STREAMDA: RECURRENT SPATIOTEMPORAL MODELING FOR HOURLY GLOBAL ATMOSPHERIC DATA ASSIMILATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Data assimilation, as the core of numerical weather prediction, is a computationally complex and extremely time-consuming process. In recent years, deep learning-based methods have substantially reduced the computational cost of data assimilation. However, existing deep learning methods still exhibit limitations in the temporal modeling of multi-temporal observations and the efficient fusion of multi-source heterogeneous observations. We propose StreamDA, a recurrent spatiotemporal modeling framework for hourly global data assimilation, powered by a Memory-based Temporal Fusion (MTF) module and a Coverage Modulation for Observation Fusion (CMOF) module. MTF retrieves temporal context from a sequence of historical latent analysis states and propagates it across time, achieving effective integration of multi-temporal observations under hourly cycling assimilation. CMOF combines local coverage with source-specific learnable weights to dynamically modulate observational contributions, improving spatial feature fusion of multi-source observations. In global assimilation experiments, StreamDA reduces mean relative RMSE across variables by 18.2% compared with the primary baseline model, with lower errors in 67 of 69 output variables. Furthermore, the model exhibits greater long-term stability and generates hourly, high-refresh analysis fields, providing robust support for real-time weather forecasting.

1 INTRODUCTION

Numerical weather prediction systems integrate a data assimilation system with a forecast model. The assimilation system combines observations with a background field to produce an analysis field. Taking this analysis as input, the forecast model generates future weather predictions, with its short-range forecast serving as the background field for the next assimilation (Carrassi et al., 2018; Rabier et al., 2000). Traditional data assimilation methods are computationally expensive (Courtier et al., 1994; Bauer et al., 2015) and typically update every 6 hours, limiting global high-refresh operational forecasting (Slivinski & Whitaker, 2025).

Deep learning-based DA directly learns the mapping from observations, with or without a background field, to an analysis field. This offers a computationally efficient alternative (Geer, 2021; Cheng et al., 2023) and the potential to maximize observation utility (Xu et al., 2025). Research has progressed from idealized simulation experiments (Fablet et al., 2021; Penny et al., 2022; Boudier et al., 2023; Rozet & Louppe, 2023; Peyron et al., 2021) toward the assimilation of real-world observations for operational applications, demonstrating encouraging performance (Vandal et al., 2025; Xu et al., 2026; Xiao et al., 2025).

However, the temporal and spatial fusion of multi-source observations remains underexplored in existing deep learning-based data assimilation methods. First, existing studies typically map multi-temporal observations directly onto the target analysis field (Xu et al., 2025; Allen et al., 2025; Sun et al., 2025; Gupta et al., 2026), lacking explicit modeling of atmospheric temporal dynamics. Due to atmospheric advection, air parcels observed at non-target times migrate to other positions by the target time (Verma et al., 2024), making it challenging for the network to resolve spatiotemporal dependencies between asynchronous observations and the target analysis state. Second, prevailing methods fuse multi-source observations through simple feature concatenation or averaging (Allen

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

et al., 2025; Gupta et al., 2026; Ni et al., 2025), ignoring differences in instrument reliability and local spatial coverage across sources, which are inevitable in real-world operational settings.

We propose **StreamDA**, an hourly cycling framework for global data assimilation. We introduce the Memory-based Temporal Fusion (MTF) to explicitly model temporal dependencies and aggregate observational information across time, and the Coverage Modulation for Observation Fusion (CMOF) to encode and spatially fuse multi-source observations. Specifically, MTF stores latent representations of historical analysis fields in a memory bank and retrieves temporal context from multiple historical memories. It then propagates this information forward as a spatiotemporal **stream**, enabling historical observational information to accumulate across hourly assimilation cycles. CMOF encodes point observations and quantifies local coverage from the encoded features. It then combines the local coverage with learnable source-specific weights to dynamically modulate observational contributions, achieving adaptive feature fusion across varying observation densities. Experiments with real-world global observations demonstrate that StreamDA significantly reduces analysis errors. Ablation studies further confirm the framework's critical role in enhancing assimilation accuracy and system stability. Our main contributions are summarized as follows:

- **Memory-based Temporal Fusion.** We design a memory mechanism to store and retrieve historical latent analysis states, thereby incorporating temporal context within a single assimilation process. By incorporating multi-step historical analysis states through MTF, the assimilation accuracy of StreamDA is improved.
- **Coverage Modulation for Observation Fusion.** We introduce regional coverage and instrument-specific learnable weights to modulate observational contributions for multi-source fusion. CMOF enables StreamDA to maintain stable analysis accuracy during observation unavailability in operational settings.
- **Hourly Cycling Framework.** We propose an hourly cycling framework that integrates latent information propagation, analysis state updates, and latent state retention. By incorporating temporal observations into this framework, StreamDA achieves superior assimilation accuracy compared to non-temporal methods. Furthermore, the framework generates high-refresh-rate, hourly analysis fields, thereby better capturing rapidly evolving weather systems to support real-time forecasting.

2 Related Work

Deep learning-based Data Assimilation. In recent years, deep learning-based data assimilation models have laid the foundation for constructing AI-driven, end-to-end weather forecasting systems (Maddy et al., 2024; Huang et al., 2024; Xiang et al., 2025; Wang et al., 2024; Vandal et al., 2025; Allen et al., 2025; Sun et al., 2025; Xu et al., 2026). Diff-DA and FlowDA relied on sparse simulated observations to achieve highly accurate atmospheric state estimation (Huang et al., 2024; Cheng & Zhu, 2026). FuXi-DA performs assimilation of real-world satellite observations, demonstrating that neural networks can effectively capture the physical relationships between atmospheric states and complex satellite observations (Xu et al., 2025). In addition, studies such as Aardvark Weather, FuXi-Weather, and HealDA have achieved assimilation of real-world multi-source observations (Allen et al., 2025; Sun et al., 2025; Gupta et al., 2026). Appendix Table 3 summarizes the resolution, observation inputs, and cycling models of deep learning-based assimilation systems.

Observation Temporal Modeling. Existing deep learning-based assimilation models assimilate multi-temporal observations within a time window to address sparsity (Fan et al., 2026) (Appendix Table 3). However, they have limitations in temporal modeling. Aardvark Weather compresses multi-temporal observations into a $C \times H \times W$ tensor with time-encoding channels, discarding same-location observations from different times (Allen et al., 2025). FuXi-Weather retains a time dimension but concatenates it with the channel dimension without temporal modeling (Sun et al., 2025). HealDA averages time-encoded observation features at each grid point rather than updating the analysis state sequentially within the observation window (Gupta et al., 2026).

