctness finding about the published sketch; it is not a defect in the formalization, which captures the source's hypothesis set exactly (see the faithfulness verdict below).

3 Consolidated account

This section is the self-contained summary of the investigation. The remaining sections record, in full, the machinery that was built and the machine-checked refutations of each closing route; this section states the conclusions a reader needs without that detail.

1. The statement and its faithful Lean hypotheses. The source corollary[13] asserts that two normal 2D PEPS tensors A, B with every $L \times K$ region injective, generating the same state on an $n \times m$ region with $n \geq 2L+1$, $m \geq 2K+1$, are related by $B^i = \lambda\,(X \otimes Y)\,A^i\,(X^{-1} \otimes Y^{-1})$ with $\lambda^{nm} = 1$ and X, Y unique up to a constant. "Normal ... such that every $L \times K$ region is injective" is the source's own normality condition[14] instantiated at the $L \times K$ blocking; it is not a second, stronger hypothesis. The faithful Lean hypothesis bundle is `TNLean.PEPS.NormalTorusArcWindowInjectivityHypotheses` (`TNLean/PEPS/TorusWindowComplement.lean`), a single field `arcWindow_injective`: every cyclic $L \times K$ window is injective in the sense of `TNLean.PEPS.RegionBlockedTensorInjective`. This neither exceeds nor falls short of the source's hypothesis.

2. What is formalized. All surrounding machinery is built and lands at the corollary's minimal size: the seam-aware window geometry (`TNLean/PEPS/TorusWindowRegion.lean`, `TNLean/PEPS/TorusWindowComplement.lean`); the deformed-window state vocabulary and the consecutive-window comparison by complement inversion (`TNLean/PEPS/TorusDeformedWindow.lean`); the chaining, corner completion, and shared-corner cancellation that produce the *open-boundary* end-pair equality `TNLean.PEPS.staircasePair_insert_eq_open` (`TNLean/PEPS/TorusWindowChain2.lean` through `TorusWindowChain6.lean`) without ever inverting the non-injective univ $\setminus S$; the single-bond peeling of that equality onto the reference edge (`TNLean.PEPS.regionInsertedCoeff_en`

[13] `Papers/1804.04964/paper_normal.tex:2297--2318`.

[14] `Papers/1804.04964/paper_normal.tex:1318`: "We call a PEPS *normal*, if blocking tensors in certain regions results in injective tensors."

dWindows_eq_of_staircase, TNLean/PEPS/TorusWindowPeeling/SingleBond.lean); the per-edge witness packaging and the Skolem–Noether read-off (TNLean/PEPS/TorusWindowWitness.lean, TNLean/PEPS/TorusWindowMult.lean); the per-step bond-insert transport TNLean.PEPS.horizontalStaircaseConsecutiveWindow_bondTransport_extend_eq and the now-uniform interior-bond rescaling TNLean.PEPS.regionInteriorBondProd_staircaseWindow_eq (TNLean/PEPS/TorusWindowBondTransport.lean, TorusWindowBondUniform.lean); and the back-half assembly that re-anchors the torus Theorem 3 layers to the $(L+1) \times (K+1)$ comparison pair. Every inverted region is a union of one-orientation $L \times K$ window translates, host-decomposable at $n \geq 2L+1$, $m \geq 2K+1$. The development is minimal-size-ready and gated on one input alone.

3. The irreducible obstruction. The one missing input is the forward-transfer multiplicativity `hmul` at the reference edge: the homomorphism $M \cdot M' \mapsto$ (transfer of M) (transfer of M'). This is the cross-bond-pushing of Part 1 — the realization of one virtual operation M on e as a coherent physical operation on every window of the chain, composed across the chain. Reading it off requires either single-vertex injectivity at the edge endpoint (TNLean.PEPS.span_stateOpenCoeff_eq_top, from TNLean.PEPS.IsVertexInjective) or a coherent three-block coarse frame around the bond (TNLean.PEPS.coeffTransferMap_mul_of_coherentFrames).

4. Every closing route is machine-checked impossible at the minimal size.

- *Single-vertex injectivity is unavailable.* Normality is by definition *weaker* than single-vertex injectivity: a normal tensor need not be injective at one site but becomes injective after blocking. So normality cannot supply TNLean.PEPS.IsVertexInjective, and the endpoint span it would feed (TNLean.PEPS.span_stateOpenCoeff_eq_top) is not derivable from the hypotheses.

- *The one-dimensional port is absent.* The one-dimensional extraction MPSChainTensor.overlapWindow_exists_bondOperator succeeds from window injectivity alone because a length-L one-dimensional window product is a square matrix on its bond. A two-dimensional $L \times K$ window touches the bond with one endpoint and gives a vector family, never a square-matrix span (Part 1).

- *The end-pair coarse frame is non-injective.* Taking the red block to be one end window forces the single-crossing partner to be the opposite end window, so the third region of the frame is univ $\setminus S$, which is not injective at the minimal size: the end pair occupies all $2L$ columns of one row and the complement band there is below L columns wide.

- *No per-bond coarse frame exists, for any single-crossing window pair.* The only landed source of complement injectivity at the minimal size needs red $\cup$ blue to be a single cyclic rectangle, but two $L \times K$ windows cross at a single edge only when they are the diagonally offset pair, whose union is a staircase, not a rectangle (`TNLean.PEPS.horizontalStaircasePair_ne_arcRectangle`). This is a joint unsatisfiability, settled combinatorially in `TNLean/PEPS/TorusWindowSingleCrossingObstruction.lean`.
- *Translation invariance dissolves only the rescaling sub-wall.* The corollary's torus translation invariance equalizes the per-step interior-bond rescalings (`TNLean.PEPS.regionInteriorBondProd_staircaseWindow_eq`), removing the sub-obstruction the plane setting could not, but it does not supply the cross-bond pushing.
- *The row-and-column one-dimensional reduction fails.* Treating each row as one coarse site and applying the one-dimensional theorem row-wise and column-wise cannot reach the per-edge conclusion: a contracted row is invariant under conjugating its bond family by an arbitrary invertible matrix, so the reduction cannot distinguish a per-edge-gauged row from a non-product conjugate that has the same state and injectivity but no per-edge factorization. This is a machine-checked refutation (`TNLean.PEPS.RowColumnReductionObstruction.rowCutGaugeFactorizes_false`, `TNLean/PEPS/TorusRowColumnReductionObstruction.lean`).

5. The source-rigor verdict. The published corollary proof is a sketch that asserts the cross-bond-pushing correspondence by analogy to Lemma 5; that analogy fails dimensionally, because the square-window geometry Lemma 5's closing read-off relies on is absent for the non-square two-dimensional window (Part 1 above). The sketch's consecutive-window agreement and corner completion are rigorous; the extraction of the single edge matrix is the step it asserts without derivation.

6. Status. The corollary remains *unformalized*: no blueprint entry claims it, it carries the `notready` marker, and no `leanok` marks it. It is not closeable from its stated hypotheses by any route rigorously explored here. Closing it would require either a new mathematical idea evading the machine-checked impossibilities above, or the source supplying the missing cross-bond-pushing argument that its sketch only asserts.

4 The filled-in derivation

This section records the complete argument the sketch compresses, in the notation of the formal torus development: the lattice is the discrete $n \times m$ torus, regions are sets of vertices, T_R denotes the tensor of the network blocked over a region R, and a region is injective when T_R is injective as a map from its virtual boundary indices to its physical indices.

4.1 The window family around an edge

Fix a horizontal edge e with endpoints $u = (x, y)$ and $w = (x + 1, y)$. Define $L + K$ windows, each a contiguous $L \times K$ rectangle:

$$W_j = [x + 1 - j,\, x + L - j] \times [y,\, y + K - 1], \qquad j = 0, \ldots, L,$$
$$W_{L+i} = [x - L + 1,\, x] \times [y - i,\, y + K - 1 - i], \qquad i = 1, \ldots, K - 1.$$

The first $L + 1$ windows slide left across the edge one column at a time; the last $K - 1$ windows descend one row at a time. The first window W_0 contains only the right endpoint w (at its lower-left corner), the windows $W_1, \ldots, W_{L-1}$ contain both endpoints, and the windows $W_L, \ldots, W_{L+K-1}$ contain only the left endpoint u. For $K = L = 2$ these are exactly the four regions of the sketch. Around a vertical edge the construction is transposed in the roles of the two axes with the same $L \times K$ window shape: $K+1$ vertical slides followed by $L-1$ horizontal ones.

A virtual operation M on the bond e is realized as a deformation of any single window containing an endpoint of e: invert the injective window tensor and recontract with M inserted on the bond. The two-dimensional analogue of Lemma 5 is the converse. Suppose each window W_j carries a deformed tensor C_j (an arbitrary tensor with the boundary structure of W_j) and all the deformed states agree: replacing T_{W_j} by C_j in the torus contraction yields one common state for $j = 0, \ldots, L + K - 1$. Then there is a single matrix X on the bond e with

$$C_0 = T_{W_0} \cdot X_e \qquad \text{and} \qquad C_{L+K-1} = X_e \cdot T_{W_{L+K-1}},$$

where X_e denotes the insertion of X on the open leg of e, and X is unique.

4.2 Step 1: comparing consecutive windows

For consecutive windows the union $U_j = W_j \cup W_{j+1}$ is an $(L + 1) \times K$ rectangle (horizontal slides) or an $L \times (K + 1)$ rectangle (vertical slides). The torus complement of U_j is injective: it decomposes into a full-height band of L (respectively $L + 1$) columns and a leftover rectangle of size $(L + 1) \times (K + 1)$ (respectively $L \times K$), and each piece is a union of translated $L \times K$ rectangles. The decomposition needs exactly $n \geq 2L + 1$ and $m \geq 2K + 1$; this is the only step of the whole argument where the torus size enters, matching the source's parenthetical "$(2L+1) \times (2K+1)$". The bands may wrap the torus seam, so the covering rectangles are translates of coordinate rectangles; injectivity transports along translations.

