ChatGPT

Residual Connections and Identity Mappings

A standard residual block takes an input vector $x_\ell \in \mathbb{R}^C$, applies some transformation $F(\cdot; W_\ell)$ (e.g. a feed-forward or attention module with parameters W_ℓ), and then *adds* the input back to the output. In one layer this is written as:

$$x_{\ell+1} \;=\; x_\ell \;+\; F(x_\ell; W_\ell) \quad (\text{Eq.1}).$$

Recursively applying Eq.1 over layers ℓ through L yields

$$x_L \;=\; x_\ell \;+\; \sum_{i=\ell}^{L-1} F(x_i; W_i) \quad (\text{Eq.2}),$$

so the original input x_ℓ is carried all the way to layer L by the *identity path* (the unmodified "skip connection") [1]. This identity mapping is crucial: it ensures a portion of the signal (and gradient) flows unchanged through arbitrarily many layers, which stabilizes training and prevents vanishing/exploding gradients [2].

> **Key Insight:** In a deep residual network, each layer's skip connection enforces $x_{\ell+1} = x_\ell + F(x_\ell)$. Iterating this shows $x_L = x_\ell + \sum F(\cdot)$ [1], so the "identity" component x_ℓ is passed through exactly. This perfect identity skip path preserves signal norm and eases gradient flow, enabling very deep networks to train stably.

Hyper-Connections: Widening the Residual Highway

Hyper-Connections (HC) generalize the single-stream residual by expanding it into n parallel streams. Concretely, instead of a single vector $x_\ell \in \mathbb{R}^C$, we create an **expanded residual state** $\widetilde{x}_\ell \in \mathbb{R}^{n \times C}$ consisting of n copies (or slots) of the input:

$$\widetilde{x}_\ell = \begin{pmatrix} x_\ell \\ x_\ell \\ \vdots \\ x_\ell \end{pmatrix} \;\in\; \mathbb{R}^{n \times C}\,.$$

This can be viewed as an "n-stream residual" [3], increasing the total width of the skip path from C to nC. To let these streams interact, HC introduces **trainable mixing matrices** that read from, write to, and mix these streams. In layer ℓ, define three matrices: - $W_\ell^{(\text{in})} \in \mathbb{R}^{1 \times n}$ (called H_ℓ^{pre} in the paper) aggregates the n streams into one vector (a weighted sum across streams).
- $W_\ell^{(\text{out})} \in \mathbb{R}^{1 \times n}$ (called H_ℓ^{post}) distributes a single output back into the n streams.
- $W_\ell^{(r)} \in \mathbb{R}^{n \times n}$ (called H_ℓ^{res}) **mixes features within the n-stream residual**.

With these, the forward pass of one HC layer is:

1. **Read-in:** Compute $u_\ell = W_\ell^{(\text{in})}\, \widetilde{x}_\ell \in \mathbb{R}^{1\times C}$. This collapses the n-stream into one C-dimensional input for the layer.
2. **Residual Function:** Apply the layer's transformation F on u_ℓ: $v_\ell = F(u_\ell; W_\ell) \in \mathbb{R}^{1\times C}$.
3. **Write-out:** Distribute the result back into n streams: $\widetilde{v}_\ell = (W_\ell^{(\text{out})})^\top\, v_\ell \in \mathbb{R}^{n\times C}$.
4. **Skip Mixing:** Simultaneously mix the original residual streams: $\tilde{s}_\ell = W_\ell^{(r)}\, \widetilde{x}_\ell \in \mathbb{R}^{n\times C}$.
5. **Combine:** The new residual state is

$$\widetilde{x}_{\ell+1} \;=\; \tilde{s}_\ell \;+\; \widetilde{v}_\ell \;\in\; \mathbb{R}^{n\times C}.$$

In short, each layer now has an "n-stream" identity skip (mixed by $W_\ell^{(r)}$) plus the layer output inserted into each stream via $W_\ell^{(\text{out})}$. Equivalently, as stated in the paper:

$$x_{\ell+1} \;=\; W_\ell^{(r)}\, x_\ell \;+\; (W_\ell^{(\text{out})})^\top\, F\big(W_\ell^{(\text{in})} x_\ell; W_\ell\big),$$

where now $x_\ell \in \mathbb{R}^{n\times C}$ is the expanded residual (Eq.3) [4]. This architecture has higher "bandwidth": multiple streams can carry different features in parallel, and the trainable matrices let streams interact. Importantly, this does **not** significantly increase FLOPs, only the memory bandwidth of the skip path [5] [4].

> **Idea:** Hyper-Connections split the single residual into n parallel lanes. The matrices $W_\ell^{(\text{in})}, W_\ell^{(r)}, W_\ell^{(\text{out})}$ govern how information flows between these lanes: one collapses lanes into the layer input, one mixes lanes among themselves, and one distributes the layer output back into the lanes [6] [4].