Observation Encoding and Fusion. Multi-source observations are discrete points with varying spatial resolutions and irregular distributions (Alexe et al., 2024). Set-based neural networks encode unordered inputs through permutation-invariant aggregation or attention (Zaheer et al., 2017; Lee et al., 2019; Vaswani et al., 2017; Qi et al., 2017). Most deep learning-based assimilation models

directly resample raw observations onto regular grids, constructing highly sparse gridded tensors (Xu et al., 2025; Sun et al., 2025; Allen et al., 2025; Ni et al., 2025; Pinnington et al., 2026). This causes a substantial loss of high-resolution observations and introduces massive computational and memory overhead through direct processing of sparse tensors. HealDA encodes discrete observation points and grids their encoded features, but neglects the relative geometric displacement between observation locations and grid nodes during the grid-mapping process (Gupta et al., 2026). For multi-source feature fusion, current studies typically rely on naive feature concatenation or averaging (Xu et al., 2025; Allen et al., 2025; Sun et al., 2025; Gupta et al., 2026), failing to account for disparities in data reliability and localized spatial coverage across different observation sources.

3 METHOD

3.1 PROBLEM FORMULATION

Data assimilation combines observations with prior information, typically a short-term forecast, to estimate the current atmospheric state. Let t denote the assimilation time, and let y_t denote the observations used for this assimilation process. The analysis field x_t^a can be expressed as:

$$x_t^a = \mathcal{A}(x_t^b, y_t), \tag{1}$$

Here, $\mathcal{A}$ denotes the assimilation model, and x_t^b represents an optional background field, obtained from the forecast of the previous analysis time:

$$x_t^b = \mathcal{F}(x_{t-\Delta t}^a), \tag{2}$$

where $\mathcal{F}$ denotes the forecast model, and Δt represents the assimilation time interval, typically 6 or 12 hours. Background fields (x^b) play a crucial role in propagating historical observational information (Feng et al., 2026); therefore, the forecast model $\mathcal{F}$ is often necessary (Sun et al., 2025). The assimilation model $\mathcal{A}$ and the forecast model $\mathcal{F}$ operate alternately to continuously generate analysis and background fields. To assimilate more observations at the analysis time, y_t typically represents the set of observations over the assimilation window $[t-L, t+R]$, expressed as $y_t = \{\mathcal{O}_{t-L}, \mathcal{O}_{t-L+1}, \ldots, \mathcal{O}_{t+R}\}$. When $R > 0$, observations subsequent to the analysis time are utilized, which introduces a longer operational latency to wait for these data.

We adopt an hourly assimilation framework that models temporal evolution in latent space without an external forecast model. For an assimilation time t, the framework utilizes only the observation $\mathcal{O}_t$ to avoid observational waiting latency in operational deployment. Meanwhile, the propagation of historical observational information is achieved by carrying forward the memory $\mathcal{M}_{t-1}$ retained from previous assimilation times (see Section 3.2.2). This process can be formulated as:

$$(x_t^a, \mathcal{M}_t) = \mathcal{A}_\theta\left(\mathcal{O}_t, \mathcal{M}_{t-1}\right). \tag{3}$$

where $\mathcal{M}_t$ stores latent states of historical analysis fields for the next assimilation time.

3.2 FRAMEWORK OVERVIEW

Figure 1(a) shows the two branches of StreamDA. The CMOF module encodes observations into point features, maps them to grid locations, and fuses source features using local coverage. The MTF module combines retrieved historical temporal context with recent state changes to predict a latent background. An 18-block Transformer backbone (Liu et al., 2021) processes the sum of the two branch outputs to produce a latent analysis state, which a convolutional decoder maps to the analysis field (Appendix A.4). Below, we describe CMOF, MTF, and the training objective.

3.2.1 COVERAGE MODULATION FOR OBSERVATION FUSION

Observation Encoding. For source s, observation j contains values $v_{t,j}^s$, validity masks $m_{t,j}^s$, and metadata $r_{t,j}^s$. A separate MLP for each source produces point features:

$$h_{t,j}^s = \mathrm{MLP}^s\left(\left[v_{t,j}^s \odot m_{t,j}^s, m_{t,j}^s, r_{t,j}^s\right]\right). \tag{4}$$

Weight Mapping in Figure 1(b) aggregates these features at nearby grid locations using learned nonnegative weights:

$$w_{t,j}^s(g) = \mathrm{softplus}\left(f^s(h_{t,j}^s, d_{j,g})\right), \qquad \widetilde{O}_t^s(g) = \frac{\sum_j w_{t,j}^s(g) h_{t,j}^s}{\sum_j w_{t,j}^s(g) + 10^{-6}}. \tag{5}$$

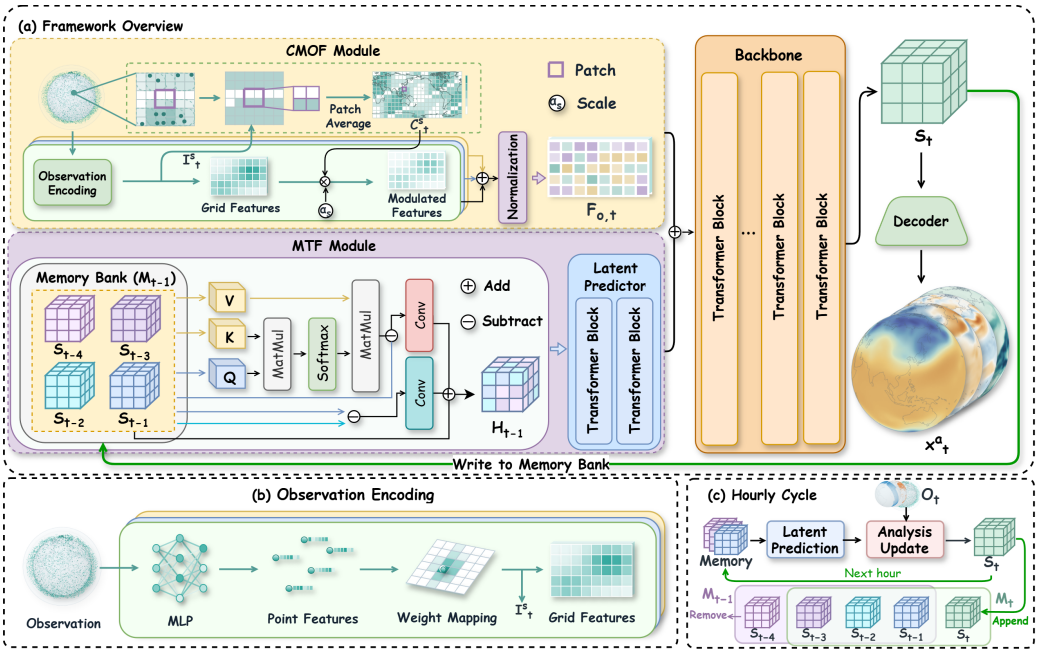


Figure 1: StreamDA architecture and data flow. (a) CMOF encodes and fuses current observations, while MTF predicts a latent background from stored analysis states. Their outputs are added before Backbone. Decoder produces the analysis field, and the green path writes the updated latent state back to memory. Purple squares mark patches; stacked panels represent observation sources, and mixed feature colors indicate their contributions. (b) Observation Encoding produces Grid Features and records grid occupancy I_t^s for coverage computation. (c) Hourly cycling and memory update. The overlapping purple and green outlines show memory before and after writeback: the oldest state is removed if the memory bank is full, and the latest state is appended for the next cycle.