Inverting the complement strips the state equality for j and $j + 1$ to an open-boundary equality on the union:

$$C_j \sqcup T_{W_{j+1} \setminus W_j} \;=\; T_{W_j \setminus W_{j+1}} \sqcup C_{j+1} \qquad \text{as tensors on } U_j.$$

This is the two-dimensional form of the inversions producing the source's displays around Lemma 5.[15]

[15] The one-dimensional inversions are at `Papers/1804.04964/paper_normal.tex:2133--2179` (`eq:normal_act_123` and `eq:normal_act_234`).

4.3 Step 2: chaining to the patch

Let $P = \bigcup_j W_j$, the staircase-shaped patch. Contracting each consecutive-window equality with the tensors of $P \setminus U_j$ extends it to P, and transitivity chains the $L + K - 1$ equalities into one comparison of the two end windows:

$$C_0 \sqcup T_{P \setminus W_0} \;=\; T_{P \setminus W_{L+K-1}} \sqcup C_{L+K-1} \qquad \text{on } P$$

with open boundary. This is the analogue of the source's plugging of A_4 and A_1 in the proof of Lemma 5.

4.4 Step 3: stripping to the staircase pair

Let $S = W_0 \sqcup W_{L+K-1}$, the disjoint union of the two end windows. The difference $P \setminus S$ is the single block $[x-L+1, x] \times [y+1, y+K-1]$ of size $L \times (K-1)$ (the sketch's "upper left corner"; the sketch presents the comparison on the full bounding box, where the lower-right corner block appears as well, and removes both). An $L \times (K-1)$ block is not injective, so it cannot be inverted directly. Instead, complete it: contract both sides of the patch equality with one added row of network tensors above the block, forming the $L \times K$ rectangle $[x-L+1, x] \times [y+1, y+K]$. The completed rectangle is injective, and inverting it removes both the added row and the corner block. This is the sketch's "add two-two tensors in the ... corner and invert the resulting regions". The result is the open-boundary equality on the staircase pair:

$$C_0 \sqcup T_{W_{L+K-1}} \;=\; T_{W_0} \sqcup C_{L+K-1} \qquad \text{on } S.$$

The inversion in this step cancels a *shared* injective block. Both sides of the patch equality carry the genuine network block on the completed rectangle $Q = [x-L+1, x] \times [y+1, y+K]$ once the added row is contracted in; they differ only on S. Injectivity of Q then cancels the common Q-block from both sides and leaves the equality on S. This is a different inversion from the consecutive-window step of Step 1, where the inverted region is the entire torus complement of the compared region and the two sides differ across that whole complement.

The complement of the end pair is not injective at the minimal size. A natural but incorrect reading of the corner completion is that it makes the torus complement univ $\setminus S$ injective, so that the consecutive-window engine of Step 1 (which inverts the full complement of the compared region) applies with the compared region taken to be S itself. This fails at the corollary's minimal size. Read the cyclic columns and rows from the lower-left corner of the left end window. The left window occupies columns $[0, L)$ at rows $[0, K)$ and the right window occupies columns $[L, 2L)$ at rows $[K-1, 2K-1)$, so at the single row $K-1$ the pair occupies *all* columns $[0, 2L)$. The complement of S in that row is the column band $[2L, \text{width})$, of width $\text{width} - 2L$, which is exactly 1 at $\text{width} = 2L+1$ and in general below L whenever $\text{width} < 3L$. Every contiguous $L \times K$ window (or wider) that contains a complement vertex at row $K-1$ must include a column in $[0, 2L)$ at that row, and every such column lies in S. So no

injective $L \times K$ translate fits through row $K-1$ of the complement, and univ $\setminus S$ is not a union of injective window translates; its blocked tensor is not injective under the one-orientation window hypotheses alone at this size. (For width $\geq 3L$ and height $\geq 3K$ the full-width and full-height gaps both reach L and K, and univ $\setminus S$ does decompose; the corollary needs the minimal size where it does not.) The same obstruction rules out the variant univ $\setminus S = (\text{univ} \setminus P) \cup$ corner: the patch P also occupies all columns $[0, 2L)$ at row $K-1$, so univ $\setminus P$ has the same sub-L band there.

The faithful step is therefore the shared-block cancellation above, which needs only injectivity of the completed rectangle Q (an $L \times K$ window, injective by hypothesis) and never asserts injectivity of univ $\setminus S$. The Lean stripping is the open-boundary shared-corner cancellation `staircasePair_insert_eq_open`: it cancels the shared injective completed rectangle Q from the patch equality and never inverts univ $\setminus S$. An earlier closed-torus stripping discharged Step 3 with the full-complement engine `deformedRegionStateAssembled_insert_eq_of_complementInjective` and so carried `RegionBlockedTensorInjective` B (univ $\setminus$ `horizontalStaircaseEndPair` $s\,L\,K$) as an explicit hypothesis not derivable at the minimal size; it has been removed in favour of the shared-injective-block cancellation engine for Q, which inverts the common completed-corner block rather than the whole complement.

4.5 Step 4: the single-crossing extraction

The two end windows are diagonally offset copies of the $L \times K$ rectangle:

$$W_0 = [x+1, x+L] \times [y, y+K-1], \qquad W_{L+K-1} = [x-L+1, x] \times [y-K+1, y].$$

Their column ranges are disjoint and their row ranges intersect exactly in $\{y\}$, so the *only* lattice edge joining them is e itself: a horizontally adjacent pair must sit in columns x and $x+1$ at a common row in $[y, y+K-1] \cap [y-K+1, y] = \{y\}$, and a vertically adjacent pair would need a common column. The staircase equality is therefore a pairing of two injective tensors across the single bond e, and the standard two-sided extraction — combine one family by coefficients representing the identity, exactly as in the injective case[16] — produces the unique matrix X on e with the two factorizations. This single-crossing geometry is the reason the extracted operator lives on one edge: no multi-edge cut is ever inverted across.

The single-crossing geometry, formalized at the end-pair boundary level. The geometric core of Step 4 is in `TNLean/PEPS/TorusWindowExtraction.lean`: the reference edge e is the unique lattice edge joining the two end windows (`isCrossingEdge_horizontalStaircaseEndWindows`, lifting the abstract `isCrossingEdge_horizontalStaircase` to the actual `horizontalStaircaseLeftWindow` and `horizontalStaircaseRightWindow`), hence a boundary edge of each window (`isRegionBoundaryEdge_horizontalStaircaseLeftWindow_referenceEdge` and its right counterpart) but *not* of the end

[16] `Papers/1804.04964/paper_normal.tex:377` (`eq:inj_O->X_argument`); the matrix-level form is `MPSChainTensor.exists_bondOperator_of_intertwine_span`.

pair $S = W_0 \sqcup W_{L+K-1}$ (`not_isRegionBoundaryEdge_horizontalStaircaseEndPair_referenceEdge`): both endpoints of e lie in S, one in each window, so e is the single interior bond of S. Every other boundary edge of either window is a boundary edge of S (`isRegionBoundaryEdge_endPair_of_leftWindow` and its right counterpart), so the boundary ∂S is the disjoint union of the two windows' external legs with e the only shared interior bond. This is the geometric data the bond-operator extraction consumes.

The matrix-intertwining route does not apply per window in two dimensions. The footnoted matrix-level form `MPSChainTensor.exists_bondOperator_of_intertwine_span` extracts X from spanning families W_1, W_2 of *square* matrices Mat_D with $F\,a \cdot W_2\,b = W_1\,a \cdot G\,b$. In the one-dimensional chain it applies directly because a length-L window has two virtual legs — a left bond and a right bond, both of dimension D — so the window product `arcEval` $A\,r\,(\mathrm{ofFn}\,a)$ and the deformed window $C\,0\,a$ are themselves $D \times D$ matrices. In two dimensions a single end window touches the bond e with *one* endpoint, so the e-leg is a single $\mathrm{Fin}(\mathrm{bondDim}\,e)$ index, not a pair: a window block read at fixed external legs and physical configuration is a *vector* on $\mathrm{Fin}(\mathrm{bondDim}\,e)$, not a square matrix. The two indices of the extracted $X(i,j)$ are the e-leg of W_0 (one) and the e-leg of W_{L+K-1} (the other); neither window alone supplies a square matrix family, and folding a window's external perimeter into a second $\mathrm{Fin}(\mathrm{bondDim}\,e)$ index would require $\prod_{\text{external legs}} \mathrm{bondDim} = \mathrm{bondDim}\,e$, false in general. So `exists_bondOperator_of_intertwine_span` cannot be fed from per-window blocks.

The faithful two-dimensional extraction is the single-crossing algebra-isomorphism route the torus Theorem 3 development already uses (`exists_regionEdgeGauge_of_blockingData` of `TNLean/PEPS/CoherentFrameInstance2.lean`): the region-inserted coefficient `regionInsertedCoeff` inserts a matrix on the single boundary edge e and contracts the red block against the host, and Skolem–Noether (`edgeGaugeFromInsertionAlgebraIsomorphism`) turns the resulting algebra isomorphism into conjugation by an invertible edge matrix Z. The Theorem 3 engine inverts the *host* univ $\setminus$ red (its hypothesis `hCB`), which the corollary's minimal size does not provide; the overlapping-window replacement plays the second end window W_{L+K-1} in the host's role, inverting only the two injective windows. Reaching X from the end-pair equality `staircasePair_insert_eq_open` therefore needs the `extendInsert` $(W_0 \subseteq S)$-to-single-bond-e peeling: reading the corner extension on S as a `regionInsertedCoeff`-style contraction of C_0 against the genuine W_{L+K-1} block across e. That peeling is the host-boundary-edge embedding and the Q-weight span lemma recorded as the unbuilt residual at the close of the previous section; the single-crossing geometry above is the prerequisite it consumes, and is now in place.