Instability of Unconstrained Hyper-Connections

While Hyper-Connections (HC) add capacity, they destroy the strict identity property of ResNets. In a standard residual block, the skip part is *exactly* identity. But in HC, the skip is multiplied by a learned matrix $W_\ell^{(r)}$. Over many layers, the **composite skip mapping** becomes

$$\widetilde{x}_L = \Big(\prod_{i=\ell}^{L-1} W_i^{(r)}\Big)\, \widetilde{x}_\ell \;+\; \sum_{i=\ell}^{L-1}\Big(\prod_{j=i+1}^{L-1} W_j^{(r)}\Big)\, (W_i^{(\text{out})})^\top F(W_i^{(\text{in})}\widetilde{x}_i; W_i).$$

This is Eq.(4) in the paper [7] [8]. If each $W_\ell^{(r)}$ were the identity, the first term would just be $\widetilde{x}_\ell$ (as in a normal ResNet). But arbitrary learned $W_\ell^{(r)}$ **no longer preserve the identity**. In practice this means small deviations compound: the signal norm can **explode or vanish** as it passes through many layers [9]. Indeed, the authors observe that in an unconstrained HC model, the effective "gain" along the skip path can reach thousands (e.g. ≈3000× amplification) [9] [10], causing gradient NaNs and training collapse.

Analytically, unconstrained $W_\ell^{(r)}$ can have a spectral norm (largest singular value) much greater than 1. Repeated multiplication $\prod W_\ell^{(r)}$ then tends to blow up or shrink any input. The paper quantifies this by looking at row/column sums of the composite mapping, finding enormous deviations from 1 [11] [10]. In short, unconstrained HC breaks the "conservation of signal" that identity skips provided, leading to **numerical instability**.

> **Key Insight:** In vanilla HC, the product of learned skip-matrices $\prod_i W_i^{(r)}$ quickly drifts away from the identity. Small mis-scalings compound, so forward activations or backward gradients **explode** (or vanish) instead of propagating cleanly [9]. This violates the ResNet's core identity behavior.

Doubly-Stochastic Constraints (mHC)

Manifold-constrained HC (mHC) fixes this by **constraining each skip-matrix to be doubly stochastic**. A matrix $M \in \mathbb{R}^{n \times n}$ is *doubly stochastic* if all entries are nonnegative and each row and each column sums to 1. Equivalently:

$$M\,\mathbf{1}_n = \mathbf{1}_n, \quad \mathbf{1}_n^\top M = \mathbf{1}_n^\top, \quad M_{ij} \geq 0,$$

where $\mathbf{1}_n \in \mathbb{R}^n$ is the all-ones vector [12]. In mHC, each $W_\ell^{(r)}$ is forced onto this "Birkhoff polytope" of doubly-stochastic matrices [12]. When $n = 1$ this simply forces the scalar to be 1, exactly recovering the original identity skip [13].

Doubly-stochastic constraints endow the skip-mappings with powerful properties: - **Norm (Spectral) Control:** Any doubly-stochastic M has spectral norm $\|M\|_2 \leq 1$ [14]. In other words, M is non-expansive, so $\|Mv\| \leq \|v\|$ for any v. This guarantees **no amplification of the signal or gradient norm** when multiplying by M, preventing explosions [14].
- **Closure under Composition:** The set of doubly-stochastic matrices is closed under multiplication [15]. Thus the product $\prod_i W_i^{(r)}$ remains doubly-stochastic, so the *entire skip path* still conserves norm and mean as it propagates through many layers [15].
- **Convex Permutation Interpretation:** The Birkhoff polytope is exactly the convex hull of permutation matrices [16]. Therefore each $W_\ell^{(r)}$ can be viewed as a weighted mixture of permutations of the streams. Intuitively, each stream at the output is a convex combination of the inputs. This enforces a form of "mass conservation": the average of the stream entries is preserved, and no single stream can dominate or vanish.

> **Key Insight:** Constraining each skip-matrix $W_\ell^{(r)}$ to be doubly-stochastic restores identity-like behavior. Such an M satisfies $M\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^\top M = \mathbf{1}^\top$ [12], so the *mean activation* is exactly preserved. Moreover $\|M\|_2 \leq 1$ [14], so signals never grow. In fact, multiplying any vector by $W_\ell^{(r)}$ simply redistributes its entries without changing the total "mass" [14] [17].