Here, f^s is the weighting network for source s, and $d_{j,g}$ is the distance from observation j to grid location g. In the expression for $\widetilde{O}_t^s(g)$, both the numerator and denominator sum over valid observations mapped to grid location g. Alongside $\widetilde{O}_t^s$, we record grid occupancy I_t^s, observation counts, and accumulated weights to preserve sampling information beyond the weighted average. At each grid location, I_t^s is one if any valid observation is mapped there and zero otherwise. Convolution and patch projection combine these statistics with $\widetilde{O}_t^s$ to produce source grid features O_t^s.

Coverage Modulation. As shown in Figure 1(a), Patch Average computes coverage C_t^s by averaging grid occupancy I_t^s within each patch. We use this coverage and the source scale α_s to modulate source features, then sum and normalize them to obtain the fused observation features $F_{o,t}$:

$$F_{o,t} = \frac{\sum_{s=1}^{S} \alpha_s C_t^s \odot O_t^s}{\sqrt{1 + \sum_{s=1}^{S} \alpha_s^2 C_t^s}}. \tag{6}$$

The positive learnable scalar α_s sets each source's overall contribution. Coverage scales all feature components of a source equally at each location. The denominator limits growth in feature magnitude as more sources contribute. The CMOF output $F_{o,t}$ provides the observation input for the analysis update. Appendix A.3 provides details of observation encoding, grid mapping, and coverage computation.

3.2.2 Memory-based Temporal Fusion

Each analysis update produces a latent state S_t that is decoded into the analysis field x_t^a and stored in memory (Figure 1(c)). The latest state S_{t-1} carries information from earlier observations, and the stored sequence records how the analysis has evolved. The MTF module uses this history and the change over the preceding hour to predict the current latent background.

Memory Bank. The memory bank in Figure 1(a) stores the most recent K_{mem} latent analysis states $S_t \in \mathbb{R}^{D\times H_{\mathrm{lat}}\times W_{\mathrm{lat}}}$:

$$\mathcal{M}_{t-1} = \{S_{t-K_{\mathrm{mem}}}, \ldots, S_{t-1}\}. \tag{7}$$

Here, D is the feature dimension, and H_{lat} and W_{lat} are the latent grid dimensions. We set $K_{\mathrm{mem}} = 4$ and mask unavailable entries during initialization (Algorithm 1). After each analysis update, we form $\mathcal{M}_t$ by appending S_t to $\mathcal{M}_{t-1}$ and removing the oldest state if capacity is exceeded (Figure 1(c)). The updated memory carries current observational information into the next hourly prediction.

Memory Retrieval and Enhancement. At each latent grid location, the latest state queries the stored states to retrieve historical context R_{t-1}. In Figure 1(a), MatMul computes query–key matching scores, Softmax gives temporal weights, and the second MatMul aggregates the stored states:

$$a_{t,i} = \mathrm{softmax}_i\left(\frac{\langle Q(S_{t-1}), K(S_{t-i}) + e_i\rangle}{\sqrt{d}}\right), \qquad R_{t-1} = \sum_{i=1}^{K_{\mathrm{mem}}} a_{t,i} S_{t-i}. \tag{8}$$

Here, Q and K are learned query and key projections, d is their feature dimension, and e_i encodes the memory position. Invalid entries are masked before softmax. The stored states serve directly as values, and the temporal weights $a_{t,i}$ determine their contributions to the retrieved context R_{t-1}. The two subtraction branches in Figure 1(a) compare this context with the latest state and compute the change between the two latest states:

$$H_{t-1} = S_{t-1} + W_c\,(R_{t-1} - S_{t-1}) + W_\Delta\,(S_{t-1} - S_{t-2}). \tag{9}$$

The two differences pass through separate convolutions W_c and W_Δ without bias. A residual connection adds their outputs to the latest state to form H_{t-1}. Latent Predictor uses H_{t-1} to estimate an increment, which is added to the latest analysis state S_{t-1} to obtain the latent background S_t^-:

$$S_t^- = S_{t-1} + \mathrm{LatentPredictor}\,(H_{t-1}). \tag{10}$$

The latent background S_t^- is added to the CMOF output $F_{o,t}$ and processed by the backbone to produce the updated analysis state S_t.

3.2.3 Training and Loss Design

The analysis state S_t and latent background S_t^- share the decoder, producing x_t^a and x_t^b, respectively. We supervise both outputs against the standardized reference x_t with a latitude-weighted L_1 loss:

$$\mathcal{L}_{\mathrm{state}}(\widehat{x}, x) = \frac{1}{|\mathcal{V}|HW}\sum_{v\in\mathcal{V}}\sum_{i=1}^{H}\sum_{j=1}^{W}\alpha_i\,|\widehat{x}_{v,i,j} - x_{v,i,j}|, \qquad \alpha_i = \frac{H\cos\Phi_i}{\sum_{k=1}^{H}\cos\Phi_k}. \tag{11}$$

Here, $\mathcal{V}$ is the set of variables, H and W are the numbers of latitude rows and longitude columns, and Φ_i is the latitude of row i. We average both losses over the backpropagation cycles:

$$\mathcal{L} = \frac{1}{|\mathcal{G}|}\sum_{t\in\mathcal{G}}\left[\mathcal{L}_{\mathrm{state}}(x_t^a, x_t) + \lambda_{\mathrm{prior}}\mathcal{L}_{\mathrm{state}}(x_t^b, x_t)\right], \tag{12}$$

where $|\mathcal{G}| = 4$ and $\lambda_{\mathrm{prior}} = 0.1$ weights the background loss (Appendix C.3).

4 Experiment

4.1 Data Setup

We evaluate hourly global assimilation at 1° resolution, estimating 69 atmospheric variables from 13 real-world observation sources. The analysis variables include geopotential, temperature, U component of wind, V component of wind, and specific humidity at 13 pressure levels, together with 2 metre temperature, 10 metre U wind component, 10 metre V wind component, and mean sea level pressure. Training labels come from the 0.25° ERA5 reanalysis (Hersbach et al., 2020), preprocessed onto the 1° grid by conservative regridding of first order using weights based on overlapping cell areas (Rasp et al., 2024). The observation sources include ATMS, MHS, AMSU-A, IASI, HIRS, and SSMI for satellite observations, SURF and ICOADS for surface and marine observations, TEMP and WIND for observations of the upper atmosphere, and SCAT, GNSS, and AIRCRAFT for scatterometer winds, radio occultation, and aircraft observations, respectively (Appendix Table 4).

4.2 Implementation Details

We train on hourly data from January 1, 2016, to May 31, 2023, and reserve July 1, 2023, to June 30, 2024, for independent testing. During testing, StreamDA performs continuous hourly assimilation (Algorithms 2). For comparisons with HealDA and FuXi-Weather, we select its outputs at 00, 06, 12, and 18 UTC. ERA5 provides the ground truth for all experiments.

