Common Sector Relabeling as an Unconditional Identity

2026-05-10

Verdict: clarification (resolved).

1 The assertion

For a blocking period p, the physical index set of the blocked tensor is canonically identified with words of length p:

$$\mathrm{Fin}(d^p) \simeq (\mathrm{Fin}\, p \to \mathrm{Fin}\, d).$$

The cyclic-sector construction uses the canonical multi-index relabeling blockwise. Write $X \sim_{\mathrm{MPV}} Y$ when two tensors generate the same matrix product vector family at every length and every word. Write $R(A_k^{[p]})$ for the length-p blocked tensor $A_k^{[p]}$ read through this canonical word relabeling.

The mathematical assertion is that assembling the weighted block-diagonal tensor before this relabeling gives the same MPV family as assembling after the relabeling:

$$\bigoplus_k \mu_k^p \cdot A_k^{[p]} \sim_{\mathrm{MPV}} \bigoplus_k \mu_k^p \cdot R(A_k^{[p]}).$$

Here $A_k^{[p]}$ denotes the length-p blocking of the k-th primitive block.

2 Relation to the paper

The paper arXiv:1606.00608 does not state this as a separate hypothesis. In the source argument the blocked-word identification is an automatic consequence of the canonical blocking construction: blocking a tensor and then reading the result through the explicit multi-index grouping of $\{1, \ldots, d\}^p$ is definitionally the same operation. The paper therefore treats the equality as transparent and does not name it.

3 Status

No deviation from the source paper is present. The relabeling assertion is the identity between two descriptions of the same blocked-word coordinates. The formalization uses the unconditional nonzero-part identity for a common cyclic-sector family directly, without introducing a separate hypothesis.[1]

4 Verdict

The relabeling identity is therefore a formal expression of the paper's implicit blocked-index identification, not an additional mathematical assumption.

References

[1] The family-level statement is `CommonBlockedCyclicSectorFamily.reindexed_nonzero_part`; the common primitive-block theorem applies it directly on both sides. The declarations are in `TNLean/MPS/CanonicalForm/SectorComparison/CommonBlockedCyclicSectorFamily.lean` and `TNLean/MPS/CanonicalForm/SectorComparison/CommonSectorTransport.lean`.