4.6 Step 5: the rest of the proof

With the per-edge extraction in place, the remainder is, as the source says, the same as the proof of Theorem 3, and the torus formalization already realizes that part as a chain of layers: the insertion correspondence turns the extraction into an algebra isomorphism of the bond algebra at one reference edge per orientation, inner by Skolem–Noether, giving a reference gauge with a coefficient identity; translation transport spreads the reference witness to every edge as a translation-covariant absorbed family; a comparison of a region R with $R \cup \{v\}$ extracts the per-vertex scalar λ; and one global contraction forces $\lambda^{nm} = 1$.[17] Two of these layers are anchored to the Theorem 3 shapes and must be re-anchored:

- The comparison pair: the formalized pair is the 3×3 square minus its corner against the full square. The one-orientation replacement is the $(L+1) \times (K+1)$ rectangle minus one corner vertex against the full $(L+1) \times (K+1)$ rectangle; the smaller region is the union of the three $L \times K$, $L \times (K+1)$-, and $(L+1) \times K$-shaped unions of $L \times K$ rectangles, the full rectangle is such a union as well, and both complements decompose into bands and one rectangle at $n \geq 2L+1$, $m \geq 2K+1$.
- The uniqueness clause of the corollary needs no system size and is already delivered by the formalized gauge-uniqueness statement for the torus family.

5 What the sketch does not say, and what it does not need

5.1 No coherence question between rows

A natural shortcut would treat each row of the torus as one site of a coarse closed chain and apply the one-dimensional overlapping-window theorem twice, row-wise and column-wise. That route produces one gauge per column *cut* — an invertible matrix on the whole product of the bond spaces crossing the cut — and would then face a genuine coherence problem: decomposing the cut gauges into one matrix per edge, uniformly across rows, which the one-dimensional argument does not provide. The source's sketch does not take this route and the question never arises. The window chain is intrinsically two-dimensional and local to one edge: the windows turn the corner around the edge (sliding in both directions), and the end pair's single-crossing geometry forces the extracted operator onto that edge alone. Coherence *across* edges is inherited from translation transport of one reference witness per orientation, exactly as in Theorem 3.

[17] `TNLean.PEPS.exists_torusCovariantAbsorbedGaugeFamily`, `TNLean.PEPS.component_eq_gaugeVertex_of_cornerProportional`, and `TNLean.PEPS.lambda_pow_card_torus_eq_one`, assembled in `TNLean/PEPS/TorusFundamentalTheorem2.lean`.

The coherence problem is not merely unaddressed by the sketch; it is a genuine obstruction, and the row-and-column shortcut cannot reach the per-edge conclusion. The one-dimensional reduction reads a contracted row only as an abstract family of matrices on the collected vertical-bond space together with the closed-chain trace coefficients, and both inputs are invariant under conjugating that family by an arbitrary invertible matrix of the collected bond space. A non-product conjugation produces a contracted row with the same coefficients and the same injectivity as a per-edge-gauged one, so the reduction cannot tell them apart, while only the latter has a per-edge factorization; and because an injective contracted row has scalar self-conjugations, the reduction's gauge is that conjugator up to a scalar, non-factoring whenever the conjugator is. The companion column-wise application does not help: a row cut crosses the vertical edges of one row boundary and a column cut the horizontal edges of one column boundary, two disjoint edge sets sharing no bond, so the two cut gauges impose no mutual consistency. This is recorded as a machine-checked refutation in `TNLean/PEPS/TorusRowColumnReductionObstruction.lean` (`TNLean.PEPS.RowColumnReductionObstruction.rowCutGaugeFactorizes_false`): an injective contracted-row family and its non-product conjugate share their state and injectivity, yet no per-edge product conjugator relates them.

5.2 Why the formalized three-block route cannot reach the corollary's sizes

The formalized witness production for Theorem 3 blocks the torus into three injective regions around the reference edge — a red block, a blue block meeting it only at the edge, and the complementary host — and the host must be injective. At the corollary's minimal size this fails. Take $L = K = 2$, the 5×5 torus with columns and rows $0, \dots, 4$, the edge e joining $(1, 2)$ and $(2, 2)$, and the natural single-crossing pair: red $[0, 1] \times [1, 2]$, blue $[2, 3] \times [2, 3]$. The host contains the vertex $(4, 2)$, but every 2×2 square containing $(4, 2)$ — including those wrapping the seam — meets red or blue. The host is therefore not a union of injective 2×2 regions, and its injectivity is not derivable from the hypotheses. The overlapping-window route avoids the host entirely: the only inverted complements are those of single windows and of consecutive-window unions, all of which decompose at $(2L + 1) \times (2K + 1)$.

5.3 Scope restrictions

- **One orientation suffices.** The corollary assumes injectivity of $L \times K$ regions only, with no transposed $K \times L$ hypothesis. The filled-in derivation confirms this: every region ever inverted is a union of $L \times K$ translates. The formalized Theorem 3 hypotheses (`TNLean.PEPS.NormalTorusRectangleInjectivityHypotheses`) require both orientations, so the corollary needs a new one-orientation hypothesis structure.

- **The window must straddle the edge: $L, K \geq 2$.** The chain around a

horizontal edge passes through windows containing both endpoints, which requires $L \geq 2$; vertical edges require $K \geq 2$. The source states the corollary for unqualified L, K but proves the sketch at $L = K = 2$ and never addresses $L = 1$ or $K = 1$, where no window straddles the relevantly oriented edge and the chain cannot cross it. The formalization should carry $2 \leq L$ and $2 \leq K$ as explicit hypotheses; this is a locally-fixable clarification of the source statement, recorded here.

- **Region conventions.** As in the landed torus theorem, positive bond dimensions are assumed (the union lemma for overlapping regions needs them), and the formal conclusion is the torus surface of the source display: a translation-covariant per-edge family in place of the literal single pair (X, Y), since the ordered-edge convention stores wraparound edges with swapped endpoints (see the section "Closure on the torus" of `docs/paper-gaps/peps_normal_ft_section3_route.tex`).

6 Formalization plan

The geometry and state-equality layers of the derivation above are complete; the cross-bond-pushing multiplicativity is the irreducible obstruction identified in the consolidated account, and the layer list below should be read as the plan for everything *except* that one input. The build is a campaign comparable to the landed three-block witness production, in independently landable layers:

1. *Window geometry* (no tensors): the one-orientation rectangle-injectivity hypotheses; injectivity of all $a \times b$ rectangles with $a \geq L$, $b \geq K$ by sliding unions; the window family around an edge; disjointness of the end pair and the single-crossing characterization; the complement decompositions of single windows and consecutive-window unions at $n \geq 2L+1$, $m \geq 2K+1$, including the seam-wrapping bands.

2. *Deformed-window states*: the vocabulary for a torus state with one region's block replaced by an arbitrary tensor, and the consecutive-window comparison by complement inversion (the existing kernel-descent and two-block machinery applies once the complement injectivity is supplied by layer 1).

3. *Chaining and corner stripping*: extension of an open-boundary equality by contraction, and the completion-inversion step removing the corner block.

4. *The single-crossing extraction* on the staircase pair, reusing the existing single-boundary-edge comparison engine.

5. *The per-edge witness*: the insertion correspondence and Skolem–Noether at one reference edge per orientation, producing the same coefficient-identity witness shape the landed translation layers consume.

6. *Re-anchoring the downstream*: the covariant family, the $(L+1)\times(K+1)$ comparison pair, the scalar extraction, and the capstone with hypotheses: one-orientation $L \times K$ injectivity for both tensors, $2 \le L$, $2 \le K$, $n \ge 2L+1$, $m \ge 2K+1$, matched bond dimensions, positive bonds, the same state.

Layer 1 is foundational and self-contained; layers 2–5 replace the landed coarse three-site witness engine; layer 6 is a re-anchoring of existing arguments. Until the capstone lands, the corollary at `Papers/1804.04964/paper_normal.tex:2297--2318` remains unformalized, and no blueprint entry should claim it.

Layer 1 is complete. The seam-avoiding window geometry — the one-orientation hypotheses, the general $a \times b$ injectivity by sliding unions, the staircase end pair, and the single-crossing characterization — is in `TNLean/PEPS/TorusWindowRegion.lean`. The seam-wrapping part — the cyclic rectangle, the translation-invariant window hypotheses, the consecutive-window unions, and the complement decompositions of single windows and consecutive-window unions at $n \ge 2L+1$, $m \ge 2K+1$ — is in `TNLean/PEPS/TorusWindowComplement.lean`. The seam-wrapping band is modelled as a cyclic rectangle read through $\mathbb{Z}/n$ value distance, so the full-height complement band of width $-L$ columns is a single coordinate-translate of a rectangle and stays injective by translation invariance of the tensor; the wraparound-free restriction recovers the coordinate rectangle. The complement of each single window and consecutive-window union is delivered both at the abstract region-injectivity level and as `RegionBlockedTensorInjective` of the tensor, the consumption shape layer 2 needs.

The complement-inversion core of layer 2 is complete. The deformed-window state — the closed-torus contraction with one region's block replaced by an arbitrary insert carrying that region's boundary structure — is `deformedRegionState`, defined through the block/complement pairing of `sum_regionBlockedWeight_mul_complement`; taking the insert to be the genuine block recovers the interior-bond multiple of the closed state coefficient. The reusable engine `deformedRegionState_insert_eq_of_complementInjective` strips a closed-torus deformed-state equality to an open-boundary equality of the two inserts on the region, once the set complement is blocked-tensor injective: at a fixed region physical configuration the deformed state is the complement tensor map applied to the insert, which is injective. Both are in `TNLean/PEPS/TorusDeformedWindow.lean`. The engine is specialized to the consecutive-window unions there as `horizontalUnion_insert_eq_of_deformedState_eq` and its vertical counterpart, the union complement injectivity supplied by layer 1. The remaining part of Step 1's display — writing each single-window deformed state as the union deformed state of the corner-extended insert $C_j \sqcup T_{W_{j+1}\setminus W_j}$ — is the contraction-extension of the chaining layer (item 3 of the plan).