4.2.1 Evaluation Metrics

We evaluate analysis accuracy using latitude-weighted root mean square error (RMSE). For variable v, we average spatial RMSE over evaluation times:

$$\mathrm{RMSE}_v = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \sqrt{\frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \alpha_i \left(x^a_{t,v,i,j} - x_{t,v,i,j} \right)^2}, \quad \alpha_i = \frac{H \cos \Phi_i}{\sum_{i'=1}^{H} \cos \Phi_{i'}}. \tag{13}$$

where x^a and x are the analysis and reference values, $\mathcal{T}$ contains the evaluation times, and Φ_i is the latitude of grid row i. Overall relative improvements are computed from these time-mean RMSEs for each variables and then averaged equally across variables.

4.2.2 Compared Models

We compare StreamDA with two existing systems:

- **HealDA (Gupta et al., 2026):** A global assimilation model producing analysis fields at 1° resolution every six hours. We evaluate the two configurations reported in the original paper using the same pretrained model: HealDA uses observations from 21 hours before to 3 hours after the analysis time, while HealDA* uses only observations from the preceding 24 hours.
- **FuXi-Weather (Sun et al., 2025):** A global system operating at 0.25° resolution with six-hourly updates. It combines a neural data assimilation component with a trained forecast model to cycle through real-world observations.

The original HealDA configuration gives lower errors and is our primary neural baseline.

4.2.3 Training Strategy

We train StreamDA in PyTorch on four NVIDIA H800 GPUs for 28,000 steps with a global batch size of 4. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight decay 0.1. The learning rate increases to 5×10^{-4} over 320 steps, follows cosine decay until step 16,000, then remains at 10^{-5}. During training, we unroll hourly updates, each using the current observations and carrying the updated latent state and memory to the next hour. The rollout length is 4 updates for the first 16,000 training steps, then 8 from step 16,001, increasing by 4 every 1,000 training steps to 52. For longer rollouts, earlier updates build the state and memory without gradient computation. We average the analysis and background losses over the final 4 updates and backpropagate through these updates. We update the exponential moving average parameters at each step with decay 0.99.

4.3 Overall Comparison

Table 1 compares mean analysis RMSE across nineteen variables. At six-hourly evaluation times, StreamDA improves all relative to both HealDA configurations. Across all 69 variables, its mean relative RMSE reductions are **18.2%** against HealDA and **20.0%** against HealDA*, with lower errors on 67 variables in each comparison (Appendix B.1). FuXi-Weather is included as a supplementary reference at its native 0.25° resolution. StreamDA analysis fields also improve downstream forecast accuracy compared with HealDA* when used to initialize the same forecast model (Appendix B.2). To examine whether these gains persist over time, Figure 2 compares RMSE throughout the test period. HealDA shows clear error spikes, particularly for Z500 and T2M, while StreamDA has no comparable spikes. StreamDA also reaches its typical error level earlier after initialization. At six-hourly evaluation times, its Z500 RMSE first falls below its own test-period mean at 6 h, compared with 48 h for FuXi-Weather. These results suggest a shorter initial adjustment for Z500 and more stable analysis accuracy across the displayed variables during subsequent cycling.

Table 1: Mean analysis RMSE across nineteen variables. Pressure-level entries follow 300/500/850 hPa, with units below variable names. StreamDA reports hourly outputs for reference; StreamDA (6 h) samples these every six hours for comparison. Bold marks the best six-hourly results. N/A indicates an unavailable output. Lower values are better.

Method	Pressure level (300/500/850 hPa)					Surface			
	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
HealDA	120.77/74.99/41.87	0.66/0.63/0.87	2.16/1.86/1.42	2.16/1.80/1.39	0.063/0.33/0.83	1.10	0.88	0.90	60.11
HealDA*	123.26/77.73/45.14	0.67/0.64/0.88	2.22/1.90/1.45	2.24/1.85/1.42	0.065/0.34/0.85	1.10	0.91	0.93	63.91
FuXi-Weather	90.63/81.72/82.41	0.66/0.68/1.10	2.68/2.26/1.84	2.59/2.10/1.74	N/A	1.30	1.15	1.14	118.43
StreamDA (6 h)	**61.41/47.10/35.37**	**0.50/0.54/0.78**	**1.91/1.73/1.21**	**1.94/1.69/1.21**	**0.048/0.27/0.68**	**0.80**	**0.67**	**0.67**	**49.93**
StreamDA	61.99/47.59/36.04	0.50/0.55/0.79	1.92/1.74/1.22	1.95/1.69/1.22	0.048/0.27/0.68	0.82	0.67	0.68	50.80

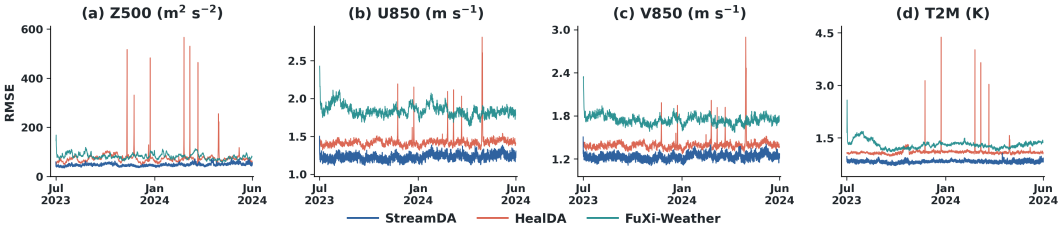


Figure 2: Analysis RMSE time series for StreamDA, HealDA, and FuXi-Weather. Panels (a)–(d) show Z500, U850, V850, and T2M. Horizontal axes show dates and vertical axes show RMSE.

4.4 Ablation Studies

We evaluate MTF module and CMOF module through two module ablations. The w/o MTF setting removes history retrieval and the tendency term, while retaining the latent predictor and recurrent updates with $H_{t-1} = S_{t-1}$. The w/o CMOF setting retains observation encoding and replaces the fusion in Equation 6 with equal source weights and fixed normalization, following HealDA (Gupta et al., 2026). Evaluated on hourly outputs, both ablations increase mean RMSE across all nineteen reported variables (Table 2). Removing CMOF has the largest relative effects on geopotential and mean sea-level pressure. These results support the contributions of both modules to analysis accuracy. The following experiments examine temporal modeling, retained history, and performance under missing observations in more detail.

4.4.1 Temporal Modeling Within an Observation Window

Comparisons with HealDA assess overall system performance but do not isolate temporal modeling, since the systems also differ in architecture, inputs, and training. We therefore compare joint observation encoding with sequential assimilation within the same six-hour window. Window-6h is trained separately using the same observation encoder architecture and backbone as StreamDA. For each source s, it concatenates six hours of point observations along the observation dimension, combining tensors in $\mathbb{R}^{B\times N_u^s\times d^s}$ into $\mathbb{R}^{B\times(\sum_{u=t-5}^{t} N_u^s)\times d^s}$ before encoding and grid mapping. Here, B is batch size, N_u^s counts observations at hour u, and d^s is the number of values per observation. It retains each observation's time offset relative to the target analysis time as an encoder input and removes the MTF branch, mapping each observation window directly to its endpoint analysis field without carrying states between windows (Algorithm 4). StreamDA (6 h rollout) uses the same checkpoint as StreamDA. For each hourly target t, it resets the state and memory, then assimilates $\mathcal{O}_{t-5}, \ldots, \mathcal{O}_t$ through six hourly updates to produce the analysis field at t (Algorithm 3). Figure 3 compares the analysis accuracy of StreamDA (6 h rollout) and Window-6h. Panel (a) shows lower mean RMSE for StreamDA (6 h rollout) across the displayed variables. The mean relative RMSE reduction across 69 variables is **7.59%**. Panel (b) shows sustained hourly reductions across the five displayed variables, with larger relative gains during hours marked by observation gaps. Across 69 variables, the mean relative RMSE reduction increases from **7.25%** without observation gaps

Table 2: Module ablation RMSE on hourly outputs. Pressure-level entries follow 300/500/850 hPa. Units appear below variable names. Bold marks the lowest displayed values, including ties.