Projecting onto the Birkhoff Polytope via Sinkhorn–Knopp

To enforce doubly-stochasticity in practice, mHC parameterizes an unconstrained matrix $\widetilde{W}_\ell^{(r)} \in \mathbb{R}^{n \times n}$ and then applies the **Sinkhorn–Knopp algorithm** to project it onto the Birkhoff polytope. Concretely, the forward pass computes:

$$W_\ell^{(r)} \;=\; \mathrm{Sinkhorn}(\widetilde{W}_\ell^{(r)})\,,$$

where Sinkhorn$(\cdot)$ iteratively normalizes rows and columns. One implementation is:

$$M^{(0)} = \exp(\widetilde{W}_\ell^{(r)}) \quad (\text{make all entries positive}),$$

and then for $t = 1, 2, \ldots, T$:

$$M^{(t)} \;=\; \mathrm{RowNormalize}\big(\mathrm{ColNormalize}(M^{(t-1)})\big),$$

where $\mathrm{RowNormalize}(M)$ scales each row of M to sum to 1, and similarly for $\mathrm{ColNormalize}$. In formula form: if $M^{(t-1)}$ is the current matrix, then

$$M_{ij}^{(t)} = \frac{M_{ij}^{(t-1)}}{\sum_k M_{ik}^{(t-1)}} \quad \text{(each row sums to 1)}$$

and then similarly normalize columns of the result. As $t \to \infty$, $M^{(t)}$ converges to a doubly-stochastic matrix [18] ; in practice a fixed T (e.g. $T = 20$) suffices for a good approximation. The final $W_\ell^{(r)} = M^{(T)}$ thus satisfies

$$W_\ell^{(r)} \mathbf{1}_n = \mathbf{1}_n, \quad (\mathbf{1}_n^\top W_\ell^{(r)}) = \mathbf{1}_n^\top,$$

up to machine precision [12] [18] .

> **Key Step:** Sinkhorn–Knopp alternately normalizes rows and columns of a positive matrix. Starting from $M^{(0)} = \exp(\widetilde{W}_\ell^{(r)})$, one repeats $\text{row-normalize}(\text{col-normalize}(M))$ until convergence. This projects $\widetilde{W}_\ell^{(r)}$ into the Birkhoff polytope of doubly-stochastic matrices [18] .

Because of this projection, each learned skip-matrix $W_\ell^{(r)}$ is guaranteed to lie on the stable manifold: its row/column sums stay 1, its spectral norm stays ≤1, and it acts like a convex combination of permutations [14] [16] .

mHC Forward Pass: Putting It All Together

We can now summarize the full forward pass of an mHC layer (mixing all components). Let $x_\ell \in \mathbb{R}^C$ be the input to layer ℓ. The layer first **expands** it to n streams:

$$\tilde{x}_\ell = \begin{pmatrix} x_\ell \\ x_\ell \\ \vdots \\ x_\ell \end{pmatrix} \in \mathbb{R}^{n \times C}.$$

Then it computes: 1. **Sinkhorn Projection:** From a raw parameter $\widetilde{W}_\ell^{(r)}$, compute the doubly-stochastic skip-matrix $W_\ell^{(r)} = \mathrm{Sinkhorn}(\widetilde{W}_\ell^{(r)})$ as above [18] . 2. **Stream Read-In:** Collapse streams for the layer input:

$$u_\ell = W_\ell^{(\mathrm{in})}\, \tilde{x}_\ell \in \mathbb{R}^{1 \times C}.$$

3. **Layer Transformation:** Apply the (shared) core network F on u_ℓ with its weights:

$$v_\ell = F(u_\ell; W_\ell) \;\in\; \mathbb{R}^{1 \times C}.$$

4. **Stream Write-Out:** Spread the output into all streams:

$$\widetilde{v}_\ell \;=\; (W_\ell^{(\mathrm{out})})^\top\, v_\ell \;\in\; \mathbb{R}^{n\times C}.$$

5. **Residual Mixing:** Apply the (doubly-stochastic) skip:

$$\tilde{s}_\ell \;=\; W_\ell^{(r)}\,\widetilde{x}_\ell \;\in\; \mathbb{R}^{n\times C}.$$

6. **Combine:** Form the new residual state:

$$\widetilde{x}_{\ell+1} \;=\; \tilde{s}_\ell + \widetilde{v}_\ell \;\in\; \mathbb{R}^{n\times C}.$$

Finally, if the network's next layer expects a single C-vector, one can collapse by $W_{\ell+1}^{(\mathrm{in})}$ as above, or else simply carry $\widetilde{x}_{\ell+1}$ forward.