The geometry of the chaining and corner-stripping layer is in place. The staircase patch P, its two end windows, the corner block, and the completed corner rectangle are in `TNLean/PEPS/TorusWindowChain.lean`, with: the

disjointness of the two end windows at $2L \leq n$; the injectivity of each end window and of the completed corner rectangle (each an $L \times K$ cyclic window); the patch decomposition $P = S \sqcup$ corner at the no-wraparound bounds $2L \leq n$, $2K \leq m$; and its corollary $P \setminus S =$ corner, the single block Step 3 completes and inverts. The decomposition reduces every shifted cyclic distance to the base distance from the patch corner through the shift arithmetic `zmod_val_sub_shift`.

The tensor content of the chaining and stripping is in `TNLean/PEPS/ToricWindowChain2.lean`. The contraction-extension `deformedRegionState_extend` writes a sub-region deformed state, read as a function of the full physical configuration, as the deformed state of a superset $S \supseteq R$ with the window's insert extended by the genuine block on the added vertices $S \setminus R$. It is built from the disjoint factorization of the complement weight $T_{\mathrm{univ}\setminus R}$ over $\mathrm{univ} \setminus R = (S \setminus R) \sqcup (\mathrm{univ} \setminus S)$: with the geometry red $= R$, blue $= S \setminus R$, complement $=$ univ $\setminus S$, the landed `ThreeBlockGeometry.regionInteriorBondProd_smul_regionBlockedWeight_threeBlockComplPhysical` splits $T_{\mathrm{univ}\setminus R}$ into the blue-coupling combination of the $T_{\mathrm{univ}\setminus S}$ weights, the sum over the $\partial(\mathrm{univ} \setminus S)$ boundary configurations being exactly the boundary-configuration split. The corner-extended insert `extendInsert` contracts the window insert against the blue coupling `ThreeBlockGeometry.threeBlockBlueCoeff`, scaled by the inverse of the univ $\setminus S$ interior-bond multiplicity the factorization introduces (a nonzero scalar at positive bond dimensions, which cancels). The deformed state is read region-independently through `assembleRegion`, so the extension equalities chain by transitivity. The corner extension thus needs no standalone $\partial R \to \partial S$ boundary-configuration equivalence; the three-block blue coupling supplies the split already packaged.

With `deformedRegionState_extend` the consecutive-window display `horizontalConsecutiveWindow_extend_eq` writes each single-window state as the union state of its corner-extended insert and passes the equality to the consecutive-window engine, leaving an open-boundary equality of inserts on each consecutive-window union. The patch chaining `horizontalStaircase_patch_extend_eq` extends the two end windows' states to the patch and chains them at the *closed-torus* level, giving an equality of the two patch assembled states. The faithful Step 3 stripping `staircasePair_insert_eq_open` instead chains the patch at the open-boundary level and cancels the shared injective completed corner Q, leaving the open-boundary equality on the end pair S with no univ $\setminus S$ hypothesis (the open-boundary route below).

The route to that stripping is the open-boundary chaining, not a complement assembly. A closed-torus stripping would take the blocked-tensor injectivity of univ $\setminus S$ (the torus complement of the staircase end pair) as an explicit hypothesis. As Step 3 records, that hypothesis is *not* derivable at the corollary's minimal size: univ $\setminus S$ is not a union of injective window translates, because the end pair occupies all columns of one row and the complement band there has width below L. Completing the corner does not repair this: enlarging the compared region from S to the patch (or to the patch with the corner completed) leaves the closed-torus state unchanged, so inverting the closed state on any

superset $P' \supseteq S$ reduces to inverting univ $\setminus S$ rather than the added block $P' \setminus S$. This collapse is recorded as `deformedRegionStateAssembled_extendInsert_eq` and its inversion form `deformedRegionStateAssembled_extendInsert_eq_of_subComplementInjective` in `TNLean/PEPS/TorusWindowChain2.lean`: the superset closed-state route consumes univ $\setminus S$ injectivity and never the completed corner's. The shared-corner cancellation of Step 3 is an open-boundary operation, cancelling the injective completed rectangle `horizontalStaircaseCompletedCorner` from an equality of inserts on the completed region as tensors. The remaining work is therefore to re-derive the patch chaining of Step 2 at the open-boundary level — chaining the consecutive-window open-boundary equalities `horizontalConsecutiveWindow_extend_eq` across the patch by contraction and transitivity, rather than collapsing to the patch closed state — after which the completed corner cancels locally by its injectivity alone.

The open-boundary route factors into two engines, neither of which inverts univ $\setminus S$. They are stated here with the machinery each needs.

The corner-extension composition. The open-boundary chaining needs the transitivity of the corner extension: for nested regions $R \subseteq S \subseteq P$ and an insert C on R,

$$\texttt{extendInsert}(S \subseteq P)\,\big(\texttt{extendInsert}(R \subseteq S)\,C\big) \;=\; \texttt{extendInsert}(R \subseteq P)\,C \qquad \text{as inserts on } P.$$

Each consecutive-window open-boundary equality lives on the union U_j; extending both sides through $\texttt{extendInsert}(U_j \subseteq P)$ and collapsing the composition by this transitivity chains the equalities into one patch-level insert equality. The identity is *not* reducible to the assembled-state collapse `deformedRegionState_extend`: both sides have the same assembled state on P, but equal assembled states determine equal inserts only when univ $\setminus P$ is injective, which fails at the minimal size. Stripping the inverse interior-bond multiplicities (the lemma `extendInsert_const_smul` commutes the rescaling through), the identity reduces to the *bare* composition law

$$\texttt{bareExtendInsert}(S \subseteq P)\,\big(\texttt{bareExtendInsert}(R \subseteq S)\,C\big) \;=\; \mathrm{ribp}(\mathrm{univ}\setminus S)\cdot\texttt{bareExtendInsert}(R \subseteq P)\,C,$$

which is the blue-coupling composition of `ThreeBlockGeometry.threeBlockBlueCoeff` across the two nested geometries, glued along the univ $\setminus S$ cut. The product $\prod_{P\setminus R}$ splits as $\prod_{S\setminus R}\cdot\prod_{P\setminus S}$, and the intermediate sum over ∂S glues the two global-configuration constrained sums along their agreeing univ $\setminus S$ boundary, with fiber multiplicity ribp(univ $\setminus S$). This is a four-block (R, $S \setminus R$, $P \setminus S$, univ $\setminus P$) merge-and-count, beyond the existing three-block fiber tooling of `TNLean/PEPS/RegionBlock/UnionInjectivityGeneral.lean`. It is landed in `TNLean/PEPS/TorusWindowChain4.lean`: the merge-and-count is the two-sub-block collapse `mergeCollapse2` over univ $\setminus S$, with the two host labels carried through by the boundary-edge embeddings `regionBoundaryLabel_regionMerge_compl_of_subset` (the univ $\setminus R$ host) and `regionBoundaryLabel_regionMerge_of_subset_left` (the univ $\setminus P$ host) and the product split `regionProd_split`; the blue-coupling composition is

`threeBlockBlueCoeff_comp`, the bare law `bareExtendInsert_trans`, and the transitivity `extendInsert_trans`.

The shared-corner cancellation. With the patch equality `extendInsert`$(W_0 \subseteq P)\, C_0$ = `extendInsert`$(W_{L+K-1} \subseteq P)\, C_{L+K-1}$ in hand, the corner cancellation is the injectivity of `extendInsert`$(S \subseteq R)$ on inserts on S, where $R = S \sqcup Q$ and Q = `horizontalStaircaseCompletedCorner` is the injective completed rectangle: extending the patch equality to R (one more transitivity step through $P \subseteq R$, since $R = P \sqcup$ added row) and cancelling the shared Q-block gives the end-pair equality. The cancellation needs *only* Q injective, never univ $\setminus S$. For the nested geometry red $= S$, blue $= R \setminus S = Q$, complement $=$ univ $\setminus R$, the corner-extended insert contracts the S-insert against `ThreeBlockGeometry.threeBlockBlueCoeff`; its σ_{blue}-dependence is a combination of the Q blocked weights, so Q injectivity (the chosen left inverse `regionBlockedTensorMap_injective_of_injective` of the Q block) reads off the coupling rows, and the host-boundary decomposition $\partial(\text{univ} \setminus S) \hookrightarrow \partial Q \times \partial(\text{univ} \setminus R)$ extracts the S-insert coefficients. This is the dual of the blue-strip inversion `ThreeBlockGeometry.complCoeff_combination_eq_zero`: that lemma pushes a host annihilation through the blue block to a complement-coupling annihilation using blue injectivity; the cancellation here inverts in the host index using Q injectivity alone, the weaker hypothesis the union lemma's two-step does not isolate.

The shared-corner cancellation is formalized in `TNLean/PEPS/TorusWindowChain5.lean`. Since the corner extension is linear in its insert, the injectivity of `extendInsert`$(S \subseteq R)$ reduces to its kernel: the only insert on S whose corner extension on R vanishes is the zero insert. The linearity lemmas `extendInsert_add`, `extendInsert_sub`, and `extendInsert_const_smul` supply this reduction as `extendInsert_injective_of_kernel_trivial`. The physical assembly of $R = S \sqcup Q$ is carried by `assembleSubRegion` and its two restriction identities `restrictSubRegion_assembleSubRegion` and `restrictSubRegion_sdiff_assembleSubRegion`; these say that the S- and Q-legs of a physical configuration on R can be fixed independently. The host-boundary extraction is `ThreeBlockGeometry.regionBoundaryLabel_host_eq_hostLabelFrom`, which identifies the host residual from the Q and univ $\setminus R$ residuals.

Together with the Q-weight span `ThreeBlockGeometry.crossingBondProd_smul_threeBlockBlueCoeff_eq`, these ingredients prove `extendInsert_kernel_trivial_of_addedInjective`: a kernel vector is read through the $S \sqcup Q$ assembly, the host-residual-weighted blue coupling vanishes for every Q-leg and complement boundary, the blue block is stripped using Q injectivity, and the S-insert coefficients are recovered by host-label surjectivity at positive bond dimensions. The cancellation theorem `extendInsert_cancel_addedInjective` then cancels the injective added block from an equality of corner extensions, using only that block's injectivity and never the injectivity of univ $\setminus S$. Its staircase specialization `staircasePair_cancel` completes the end pair to $S \sqcup Q$ with Q the completed corner and cancels Q from an open-boundary equality of the two end-window inserts on the completed region.