Method	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
StreamDA	**61.99/47.59/36.04**	**0.50/0.55/0.79**	**1.92/1.74/1.22**	**1.95/1.69/1.22**	**0.048/0.27/0.68**	**0.82**	**0.67**	**0.68**	**50.80**
w/o MTF	66.79/50.53/37.42	0.52/0.57/0.81	1.97/1.79/1.25	2.00/1.73/1.24	0.049/0.28/0.70	0.84	0.69	0.69	52.60
w/o CMOF	73.17/54.71/41.78	0.52/0.57/0.81	1.95/1.77/1.24	1.98/1.72/1.24	**0.048**/0.28/0.69	0.85	0.69	0.69	58.40

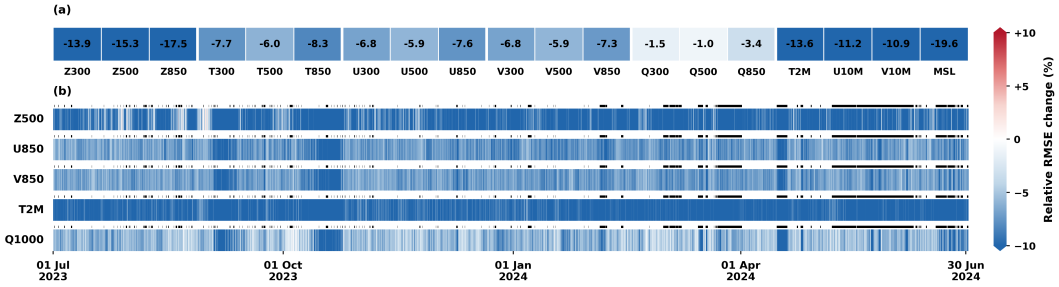


Figure 3: Percentage changes in analysis RMSE for StreamDA (6 h rollout) relative to Window-6h. Panel (a) compares mean hourly RMSE across nineteen variables; panel (b) shows hourly changes for Z500, U850, V850, T2M, and Q1000. Negative values (blue) indicate that StreamDA (6 h rollout) outperforms Window-6h, while positive values (red) indicate that Window-6h performs better. Black bars mark hours when at least one of the twelve sources other than TEMP has no usable observations. TEMP is excluded because its input interval is six hours.

to **8.44%** with gaps. These results suggest that explicit temporal modeling helps capture atmospheric evolution within the observation window, improving analysis accuracy. When observations are missing, StreamDA (6 h rollout) also maintains more stable analysis accuracy than Window-6h. Appendix B.3 reports computational costs.

4.4.2 Effect of Memory Capacity

We vary memory capacity to examine how many historical analysis states are useful for temporal modeling. The w/o MTF setting provides an equivalent single-state reference: it retains the latest state and latent predictor but removes history retrieval and the recent state difference. We compare this reference with capacities of two, four, six, and eight states. For each configuration, we average the relative RMSE changes from the default four-state setting equally across all output variables:

$$\Delta_{\mathrm{RMSE}}(K_{\mathrm{mem}}) = \left[\frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \left(\frac{\overline{\mathrm{RMSE}}_v(K_{\mathrm{mem}})}{\overline{\mathrm{RMSE}}_v(4)} - 1 \right) \right] \times 100\%. \tag{14}$$

where $\overline{\mathrm{RMSE}}_v(K_{\mathrm{mem}})$ is mean hourly analysis RMSE and $\mathcal{V}$ contains all output variables. Negative values indicate improvement over $K_{\mathrm{mem}} = 4$. Figure 4 shows that retaining multiple states lowers aggregate RMSE compared with the single-state reference. Six states yield the lowest error, but gains beyond four states are small and do not increase consistently with capacity. This may be because we predict only the next hour rather than a longer forecast period, so storing more historical states offers limited additional benefit. We therefore retain four states to limit memory storage and retrieval costs while maintaining similar accuracy. Additional ablations support the benefit of our retrieval method (Appendix C.1).

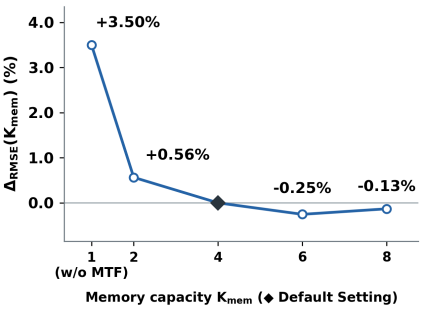


Figure 4: Mean relative RMSE change across all variables versus memory capacity.

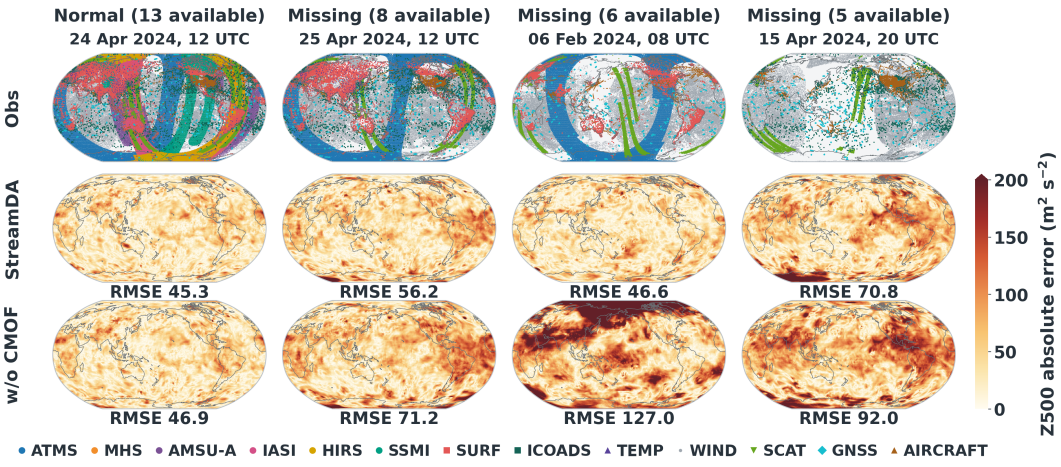


Figure 5: Observation distributions and Z500 analysis errors. Column headings give input conditions, available sources out of thirteen, and timestamps. The top row shows valid observations colored by source (bottom legend). The middle and bottom rows show absolute errors for StreamDA and w/o CMOF, with latitude-weighted global RMSE below each map. Error maps share the right colorbar; its extension marks errors beyond the displayed range.