In effect, each layer's output is the sum of a **doubly-stochastic-mixed copy** of the input plus the output of F redistributed across streams. Symbolically:

$$\widetilde{x}_{\ell+1} = W_\ell^{(r)}\,\widetilde{x}_\ell \;+\; (W_\ell^{(\mathrm{out})})^\top\, F\big(W_\ell^{(\mathrm{in})}\,\widetilde{x}_\ell; W_\ell\big).$$

This matches the general HC formula (Eq.3) but with the key addition that $W_\ell^{(r)}$ is doubly-stochastic. By writing out the shapes explicitly, one sees exactly how information flows from input $\widetilde{x}_\ell$ through the layer and back into $\widetilde{x}_{\ell+1}$.

> **Intuition:** The term $W_\ell^{(r)}\,\widetilde{x}_\ell$ is the *identity-path* component, mixing the n streams but preserving the overall signal (since $W_\ell^{(r)}$ is convex/mean-preserving). The term $(W_\ell^{(\mathrm{out})})^\top F(W_\ell^{(\mathrm{in})}\widetilde{x}_\ell)$ is the new residual update, injected into each stream via $W_\ell^{(\mathrm{out})}$. Together they yield a high-capacity "feature fusion" layer that remains numerically stable.

Backpropagation through the Sinkhorn Projection

Since each $W_\ell^{(r)}$ is computed by iterating the differentiable Sinkhorn steps, gradients from the loss can propagate through the entire normalization process back to $\widetilde{W}_\ell^{(r)}$. Concretely, let the loss $\mathcal{L}$ depend (through the network) on $W_\ell^{(r)}$. Because Sinkhorn–Knopp consists of elementary operations (exponentials, divides by row-sums, divides by column-sums), one can apply the chain rule through each iteration to compute $\partial\mathcal{L}/\partial\widetilde{W}_\ell^{(r)}$. In practice, the implementation often **recomputes** the intermediate matrices on-the-fly during the backward pass [19], but the upshot is that the constraint is smooth and differentiable. Thus training can adjust $\widetilde{W}_\ell^{(r)}$ (and indirectly $W_\ell^{(r)}$) by standard gradient descent, while keeping $W_\ell^{(r)}$ doubly-stochastic at each forward step [19].

Importantly, because the projection ensures $\|W_\ell^{(r)}\|_2 \le 1$, the backward gradients are **norm-preserving** as well: the effective Jacobian of each skip is a non-expansive mapping. In other words, the gradient backpropagating through $\widetilde{x}_{\ell+1} = W_\ell^{(r)}\widetilde{x}_\ell + \cdots$ is multiplied by $W_\ell^{(r)\top}$, which also has spectral norm ≤1. This closes the loop: just as forward activations are kept stable, so are the gradients.

> **Key Insight:** The Sinkhorn projection is fully differentiable. In practice, mHC uses a custom backward kernel that re-runs the row/column normalizations in reverse to propagate

gradients [19]. Thus constraints do not block learning: $\widetilde{W}_\ell^{(r)}$ is updated by $\nabla_{\widetilde{W}_\ell^{(r)}}\mathcal{L}$, computed through all Sinkhorn iterations.

Summary: A Stable, High-Capacity Residual Architecture

Putting it all together, mHC integrates several ideas: it **widens** the residual connection (increasing capacity) while **constraining** the skip-matrices to preserve identity-like behavior (ensuring stability). Each residual path has n streams, mixed by trainable matrices, so the network can represent richer transformations. But by projecting those matrices onto the Birkhoff polytope (doubly-stochastic matrices), mHC **guarantees** that the overall residual mapping remains non-expansive and mean-preserving at every layer [14] [17]. The Sinkhorn algorithm provides a practical way to enforce this constraint online.

In summary:

- **Standard ResNet:** 1 stream, identity skip I; stable but limited capacity.
- **Unconstrained HC:** n streams, arbitrary mixing $W^{(r)}$; high capacity but *unstable* (skip path no longer identity) [9].
- **mHC:** n streams, but $W^{(r)}$ is doubly-stochastic; high capacity **and** each skip is effectively a convex combination of permutations [14] [16]. This yields a “physics-inspired” conservation of signal: each layer’s skip neither amplifies nor diminishes the average signal, preserving the benefits of identity skips while adding flexibility.

Key Takeaway: mHC achieves the best of both worlds. By constraining residual mixing matrices to lie on the doubly-stochastic manifold, it restores the identity-mapping property (mean and norm preservation) while still allowing complex multi-stream interactions [14] [17]. The result is a deep residual network that is both *high-capacity* and *numerically stable*, scaling to very deep/wide models without runaway signals or exploding gradients.

Sources: These derivations and explanations follow the formulation of mHC in Xie et al. (2026) [4] [14] [18] and related analyses in the mHC literature [9] [17] [19]. Key mathematical definitions (e.g. of doubly-stochastic matrices and the Sinkhorn algorithm) are taken from that paper.

---

[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] mHC: Manifold-Constrained Hyper-Connections
https://arxiv.org/pdf/2512.24880