The required open-boundary patch equality is formalized in `TNLean/PEP`

S/TorusWindowChain6.lean. The descending-arm comparison verticalConsecutiveWindow_extend_eq is the transpose of the horizontal comparison horizontalConsecutiveWindow_extend_eq: both give open-boundary equalities of corner-extended inserts on a consecutive-window union. The per-step theorem staircaseStep_patch_extend_eq lifts such a consecutive-window equality from the union U_j to the full patch P by extending both sides through extendInsert$(U_j \subseteq P)$ and using extendInsert_trans along $W_j \subseteq U_j \subseteq P$. The chained equality staircasePatch_insert_eq advances by induction from W_0 to W_{L+K-1}; the intermediate windows disappear because their deformed states are all equal to the common state.

Finally, staircasePair_insert_eq_open extends the patch equality through the completed union $S \sqcup Q$, collapses the nested extensions by extendInsert_trans, and applies staircasePair_cancel. The output is the open-boundary equality on the end pair S. Its hypotheses are the one-orientation window injectivity, the union closure, positive bonds, $2 \le L$, $2 \le K$, $2L+1 \le$ width, $2K+1 \le$ height, and the all-windows-agree input. No injectivity of univ $\setminus S$ is used. The obstruction that led to the Step 3 gap is therefore eliminated.

7 The single-bond extraction and the host-injectivity scope of the back half

This section records two findings that follow the landing of the open-boundary end-pair equality staircasePair_insert_eq_open and the single-crossing geometry of Step 4: the geometric peeling of the end-pair equality onto the single bond e, and a precise scope verdict on what host injectivity the back-half of Theorem 3 (Step 5) actually needs.

7.1 The boundary of the end pair is the windows' external legs

Step 3 now delivers the open-boundary equality on the end pair $S = W_0 \sqcup W_{L+K-1}$ without inverting univ $\setminus S$: the corner-extension composition extendInsert$(S \subseteq P) \circ$ extendInsert$(R \subseteq S) =$ extendInsert$(R \subseteq P)$ and the shared-corner cancellation staircasePair_cancel are landed, and the end-pair equality

$$\texttt{extendInsert}(W_0 \subseteq S)\, C_0 \;=\; \texttt{extendInsert}(W_{L+K-1} \subseteq S)\, C_{L+K-1} \qquad \text{as inserts on } S$$

is staircasePair_insert_eq_open, with no univ $\setminus S$ hypothesis. The single-crossing geometry of Step 4 shows e is the only lattice edge joining the two end windows, hence the only interior bond of S.

The geometric peeling of this equality onto the bond e rests on the complete characterization of the boundary ∂S. The landed direction (isRegionBoundaryEdge_endPair_of_leftWindow and its right-window

counterpart) shows a boundary edge $f \neq e$ of one window is a boundary edge of S. The converse, supplying the partition, is `isRegionBoundaryEdge\_window\_of\_endPair`: a boundary edge of S is a boundary edge of one of the two windows, read off the in-region endpoint, whose out-of-region endpoint lies outside that window since it lies outside all of S. Combined with e's interiority, this gives `isRegionBoundaryEdge\_endPair\_iff`: f is a boundary edge of S iff it is a boundary edge of one window and $f \neq e$; and `isRegionBoundaryEdge\_endPair\_exactlyOne\_window`: each boundary edge of S lies on *exactly* one window, since the only edge on both windows is e (`eq\_referenceEdge\_of\_isRegionBoundaryEdge\_both\_endWindows`), which is not a boundary edge of S. So ∂S is the disjoint union of the two windows' external legs, with e the single interior bond summed over in the end-pair contraction. This is the geometric content the bond-e coefficient identity reads: the open legs of S split into the left-window external legs and the right-window external legs, and the matrix on e is the only datum the two windows share. These facts are in `TNLean/PEPS/TorusWindowPeeling/BoundaryGeometry.lean`.

7.2 The back half transfers with a window as witness region

The torus back half of Theorem 3 consumes host injectivity in exactly two places, and at the corollary's minimal size $(2L+1) \times (2K+1)$ both are supplied by one-orientation $L \times K$ window translates — the obstruction of Step 3 (that univ $\setminus S$ is not injective at the minimal size) does not propagate to either.

The witness region. The per-edge interface `EdgeCoeffIdentityWitness` bundles a region whose boundary contains e, the second tensor's region and host blocked-tensor injectivity, and the conjugation coefficient identities on e. Its host field `hCB` : `RegionBlockedTensorInjective` B (univ $\setminus$ `region`) is host injectivity of the *witness* region, not of S. The rectangle route takes `region` to be the reference rectangle's red block, host injective only at the three-block sizes (≥ 7, or the seam-touching $+5$ = width special case). At the window route, `region` may be a single end window W: e is a boundary edge of each end window (`isRegionBoundaryEdge\_horizontalStaircaseLeftWindow\_referenceEdge` and its right counterpart), W is region injective (`horizontalStaircaseLeftWindow\_injective`), and crucially its host univ $\setminus W$ *is* injective at $(2L+1) \times (2K+1)$ — this is the landed `regionBlockedTensorInjective\_windowComplement`, whose complement decomposition (a full-height band of width $- L \geq L+1$ columns and an L-column band of height $- K \geq K+1$ rows, each an $L \times K$ window union) needs exactly $2L+1 \leq$ width, $2K+1 \leq$ height. So a single window is a valid `EdgeCoeffIdentityWitness` region with host injectivity available at the minimal size, once the peeling of §5.1 produces the bond-e coefficient identity `regionInsertedCoeff` $A\, W\, \langle e \rangle\, M$ = `regionInsertedCoeff` $B\, W\, \langle e \rangle\, (Z\, M\, Z^{-1})$ with W in place of the rectangle red block.

The comparison regions. The scalar extraction `lambda\_pow\_card\_torus\_eq\_one`

takes a single comparison region R with R and univ $\setminus R$ injective; the per-vertex relation `component\_eq\_gaugeVertex\_of\_cornerProportional` compares R against `insert` $v\,R$, needing both regions' hosts injective. The formalized R is `cornerRegion` (the 3×3 square minus its corner) and `insert` $v\,R$ = `cornerSquare` (the 3×3 square), at offset $(2, 2)$ with hardcoded $7 \le$ width, $7 \le$ height; their host decompositions (`compl\_cornerRegion\_injective`, `compl\_cornerSquare\_injective`) use both rectangle orientations (`rect23`, `rect32`, `wideRectangle`, `shortRectangle`) and width-2/width-(width $-$ 5) bands tied to the 3×3 shape. These are the $L = K = 2$ instance of the source's $(L + 1) \times (K + 1)$ comparison pair. The one-orientation replacement, as Step 5 of this note prescribes, is the $(L+1) \times (K+1)$ cyclic rectangle minus one corner vertex against the full $(L+1) \times (K+1)$ rectangle. Both are region injective with one orientation by `arcRectangle\_injective` (a cyclic rectangle of width $\ge L$, height $\ge K$ is the union of the $L \times K$ windows sliding over it). Both hosts decompose at the minimal size: by `compl\_torusArcRectangle` the complement of a $(L+1) \times (K+1)$ cyclic rectangle is a full-height band of width $-(L+1) \ge L$ columns and an $(L + 1)$-column band of height $-(K + 1) \ge K$ rows, each injective by `arcRectangle\_injective`; the band widths reach exactly L and the heights exactly K at $2L + 1$, $2K + 1$, which is why the corollary needs no larger size. The minus-corner region differs by one vertex and decomposes the same way with one extra corner rectangle, the transpose of the formalized `cornerRegion` band decomposition.

Verdict. The back half transfers to the corollary's minimal size with the window as witness region and the $(L + 1) \times (K + 1)$-shaped comparison pair, both host-decomposable into one-orientation $L \times K$ window translates at $2L+1$, $2K+1$. No host injectivity genuinely fails at the minimal size for any region the back half inverts: the sole failure, univ $\setminus S$, is confined to the Step 3 stripping, which the open-boundary route already circumvents. The capstone is therefore an *assembly* of landed machinery — the peeling of §5.1 to the bond-e coefficient identity (the one genuinely-new mechanical piece, whose geometric foundation is now landed), the window-region witness with its already-available host injectivity, and the re-anchored comparison pair built from `arcRectangle\_injective` and `compl\_torusArcRectangle` — not further research. A host-free variant of the back-half engines is *not* needed.

7.3 The bond-insertion bridge and the single-bond peeling are landed

The peeling of §5.1 onto the single bond e rests on a bridge between the two vocabularies the back half mixes: the *deformed-window* inserts the overlapping-window staircase produces, and the *region-inserted coefficient* `regionInsertedCoeff` the per-edge gauge of Step 5 reads. These were two parallel developments with no connecting lemma. The bridge is now in `TNLean/PEPS/TorusWindowPeeling/BondInsertedRegion.lean`.

The bond-inserted insert `bondInsertedRegionInsert` $A\,R\,f\,M$ on a region R is the insert whose boundary configuration ν (read by the complement

contraction) carries the M-coupled combination of the genuine R block over the boundary configurations agreeing with ν away from f. Its deformed state is exactly the region-inserted coefficient: `deformedRegionState\_bondInsertedRegionInsert` proves `deformedRegionState` $A\,R\,($`bondInsertedRegionInsert` $A\,R\,f\,M) =$ `regionInsertedCoeff` $A\,R\,f\,M$, the inner boundary sum of the insert reproducing the double boundary sum of the coefficient. Composing with the corner-extension identity `deformedRegionState\_extend` gives `regionInsertedCoeff\_eq\_extendInsert\_bondInserted`: the coefficient on R, read off a global configuration, equals the assembled deformed state on a superset S of the corner-extended bond-inserted insert. With R an end window and S the end pair, the genuine block of the *opposite* window the bond e joins to is carried by the corner extension — the geometric content the univ $\setminus W = W' \sqcup (\text{univ} \setminus S)$ split of §5.1 describes, now realized through the landed corner extension rather than a bespoke contraction split.