4.4.3 Stability Under Missing Observations

Observation sources are not always available in operational settings due to temporary sensor outages or communication latency. We compare StreamDA with w/o CMOF under recorded observation gaps to examine whether adjusting source contributions helps maintain analysis accuracy when inputs are missing. Figure 5 shows observation distributions and Z500 errors for four cases. StreamDA has lower global RMSE in all cases, reducing widespread large errors in the third column and maintaining lower error with only five of thirteen sources available in the fourth column. Over the full test period, gains are also larger under missing observations than under normal coverage (Appendix D), supporting the role of CMOF in maintaining analysis accuracy with incomplete inputs. Controlled removal and component ablations further support CMOF's robustness to missing inputs and the benefit of coverage modulation (Appendices D.2 and C.2).

5 Conclusion

We presented StreamDA, an hourly global assimilation framework that models temporal evolution in latent space and fuses observations according to local coverage. MTF retains and retrieves historical latent analysis states for temporal context propagation across time, while CMOF adjusts source contributions as observation availability changes. Experiments show lower analysis errors and greater temporal stability. Ablations support explicit temporal modeling within the same observation window, the use of multiple historical states, and robust analysis under missing observations. Analysis fields produced by StreamDA also improve downstream forecast accuracy. However, the current 1° resolution limits fine spatial detail, and analysis uncertainty remains unquantified. Extending the framework to 0.25° also poses challenges in computational cost and memory usage. Future work will address these limitations through efficient assimilation at higher resolutions and uncertainty estimation using probabilistic modeling or ensembles.

AI Use Statement

We used generative AI tools to assist with language editing, literature discovery, and coding. The authors designed the method and experiments, reviewed the AI-assisted code, and were responsible for its execution and testing. The authors also verified the references and checked numerical results

against recorded experimental outputs. We take responsibility for the final content of this work, including all text, claims, code, and results produced with AI assistance.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 4895–4901, 2023.

Mihai Alexe, Eulalie Boucher, Peter Lean, Ewan Pinnington, Patrick Laloyaux, Anthony McNally, Simon Lang, Matthew Chantry, Chris Burrows, Marcin Chrust, et al. Graphdop: Towards skilful data-driven medium-range weather forecasts learnt and initialised directly from observations. *arXiv preprint arXiv:2412.15687*, 2024.

Anna Allen, Stratis Markou, Will Tebbutt, James Requeima, Wessel P Bruinsma, Tom R Andersson, Michael Herzog, Nicholas D Lane, Matthew Chantry, J Scott Hosking, et al. End-to-end data-driven weather prediction. *Nature*, 641(8065):1172–1179, 2025.

Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

Pierre Boudier, Anthony Fillion, Serge Gratton, Selime Gürol, and Sixin Zhang. Data assimilation networks. *Journal of Advances in Modeling Earth Systems*, 15(4):e2022MS003353, 2023.

Alberto Carrassi, Marc Bocquet, Laurent Bertino, and Geir Evensen. Data assimilation in the geosciences: An overview of methods, issues, and perspectives. *Wiley Interdisciplinary Reviews: Climate Change*, 9(5):e535, 2018.

Ran Cheng and Lailai Zhu. Flowda: Accurate, low-latency weather data assimilation via flow matching. *arXiv preprint arXiv:2602.06800*, 2026.

Sibo Cheng, César Quilodrán-Casas, Said Ouala, Alban Farchi, Che Liu, Pierre Tandeo, Ronan Fablet, Didier Lucor, Bertrand Iooss, Julien Brajard, et al. Machine learning with data assimilation and uncertainty quantification for dynamical systems: a review. *IEEE/CAA Journal of Automatica Sinica*, 10(6):1361–1387, 2023.

PHILIPPE Courtier, J-N Thépaut, and Anthony Hollingsworth. A strategy for operational implementation of 4d-var, using an incremental approach. *Quarterly Journal of the Royal Meteorological Society*, 120(519):1367–1387, 1994.

Ronan Fablet, Bertrand Chapron, Lucas Drumetz, Etienne Mémin, Olivier Pannekoucke, and François Rousseau. Learning variational data assimilation models and solvers. *Journal of Advances in Modeling Earth Systems*, 13(10):e2021MS002572, 2021.

Hang Fan, Lei Bai, Ben Fei, Yi Xiao, Kun Chen, Yubao Liu, Yongquan Qu, Fenghua Ling, and Pierre Gentine. Physically consistent global atmospheric data assimilation with machine learning in latent space. *Science Advances*, 12(1):eaea4248, 2026.

Jie Feng, Zoltan Toth, and Malaquias Peña. The limits of knowing: An equilibrium between the observational gain and chaotic loss of information in data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 152(775):e70067, 2026.

Alan J Geer. Learning earth system models from observations: machine learning or data assimilation? *Philosophical Transactions of the Royal Society A*, 379(2194):20200089, 2021.

Aayush Gupta, Akshay Subramaniam, Michael S Pritchard, Karthik Kashinath, Sergey Frolov, Kelsey Lieberman, Christopher Miller, Nicholas Silverman, and Noah D Brenowitz. Healda: Highlighting the importance of initial errors in end-to-end ai weather forecasts. *arXiv preprint arXiv:2601.17636*, 2026.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020.

Langwen Huang, Lukas Gianinazzi, Yuejiang Yu, Peter D Dueben, and Torsten Hoefler. Diffda: a diffusion model for weather-scale data assimilation. *arXiv preprint arXiv:2401.05932*, 2024.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9992–10002. Ieee, 2021.

Eric S Maddy, Sid A Boukabara, and Flavio Iturbide-Sanchez. Assessing the feasibility of an nwp satellite data assimilation system entirely based on ai techniques. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 17:9828–9845, 2024.

Zekun Ni, Jonathan Weyn, Hang Zhang, Yanfei Xiang, Jiang Bian, Weixin Jin, Kit Thambiratnam, Qi Zhang, Haiyu Dong, and Hongyu Sun. Huracan: A skillful end-to-end data-driven system for ensemble data assimilation and weather prediction. *arXiv preprint arXiv:2508.18486*, 2025.

Stephen G Penny, Timothy A Smith, T-C Chen, Jason A Platt, H-Y Lin, Michael Goodliff, and Henry DI Abarbanel. Integrating recurrent neural networks with data assimilation for scalable data-driven state estimation. *Journal of Advances in Modeling Earth Systems*, 14(3): e2021MS002843, 2022.

Mathis Peyron, Anthony Fillion, Selime Gürol, Victor Marchais, Serge Gratton, Pierre Boudier, and Gael Goret. Latent space data assimilation by using deep learning. *Quarterly Journal of the Royal Meteorological Society*, 147(740):3759–3777, 2021.