The single-bond peeling itself is `regionInsertedCoeff\_endWindows\_eq\_of\_staircase` in `TNLean/PEPS/TorusWindowPeeling/SingleBond.lean`. Feeding the end-pair equality `staircasePair\_insert\_eq\_open` the bond-inserted inserts on the two end windows (and arbitrary inserts on the intermediate windows), all sharing one deformed state, and bridging both sides through `regionInsertedCoeff\_eq\_extendInsert\_bondInserted`, peels the equality onto e: the two end windows' region-inserted coefficients with M and M' on e agree, read off any global configuration. The reference edge is a boundary edge of each end window by the landed single-crossing geometry (`isRegionBoundaryEdge\_staircaseWindow\_zero\_referenceEdge` and its last-window counterpart), so M and M' live on the same bond.

The window-region witness packaging of §5.2 is `windowEdgeCoeffIdentityWitness` (and its hypotheses form `windowEdgeCoeffIdentityWitness\_of\_hypotheses`) in `TNLean/PEPS/TorusWindowWitness.lean`: given the bond-e conjugation coefficient identity, every other `EdgeCoeffIdentityWitness` field with the left end window as region is discharged from the landed window geometry, the host injectivity supplied at the minimal size by `regionBlockedTensorInjective\_windowComplement`. The witness is the reference-edge interface the torus covariant-family machinery consumes.

The residual. What remains of §5.1 is the production of the bond-e *conjugation* coefficient identity the witness packaging consumes — `regionInsertedCoeff` $A\,W\,\langle e\rangle\,M =$ `regionInsertedCoeff` $B\,W\,\langle e\rangle\,(Z\,M\,Z^{-1})$. The Skolem–Noether back end of this production is now landed and reduces it to a single cross-tensor input. The reusable region-level algebra of `RegionInsertionTransfer` derives the gauge Z and the conjugation form of the coefficient identity from a *coefficient transfer* on the window: the read-off `exists\_regionEdgeGauge\_of\_coeffTransfer` turns the two cross-tensor coefficient transfers and the forward-transfer multiplicativity into Z and the conjugation identity (the lemma `exists\_regionConjCoeffIdentity\_of\_coeffTr` and feeding that to the window-region witness packaging discharges the witness's conjugation field (the lemma `exists\_windowEdgeCoeffIdentityWitness\_of\_coeffTransfer`). These are in `TNLean/PEPS/TorusWindowMult.lean`. All region-level algebra

— additivity, homogeneity, identity preservation, the two-sided inverse, the Skolem–Noether read-off — is unconditional on the window and host injectivity, both available at the minimal size, so the window witness assembles from the coefficient transfer alone, with no conjugation field assumed.

What remains is therefore exactly the staircase *coefficient transfer* on the window: for each inserted matrix M on A's edge e a matching matrix N on B with equal window coefficient (`htransferAB`), its reverse (`htransferBA`), and the multiplicativity of the chosen forward transfer (`hmul`). Two pieces feed the transfer. First, the bond-inserted *family* the single-bond peeling consumes: an insert on every staircase window all sharing one deformed state, with the two end windows carrying the bond-inserted inserts of M and M'. For the windows containing both endpoints of e the bond is interior, so the insert is the M-on-interior-bond deformation of the genuine block, and the common state is the closed torus state with M inserted on e — the "a virtual operation on a given bond is a physical operation on any window containing an endpoint" correspondence of the sketch, not yet built as an insert family. Second, the cross-tensor multiplicative transfer: the homomorphism the rectangle route reads from the physical realization of the inserted operator. The region-level recovery `regionInsertionTransfer\_of\_realizes` derives all three transfer fields — `htransferAB`, `htransferBA`, and `hmul` — from one region physical-to-virtual realization `RegionTransferRealizes` in each direction, with no further hypothesis; so the window route's residual collapses to producing that realization on the window from the staircase, in place of the edge-middle injectivity (univ $\setminus \{u, v\}$ injective) the rectangle route uses, which fails at the minimal size — the same obstruction, at the edge, that univ $\setminus S$ exhibits at the end pair. A coefficient transfer alone does not yield `hmul`: the multiplicativity is the homomorphism of the physical insertion operator, which the single-bond geometry localizes to one edge but does not on its own make multiplicative.

8 The multiplicativity verdict: it is the coarse three-site super-bond, not a single-window span

This section records the definitive resolution of the one residual question the overlapping-window corollary reduces to: whether the forward-transfer multiplicativity `hmul` at the left end window is obtainable from window and host injectivity at the corollary's minimal size, or genuinely needs a block-level construction the landed single-window machinery does not provide.

8.1 The single-window span does not exist, and is not the mechanism

The natural reading of `hmul` as a single-window fact is the question whether the reference-edge-restricted matrix family of the left end window W —

the bond matrix on e read off the window block at fixed external legs and physical configuration — spans the full $\mathrm{Fin}(\mathrm{bondDim}\, e)$ algebra, the way a one-dimensional length-L window product spans the full $D \times D$ algebra and feeds the homomorphism extraction `exists\_insertionHom` through `LinearMap.ext\_on\_range`. It does not. A one-dimensional window has exactly two virtual legs, both of dimension D, so its product *is* a square matrix on the bond and window injectivity is bond-algebra spanning directly. A two-dimensional $L \times K$ end window touches e with *one* endpoint, so its e-restricted family is a family of *vectors* on $\mathrm{Fin}(\mathrm{bondDim}\, e)$, and the window's whole perimeter is open. Folding the perimeter into a second $\mathrm{Fin}(\mathrm{bondDim}\, e)$ index would require $\prod_{\text{external legs}} \mathrm{bondDim} = \mathrm{bondDim}\, e$, false in general; the full boundary-configuration family is linearly independent (window block injectivity), but its projection onto the e-leg at fixed external legs is not a square-matrix span. So the one-dimensional template port has no two-dimensional single-window analogue, and the homomorphism cannot be read from one window's block.

8.2 The faithful mechanism is the coarse super-bond, available without single-vertex injectivity, but obstructed for the single-window red block at the minimal size

The source supplies `hmul` not from a single window but from the homomorphism of the physical insertion operator, which the formalization realizes through the *coarse three-site* route already landed for the rectangle blocking: three blocked regions (a red block, a blue block meeting it only at e, and the complement) are assembled into a coarse tensor on a three-vertex graph, whose three super-sites are injective directly from the three blocked-region injectivities, with no single-vertex injectivity of the original tensor. The coarse tensor is edge-blocked three-site injective at the distinguished super-bond, so the single-vertex transfer-matrix multiplicativity `edgeTransferMatrix\_mul` applies *at the super-bond*; conjugated by the two bond models this is `regionEdgeTransfer\_mul`, and through the identification of the chosen forward transfer with the concrete one (`coeffTransferMap\_eq\_regionEdgeTransfer`, by `regionInsertedCoeff\_injective`) it is exactly `hmul` for the chosen transfer over the red block. The super-vertex injectivity stands in for the single-vertex spanning the vertex-injective route would use; this is why the route needs only blocked-region injectivity of the three regions.

This mechanism applies to the window route iff the left end window W can play the coarse red block, which requires a blue block making e the single red-to-blue crossing and a complement univ $\setminus$ ($W \cup$ blue) injective. The single-crossing requirement is met canonically: the right end window W' is the only block joined to W by exactly the reference edge e (`isCrossingEdge\_horizontalStaircaseEndWindows`). But that forces $W \cup \mathrm{blue} = W \cup W' = S$, the staircase end pair, so the complement is univ $\setminus\, S$

— the very region Step 3 proves is *not* injective at the corollary's minimal size, because S occupies all columns of one row and the complement band there has width below L. So the coarse three-site frame with red $= W$ and the natural single-crossing partner is unbuildable at the minimal size: its `complement\_injective` field is exactly the obstructed univ $\setminus S$ injectivity, the same obstruction the open-boundary staircase route was built to circumvent at the *state-equality* level, now reappearing at the *multiplicativity* level. Enlarging the blue block beyond W' cannot repair this within the single-window red frame: a blue block making e the sole red-to-blue crossing must contain e's out-of-W endpoint and avoid creating any second crossing of ∂W, and any such enlargement only shrinks univ $\setminus$ ($W \cup$ blue), making its injectivity no easier than for univ $\setminus S$. The faithful repair is not a larger partner but a different mechanism, below.

8.3 Verdict

The verdict is **(b)**: the e-bond multiplicativity at the minimal size is *not* derivable from window and host injectivity of the single left end window. It is the coarse three-site super-bond multiplicativity, which needs a third injective block — the complement of the red-blue pair — and at the minimal size, with the red block a single window and the single-crossing partner forced to be the opposite window, that complement is univ $\setminus S$, non-injective. The read-off side (`htransferAB`, `htransferBA`, and the whole region-level gauge algebra) *is* powered by window and single-window-complement injectivity, both available at the minimal size; only the homomorphism is not.

The block-level construction the staircase must supply is therefore precisely a coarse blocking frame around e whose three regions are all injective at the minimal size and whose red-to-blue crossing is the singleton $\{e\}$ — equivalently, a coefficient transfer on the window that is *independently* known to be multiplicative, since the multiplicativity is the homomorphism of the physical insertion operator and no single-window read-off makes it one. The source's sketched overlapping-window realization is that the same virtual operation M on e is a physical operation on *every* window of the staircase chain, all sharing one deformed state; the chain of consecutive-window inversions (landed at the state-equality level as `staircasePair\_insert\_eq\_open`, never inverting univ $\setminus S$) transports M around the corner, and the homomorphism $M \cdot M' \mapsto$ (transfer of M) (transfer of M') is read from the composition of two such transports across the chain, not from any one window's block. Building that insert-family transport — the "a virtual operation on a given bond is a physical operation on any window containing an endpoint" correspondence as an actual insert family with the composition law — is the residual, and it is a *block-level* (insert-family-and-chain) construction, not a single-window span.

8.4 The foundation that lands

The reusable multiplicativity is exposed as `TNLean.PEPS.coeffTransferMap_mul_of_coherentFrames` (`TNLean/PEPS/RegionBlock/CoarseThreeSiteMul.lean`): two coherent coarse blocking frames over the two tensors sharing the three regions, the bond dimensions, and the single-crossing edge make the chosen forward transfer over the red block multiplicative, with no single-vertex injectivity. This is the `hmul` field extracted from the bundled gauge assembly `TNLean.PEPS.exists_regionEdgeGauge_of_coherentFrames`, so a consumer holding only the coefficient transfer can obtain the multiplicativity from a frame. It makes the verdict auditable in Lean: the window route's `hmul` is one application of this theorem with red $= W$, and the unmet hypothesis is exactly the frame's `complement\_injective` at univ $\setminus S$, the documented obstruction. The remaining work is the insert-family transport that produces a multiplicative window coefficient transfer without a single-window red-block frame.

8.5 The per-step bond-insert transport, and where the chain composition walls

The insert-family transport called for above factors into per-step transports between *consecutive* windows, and the single load-bearing per-step transport is now built: `TNLean.PEPS.horizontalStaircaseConsecutiveWindow_bondTransport_extend_eq` (`TNLean/PEPS/TorusWindowBondTransport.lean`). For a sliding-arm index $j < L$, given the consecutive windows W_j and W_{j+1} sharing a bond g as a boundary edge and inserted matrices M_1, M_2 on g whose orientations through the two windows agree, the corner-extended bond inserts on the injective overlap $U_j = W_j \cup W_{j+1}$ are equal. The window-independence of the bond insertion (`TNLean.PEPS.bondInserted_windowIndependent`) makes the two inserts' deformed states cross-proportional — the interior-bond product of one window times the state of the other matches the swapped product — and rescaling each by the *other* window's interior-bond product turns that cross-proportionality into an honest equality of states, which the consecutive-window comparison (`TNLean.PEPS.horizontalStaircaseConsecutiveWindow_extend_eq`) strips to the open-boundary equality on U_j by inverting the injective $(L+1) \times K$ rectangle, never univ $\setminus S$. This is the genuine per-step transport through the injective overlap.

Two facts pin where the chain composition stops short of the multiplicativity. First, the *telescoping engine itself exists*: the patch chaining `TNLean.PEPS.staircasePatch_insert_eq` already composes per-step equalities into one comparison of the two end windows on the patch, and the end-pair peeling `TNLean.PEPS.regionInsertedCoeff_endWindows_eq_of_staircase` reads that off as the single-tensor relation `regionInsertedCoeff`$(W_0, M) =$ `regionInsertedCoeff`(W_{L+K-1}, M'). So the chain composition delivers the *single-operation* window-independence, end to end, with the injective overlaps mediating.

Second, that engine consumes a family of inserts on *all* $L + K$ windows

sharing one common deformed state, with the two end inserts the bond inserts of M and M'. The per-step transport does not assemble such a family. Two sub-obstructions sit here, and they have different status under the corollary's hypotheses.

The interior-bond rescaling is uniform under the corollary's translation invariance. The per-step transport agrees the two scaled inserts only up to the *neighbour-dependent* interior-bond rescaling: step j scales by $\mathrm{ribp}(W_{j+1})$ on the left and $\mathrm{ribp}(W_j)$ on the right, and a telescoping to one common state needs all the interior-bond products equal. The corollary is the strengthening of the *torus* Theorem 3, whose tensors are translation invariant, and the two end windows are diagonally offset $L \times K$ translates of one another — indeed every staircase window is a translate of every other. So the interior-bond products are all equal: `TNLean.PEPS.regionInteriorBondProd_staircaseWindow_eq` (`TNLean/PEPS/TorusWindowBondUniform.lean`) derives this from the bond-dimension invariance of a translation-invariant tensor under translation (`TNLean.PEPS.bondDim_translateEdge_of_translationInvariant`) reindexing the interior-bond product (`TNLean.PEPS.regionInteriorBondProd_map`). With the products uniform the per-step rescalings agree across the chain, and this sub-obstruction — the one the normal-PEPS *plane* setting could not dissolve — is removed by the corollary's translation invariance.

The cross-bond pushing on the intermediate windows is the genuine residual. The bond e is a boundary edge of only the two end windows; the intermediate windows touch neither endpoint of e, so a bond insert of M on them is not even typed. Consecutive windows W_j, W_{j+1} share a bond g_j *other* than e, and the per-step transport relates a bond insert on g_j in W_j to one on the *same* g_j in W_{j+1}. But a window W_j shares the bond g_{j-1} with its left neighbour and the *different* bond g_j with its right neighbour, so chaining the two incident per-step transports through one insert C_j requires that C_j realize the operation on g_{j-1} and on g_j simultaneously — i.e. that the closed-state deformation $\mathrm{edgeInsertedCoeff}(g_{j-1}, M)$ equals $\mathrm{edgeInsertedCoeff}(g_j, M')$ for a matched M'. That is the two-dimensional bond-operator pushing: inverting the injective window W_j to read M's physical action off one boundary bond and re-express it on another. The deformed window tensor realizing this is the residual the per-step transport does not supply.

This is exactly the unbuilt construction: the two-dimensional bond-operator extraction that builds, from one virtual operation M on e, the common-state family of deformed window tensors on all $L + K$ windows — the matrix-tensor analogue of `MPSChainTensor.overlapWindow_exists_bondOperator`. Its missing ingredient is the window-inversion that pushes the operation across distinct boundary bonds, and that inversion reads M's action off the in-region endpoint of the boundary bond, which is the endpoint spanning `TNLean.PEPS.span_stateOpenCoeff_eq_top` — available from *single-vertex* injectivity, not from region injectivity. With that family the end-pair peeling fires and the homomorphism $M \cdot M' \mapsto$ (transfer of M)(transfer of M') would be read from composing two transports as in `MPSChainTensor.exists_insertionHom`, whose multiplicativity step (`eq\_of\_span\_mul\_left`) consumes the same

window-product span. The per-step transport and the now-uniform rescaling are the foundation of that extraction; the extraction itself — the deformed-window-tensor family on the intermediate windows, requiring the endpoint span — is the unbuilt block-level construction, and it is the same endpoint-spanning residual the coarse three-site route hits as the non-injective univ $\setminus S$ at the minimal size. The translation invariance of the corollary removes the rescaling sub-obstruction but not this one.

9 Faithfulness verdict: normality is not a dropped hypothesis, and the obstruction is the source's own block-level pushing

This section answers a hypothesis-faithfulness question raised against the campaign: the source corollary assumes *normal* 2D PEPS tensors *such that every* $L \times K$ *region is injective*, and a reading of the campaign suspected that "normal" is a second, stronger hypothesis the campaign's window-injectivity bundle dropped — one that would supply the single-vertex endpoint spanning the cross-bond pushing wants, and so unblock the corollary. The verdict is that this reading is incorrect on the mathematics. Normality is not a separate stronger hypothesis here; the campaign faithfully captured the source's hypothesis set; and the obstruction recorded above is genuine, not an artifact of a weakened hypothesis.

9.1 What "normal" means in the source, and why it is not stronger than region injectivity

The source defines normality as a region-blocking condition.[18] A tensor (or region) is injective in the source's sense when, "interpreted as a map from the virtual space to the physical one," it is injective.[19] The corollary's clause "every $L\times K$ region is injective" names the "certain regions" of the normality definition: the $L\times K$ rectangles are the blocking regions that witness normality. So "normal ... such that every $L \times K$ region is injective" is not normality *plus* a region condition; the region condition *is* the operative content of normality for this corollary, exactly as the one-dimensional companion reads "normal ... with the property that blocking L consecutive sites results in an injective tensor"[20] — there "blocking L sites is injective" is the witness and nothing single-site is assumed.

Crucially, normality is by definition *weaker* than single-vertex injectivity, not stronger. A normal tensor is precisely one that need *not* be injective at a single

[18] `Papers/1804.04964/paper_normal.tex:1318`: "We call a PEPS *normal*, if blocking tensors in certain regions results in injective tensors."

[19] `Papers/1804.04964/paper_normal.tex:980`.

[20] `Papers/1804.04964/paper_normal.tex:2258`; the same phrasing recurs at the $n \geq 3L$ corollary, `Papers/1804.04964/paper_normal.tex:1664`.

site but *becomes* injective after blocking; the canonical normal-but-not-injective examples (a single site whose physical dimension is below its bond dimension) have linearly dependent single-site components and so fail `TNLean.PEPS.IsVertexInjective`. Therefore normality cannot supply the single-vertex endpoint spanning `TNLean.PEPS.span_stateOpenCoeff_eq_top` (`TNLean/PEPS/RegionBlock/Recovery3.lean`), whose hypothesis is `TNLean.PEPS.IsVertexInjective` B at the in-region endpoint, derived from linear independence of the single-vertex complement family (`vertexComplementTensorInjective_of_isVertexInjective`). The hypothesised mechanism — "normality gives the single-vertex span region injectivity lacks" — asks normality to deliver a property strictly above itself.

9.2 The campaign captured the source hypothesis faithfully

The campaign's bundle `TNLean.PEPS.NormalTorusArcWindowInjectivityHypotheses` (`TNLean/PEPS/TorusWindowComplement.lean`, line 180) carries a single field, `arcWindow_injective`: every cyclic $L \times K$ window is injective, where window injectivity is `TNLean.PEPS.RegionBlockedTensorInjective` (`TNLean/PEPS/RegionBlock/Basic.lean`, line 118), the linear independence of the $L \times K$-blocked tensor family read as a map from the window's boundary configuration to its physical configuration. This is the source's "$L \times K$ region is injective" verbatim, and it is the *whole* of the source's normality hypothesis for the corollary. The bundle adds no single-vertex field and drops no normality field: there is no source field to drop, because the source never assumes single-vertex injectivity. The faithfulness rule is satisfied — the Lean hypothesis set neither exceeds nor falls short of the source's.