Ewan Pinnington, Peter Lean, Mihai Alexe, Eulalie Boucher, Simon Lang, Patrick Laloyaux, Gert Mertes, Tomas Kral, Patricia de Rosnay, Matthew Chantry, et al. Aifs-dop: End-to-end medium-range weather prediction from observations alone with machine learning. *arXiv preprint arXiv:2606.19093*, 2026.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Florence Rabier, Heikki Järvinen, Erland Klinker, J-F Mahfouf, and Adrian Simmons. The ecmwf operational implementation of four-dimensional variational assimilation. i: Experimental results with simplified physics. *Quarterly Journal of the Royal Meteorological Society*, 126(564):1143–1170, 2000.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

François Rozet and Gilles Louppe. Score-based data assimilation. *Advances in Neural Information Processing Systems*, 36:40521–40541, 2023.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Laura C Slivinski and Jeffrey S Whitaker. Investigating the impact of rapid cycling in a global data assimilation system. *Monthly Weather Review*, 153(12):2933–2945, 2025.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Xiuyu Sun, Xiaohui Zhong, Xiaoze Xu, Yuanqing Huang, Hao Li, J David Neelin, Deliang Chen, Jie Feng, Wei Han, Libo Wu, et al. A data-to-forecast machine learning system for global weather. *Nature Communications*, 16(1):6658, 2025.

Thomas J Vandal, Kate Duffy, Yoni Nachmany, and Daniel McDuff. Global atmospheric data assimilation with multi-modal masked autoencoders. In *2025 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 863–872. IEEE, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yogesh Verma, Markus Heinonen, and Vikas Garg. Climode: Climate and weather forecasting with physics-informed neural odes. In *International Conference on Learning Representations*, volume 2024, pp. 8408–8430, 2024.

Wuxin Wang, Jinrong Zhang, Qingguo Su, Xingyu Chai, Jingze Lu, Weicheng Ni, Boheng Duan, and Kaijun Ren. Accurate initial field estimation for weather forecasting with a variational constrained neural network. *npj Climate and Atmospheric Science*, 7(1):223, 2024.

Yanfei Xiang, Weixin Jin, Haiyu Dong, Jonathan Weyn, Mingliang Bai, Zuliang Fang, Pengcheng Zhao, Hongyu Sun, Kit Thambiratnam, Qi Zhang, et al. Adaf: An artificial intelligence data assimilation framework for weather forecasting. *Journal of Advances in Modeling Earth Systems*, 17(9):e2024MS004839, 2025.

Yi Xiao, Qilong Jia, Kun Chen, Lei Bai, and Wei Xue. Vae-var: variational autoencoder-enhanced variational methods for data assimilation in meteorology. In *International Conference on Learning Representations*, volume 2025, pp. 63049–63077, 2025.

Xiaoze Xu, Xiuyu Sun, Wei Han, Xiaohui Zhong, Lei Chen, Zhiqiu Gao, and Hao Li. Fuxi-da: A generalized deep learning data assimilation framework for assimilating satellite observations. *npj Climate and Atmospheric Science*, 8(1):156, 2025.

Xiaoze Xu, Xiuyu Sun, Songling Zhu, Xiaohui Zhong, Yuanqing Huang, Zijian Zhu, Jun Liu, and Hao Li. Fuxiweather2: Learning accurate atmospheric state estimation for operational global weather forecasting. *arXiv preprint arXiv:2603.15358*, 2026.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

A Data and Implementation Details

A.1 Comparison of Global Assimilation Systems

Table 3: Global learning-based data assimilation systems. Columns report spatial and temporal resolution, observation type, observation span relative to analysis time t, and cycling model. In situ denotes direct measurements at observing locations and Satellite denotes satellite observations. External and Internal indicate a separate forecast model and an internal predictor, respectively; — indicates no cycling.

Method	Spatial / Temporal	Observation type	Observation span	Cycle model
DiffDA (Huang et al., 2024)	0.25° / 6 h	Simulated	t	External
FlowDA (Cheng & Zhu, 2026)	0.25° / 6 h	Simulated	t	External
FuXi-DA (Xu et al., 2025)	0.25° / 6 h	Satellite	$[t-1\text{h}, t+1\text{h}]$	—
FuXi Weather (Sun et al., 2025)	0.25° / 6 h	Satellite	$[t-3\text{h}, t+4\text{h}]$	External
HealDA (Gupta et al., 2026)	1° / 6 h	In situ + Satellite	$[t-21\text{h}, t+3\text{h}]$	—
StreamDA	**1° / 1 h**	**In situ + Satellite**	$\boldsymbol{t}$	**Internal**
Latent DA (Fan et al., 2026)	1.4° / 6 h	In situ	$[t-36\text{h}, t]$	External

A.2 Observation Sources

Table 4: Observation sources and their input characteristics. Columns give the source name and description, input dimensionality, input interval, number of channels, and hourly availability. Dimension describes spatial organization: 2D observations have horizontal locations, whereas 3D observations also include vertical levels. The Channels column counts scalar observation values after flattening variables and vertical levels. Hourly availability is the percentage of evaluation hours with at least one valid observation from a source.

Source	Name or description	Dimension	Interval	Channels	Availability (%)
ATMS	Advanced Technology Microwave Sounder	2D	1 h	22	99.77
MHS	Microwave Humidity Sounder	2D	1 h	5	94.56
AMSU-A	Advanced Microwave Sounding Unit-A	2D	1 h	15	94.75
IASI	Infrared Atmospheric Sounding Interferometer	2D	1 h	38	94.67
HIRS	High-resolution Infrared Radiation Sounder	2D	1 h	20	91.21
SSMI	Special Sensor Microwave Imager	2D	1 h	10	83.65
SURF	Surface observations	2D	1 h	7	99.93
ICOADS	International Comprehensive Ocean–Atmosphere Data Set	2D	1 h	6	97.55
TEMP	Upper-air profile observations	3D	6 h	65	15.08
WIND	Upper-air wind observations	3D	1 h	26	100.00
SCAT	Scatterometer wind observations	2D	1 h	2	99.76
GNSS	Global Navigation Satellite System radio occultation observations	3D	1 h	26	100.00
AIRCRAFT	Aircraft observations	3D	1 h	39	97.40

A.3 Observation Encoding Details

For source s, observation values, validity masks, and metadata have shapes

$$V_t^s \in \mathbb{R}^{B\times N^s\times d^s}, \qquad M_t^s \in \{0,1\}^{B\times N^s\times d^s}, \qquad R_t^s \in \mathbb{R}^{B\times N^s\times r^s}. \tag{15}$$

Here, B is batch size, N^s is the number of observations, and d^s and r^s are the value and metadata dimensions. Horizontal locations are stored separately. To prepare profile observations for point encoding, we flatten tensors in $\mathbb{R}^{B\times N^s\times C\times L}$ into $\mathbb{R}^{B\times N^s\times d^s}$, where C and L count variables and

vertical levels and $d^s = CL$. This groups all variables and levels at each horizontal location into one vector, with a fixed order preserving their identities.

The point encoder, shown as MLP in Figure 1(b), then processes the masked observation values together with validity masks, metadata, the fraction of valid variables, and a learned embedding. The embedding is derived from platform identifiers or from observation values, masks, and metadata. A separate projection of the masked values and validity masks is added to the nonlinear encoding to produce point features $h^s_{t,j}$.

These point features are aggregated onto nearby grid locations using learned nonnegative weights conditioned on observation features, metadata, and relative position, producing $\widetilde{O}^s_t$. Longitude indices wrap across the grid boundary, allowing observations near either edge to contribute to neighboring cells on the opposite edge. Grid locations outside the latitude range are excluded.