A blocked $L \times K$ window read as one super-vertex has *its* component map equal to `regionBlockedTensorFamily`, so window injectivity is the super-vertex injectivity of the coarse tensor. The coarse three-site machinery already exploits exactly this: `TNLean.PEPS.coeffTransferMap_mul_of_coherentFrames` (`TNLean/PEPS/RegionBlock/CoarseThreeSiteMul.lean`) derives the forward-transfer multiplicativity from three blocked-region injectivities *with no single-vertex injectivity*, the super-vertex injectivity standing in for the single-vertex spanning. So the campaign does have a coarse substitute for the endpoint span; the question is only whether it is available for the regions the corollary's cross-bond pushing needs, at the minimal size.

9.3 Why the coarse substitute does not reach the intermediate windows, and why the one-dimensional precedent does not port

The cross-bond pushing on an intermediate window W_j must read the inserted operation off the in-region endpoint of one boundary bond g_{j-1} and re-express it on a *different* boundary bond g_j of the same window. Blocking W_j to one

super-vertex does make g_{j-1} and g_j two legs of one injective super-vertex, and the super-vertex injectivity `RegionBlockedTensorInjective` W_j is in hand. But that injectivity is linear independence across the window's *whole multi-bond perimeter*, not a single-bond matrix-algebra span between g_{j-1} and g_j. The operation pushed across a single bond pair is a square matrix on $\mathrm{Fin}(\mathrm{bondDim}\, g)$; reading it off requires the family of window blocks, restricted to the two bonds g_{j-1}, g_j at fixed remaining external legs, to span the full $\mathrm{Mat}_{\mathrm{bondDim}\, g}$. That single-bond-pair span is *strictly stronger* than window injectivity and does not follow from it, for the same dimension-count reason the per-window square-matrix family fails to exist (the section on the multiplicativity verdict): folding the perimeter into a second $\mathrm{Fin}(\mathrm{bondDim}\, g)$ index would force $\prod_{\text{other external legs}} \mathrm{bondDim} = \mathrm{bondDim}\, g$, false in general.

This is exactly where the one-dimensional precedent fails to port, and the reason is geometric, not a missing hypothesis. The formalized one-dimensional overlapping-window extraction `MPSChainTensor.overlapWindow_exists_bondOperator` (`TNLean/PEPS/CycleMPSChainOverlapWindow.lean`, line 356) consumes only `IsWindowInjective` $A\ L$ — window (region) injectivity, never single-site injectivity — and *succeeds*. It succeeds because a length-L one-dimensional window has exactly two virtual legs, a left bond and a right bond, both of dimension D, so the window product *is* a $D \times D$ matrix and window injectivity is precisely the statement that these products span the full matrix algebra at the single bond; the cross-bond pushing is then the intertwining-span read-off `exists_bondOperator_of_intertwine_span` (same file, line 74), and the homomorphism is `MPSChainTensor.exists_insertionHom` fed by the same window span. The two-dimensional $L \times K$ window touches the reference bond with *one* endpoint, so its bond-restricted family is a family of *vectors*, never a square-matrix span, and the window's whole perimeter is open. The one-dimensional mechanism that powered the precedent *is* window-injectivity-only — the precedent never needed single-site injectivity — but its enabling fact, "window product is a square matrix on the pushed bond," is a property of the square one-dimensional window geometry that the non-square $L \times K$ window does not have. The campaign's analysis that "the two-dimensional analogue does not exist" is therefore correct and not a mis-analysis: it is the absence of the square-window geometry, not the absence of a normality hypothesis.

9.4 Verdict and the precise route

The verdict, sharpened by the hypothesis audit, is that the obstruction is not removable by adding a hypothesis: *the source genuinely needs region injectivity only, the campaign captured that hypothesis faithfully, and the obstruction is real and intrinsic to the source's own block-level pushing.* Normality is not the dropped hypothesis; there is nothing to add. The corollary's hypotheses are faithfully captured and admit no counterexample, but the source's sketch is *not* a rigorous proof: as Part 1 records, it asserts the cross-bond pushing by a Lemma 5 analogy that fails dimensionally. That pushing is a *block-level* construction, the two-dimensional analogue of the one-dimensional window inversion, and it is

not a single-window span nor a single-vertex span. At the corollary's minimal size the natural coarse frame carrying it (red block the left end window, single-crossing partner the right end window) has third block univ $\setminus S$, which is not injective; this is the documented wall, and it is a wall of the *construction*, not of the hypotheses.

The route to the corollary is therefore the one this note already prescribes, and it does *not* involve adding a normality or single-vertex hypothesis. It is the deformed-window-tensor family on all $L+K$ windows sharing one common state, built so that each intermediate window W_j carries the simultaneous realization of the operation on its two incident boundary bonds g_{j-1} and g_j — the two-dimensional bond-operator extraction, the matrix-tensor analogue of `MPSChainTensor.overlapWindow_exists_bondOperator`. Its single missing ingredient is the window-inversion that pushes one operation across two distinct boundary bonds of one window. That ingredient is supplied not by single-vertex injectivity and not by window injectivity alone, but by a coarse blocking frame around *each intermediate bond crossing* whose three regions are injective at the minimal size and whose single red-to-blue crossing is the relevant bond — the same coarse-three-site mechanism (`coeffTransferMap_mul_of_coherentFrames`) the back half already uses, applied per intermediate bond rather than at the end-pair crossing. The research content that remains is whether such a per-bond coarse frame exists at the minimal size for the intermediate bonds (where the partner block is a *neighbouring* window, not the diagonally opposite one), or whether those crossings inherit the same univ $\setminus S$-type non-injectivity. The per-step transport `TNLean.PEPS.horizontalStaircaseConsecutiveWindow_bondTransport_extend_eq` (`TNLean/PEPS/TorusWindowBondTransport.lean`) and the now-uniform interior-bond rescaling (`TNLean.PEPS.regionInteriorBondProd_staircaseWindow_eq`) are the landed foundation; the per-bond coarse frame for the simultaneous two-bond realization is the unbuilt block-level construction. No hypothesis change to the capstone is warranted by this audit.

9.5 The per-bond coarse frame does not exist at the minimal size, for any single-crossing window pair

The open research content the previous subsection isolated — whether a per-bond coarse frame exists at the minimal size for an intermediate bond, where the partner block is a neighbouring window rather than the diagonally opposite one, or whether those crossings inherit the same univ $\setminus S$-type non-injectivity — is now settled, in the negative, by the geometry of the single-crossing partition. The settlement is purely combinatorial and applies to *every* single-crossing $L\times K$ window pair, intermediate or end-pair.

A coarse frame around a bond e feeds the multiplicativity engine `TNLean.PEPS.coeffTransferMap_mul_of_coherentFrames` only through a partition of the lattice into three injective regions, red, blue, and the complement of their union, with e the single red-to-blue crossing. The third region is forced to be univ $\setminus$ (red $\cup$ blue). The only landed source of complement injectivity at the minimal sizes $2L+1 \le$ width, $2K+1 \le$ height is the two-band decomposition

of the complement of a *single cyclic rectangle* (`TNLean.PEPS.NormalTorusArcWindowInjectivityHypotheses.windowComplement_injective` and its consecutive-union counterparts): the complement of a $w \times h$ cyclic rectangle splits into a full-height band of width $-\,w$ columns and a $w \times (\text{height} - h)$ leftover, both injective when $w \le \text{width} - L$ and $h \le \text{height} - K$. Injectivity of the third region by this machinery therefore requires red $\cup$ blue to be a single cyclic rectangle.

But two $L \times K$ windows cross at a single lattice edge only when they are the diagonally offset staircase pair: column-adjacent across one shared row (`TNLean.PEPS.isCrossingEdge_horizontalStaircase`). Their union then covers all $2L$ contiguous columns at the shared row (`TNLean.PEPS.horizontalStaircasePair_row_span`) while the rows above and below it carry only one window's L columns, so the union is a staircase, not a rectangle. The product structure of a cyclic rectangle — membership is a column condition and a row condition read independently — forces the swapped corner $(a + 2L - 1,\, b)$ to belong whenever the three corners $(a,\, b+K-1)$, $(a+2L-1,\, b+K-1)$, $(a,\, b)$ do; the swapped corner lies outside the union, so the union is not any cyclic rectangle (`TNLean.PEPS.horizontalStaircasePair_ne_arcRectangle`). At the minimal width the obstruction is quantitative as well: the complement band at the shared row is width $-\,2L < L$ columns wide (`TNLean.PEPS.horizontalStaircasePair_complement_band_lt_window`), so no $L \times K$ window translate passing through that row avoids the union (`TNLean.PEPS.horizontalStaircasePair_window_meets_row`), and the complement is not a union of injective window translates.

This is the precise conflict, and it is a joint unsatisfiability of two of the frame's conditions: the single-crossing requirement forces the diagonal offset, the diagonal offset forces the double-width row span, and the double-width row span forces the union out of the cyclic-rectangle class, which is the only class whose complement the landed machinery proves injective. Choosing a *neighbouring* window as the partner does not escape this: a window sharing more than the single edge e with red is not a single-crossing partner, and a single-crossing partner is, up to translation, the diagonal offset itself. Enlarging red or blue beyond a single window to make the union a rectangle does not help either, since a cyclic rectangle admits no bipartition into two regions each injective (each containing an $L \times K$ window) with a single edge between them: the rectangle is two-edge-connected, so any bipartition into two non-degenerate parts has at least two crossing edges. The per-bond coarse frame is therefore unbuildable at the minimal size for every bond of the chain, which confirms verdict **(b)** and redirects the residual to the insert-family chain transport of the previous subsection rather than to a per-bond coarse frame. The geometry is in `TNLean/PEPS/TorusWindowSingleCrossingObstruction.lean`.

References

[MGRPG+18] András Molnár, José Garre-Rubio, David Pérez-García, Norbert Schuch, and J. Ignacio Cirac. Normal projected entangled pair states generating the same state. *New Journal of Physics*, 20(11):113017, 2018.