During this aggregation, we also record grid occupancy I^s_t, the number of valid observations mapped to each grid location, N^s_t, and the sum of their mapping weights, W^s_t. Convolution and patch projection combine these quantities with $\widetilde{O}^s_t$ to produce source grid features O^s_t:

$$O^s_t = \mathrm{PatchProj}^s\Big(\mathrm{Conv}^s\Big([\widetilde{O}^s_t, I^s_t, \log(1+N^s_t), \log(1+W^s_t)]\Big)\Big). \tag{16}$$

As illustrated in Figure 1(a), averaging I^s_t within each patch gives coverage C^s_t, the fraction of grid locations receiving valid observations. This coverage weights each source's local contribution during fusion.

A.4 Backbone and Decoder Architecture

The analysis backbone receives the combined observation representation and latent background as a tensor in $\mathbb{R}^{768\times 90\times 180}$. Its 18 Transformer blocks preserve the spatial resolution. Attention windows span ten positions along each spatial axis and shift by five positions in alternate blocks. Each block uses grouped-query attention (Ainslie et al., 2023) with 16 query heads and four key/value heads. It also uses RMSNorm (Zhang & Sennrich, 2019), rotary position embeddings (Su et al., 2024) along both spatial axes, and a SwiGLU feedforward network (Shazeer, 2020) with hidden width 3072. An embedding of the hour and day of year scales and offsets normalized features and controls residual branch gains in the attention and feedforward layers. A final RMSNorm produces the latent analysis state S_t. The decoder applies a convolution with kernel size three and SiLU, then a transposed convolution with kernel size and stride two. This restores the 1° grid and produces the analysis field x^a_t with all variables.

A.5 Baseline Configuration and Evaluation

We evaluate HealDA with its released implementation, pretrained weights, and observation preprocessing pipeline (Gupta et al., 2026), without retraining, fine-tuning, or further checkpoint selection. The model was trained on UFS Replay observations and ERA5 targets from 2000 to 2021. Inputs include AMSU-A, AMSU-B, ATMS, MHS, conventional surface and upper-air observations, GNSS radio occultation, scatterometer winds, and atmospheric motion vectors. Both observation windows in the main text use the same weights, following the published protocol for window sensitivity. HealDA produces an analysis field with 74 variables. We evaluate the 69 variables shared with StreamDA on a 1° latitude–longitude grid, excluding TCWV, U100M, V100M, SST, and SIC from HealDA's output.

A.6 Initialization and Evaluation Protocols

Algorithms 1–4 describe the hourly update and three inference protocols. Each update uses the observations assigned to that hour. StreamDA and StreamDA (6 h rollout) share a checkpoint; Window-6h is trained separately. Selecting continuous outputs every six hours changes only the evaluation times, not the assimilation trajectory.

At initialization, the model projects and normalizes its learned spatial embedding to obtain S_{init}. The first update bypasses MTF module and stores the resulting analysis state as the only valid memory entry. Subsequent updates mask unavailable entries and set the tendency to zero until two analysis states are available.

Algorithm 1 Single hourly update with cold start

Require: Trained StreamDA model and initial embedding S_{init}
Ensure: Analysis field, updated state, and memory
function STEP($\mathcal{O}_t, S, \mathcal{M}$)
  $F_{o,t} \leftarrow \text{CMOF}(\mathcal{O}_t)$
  if $S = \emptyset$ **then** ▷ Cold start
    $S_t^- \leftarrow S_{\text{init}}$ ▷ Bypass MTF
  else
    $H_{t-1} \leftarrow \text{MemoryEnhancement}(S, \mathcal{M})$ ▷ Use valid memory entries
    $S_t^- \leftarrow S + \text{LatentPredictor}(H_{t-1})$
  end if
  $S_t \leftarrow \text{Backbone}(S_t^- + F_{o,t})$
  $x_t^a \leftarrow \text{Decode}(S_t)$
  Append S_t to $\mathcal{M}$; retain the latest four valid states
  return $(x_t^a, S_t, \mathcal{M})$
end function

Algorithm 2 Continuous assimilation: StreamDA

Require: Hourly observations $\{\mathcal{O}_t\}$ and STEP from Algorithm 1
Ensure: Hourly analysis fields $\{x_t^a\}$
$S \leftarrow \emptyset$; $\mathcal{M} \leftarrow \emptyset$ ▷ Initialize once
for each successive hour t **do**
  $(x_t^a, S, \mathcal{M}) \leftarrow \text{STEP}(\mathcal{O}_t, S, \mathcal{M})$
  Save x_t^a ▷ Carry state and memory to the next hour
end for
if evaluation is every six hours **then**
  Select saved outputs at 00, 06, 12, and 18 UTC
end if

Algorithm 3 Independent sequences: StreamDA (6 h rollout)

Require: Hourly observations $\{\mathcal{O}_t\}$ and STEP from Algorithm 1
Ensure: Endpoint analysis fields $\{x_t^a\}$
for each hourly target t **do**
  $S \leftarrow \emptyset$; $\mathcal{M} \leftarrow \emptyset$ ▷ Restart for each target
  for $u = t - 5, \ldots, t$ **do**
    $(x_u^a, S, \mathcal{M}) \leftarrow \text{STEP}(\mathcal{O}_u, S, \mathcal{M})$
  end for
  Save x_t^a ▷ Retain only the endpoint
end for

Algorithm 4 Window-6h inference

Require: Hourly observations $\{\mathcal{O}_t\}$ and trained window model
Ensure: Endpoint analysis fields $\{x_t^a\}$
for each target t **do**
  $x_t^a \leftarrow \text{WindowModel}(\mathcal{O}_{t-5:t})$ ▷ Joint window processing
  Save x_t^a ▷ No state passed between targets
end for

B ADDITIONAL ANALYSIS AND FRAMEWORK RESULTS

B.1 RELATIVE RMSE ACROSS ALL VARIABLES

The main text reports selected variables and overall improvements. Here, we examine whether these gains extend across variables, pressure levels, and evaluation times. Figure 6 compares all variables.

For variable v at time t, the relative RMSE change, labeled NRMSE in the figure, is

$$\mathrm{NRMSE}_{v,t} = \left(\frac{\mathrm{RMSE}_{\mathrm{StreamDA},v,t}}{\mathrm{RMSE}_{\mathrm{HealDA},v,t}} - 1 \right) \times 100\%. \tag{17}$$

Negative values indicate lower RMSE for StreamDA, and positive values indicate higher RMSE. StreamDA has lower RMSE in **94.0%** of variable and time combinations. The mean and median of these changes across variables and evaluation times are **-17.2%** and **-14.6%**, respectively. Geopotential shows the largest improvement, with a mean relative change of **-35.1%** across levels. T2M and V10M improve at every evaluated time, and U10M improves at all but one. U50 and U100 have higher errors on average. The gains therefore cover most variables and times.

B.2 Downstream Forecast Performance

We test whether improved analysis accuracy translates into better subsequent forecasts by initializing the same Pangu model (Bi et al., 2023) with analysis fields from StreamDA and HealDA*. HealDA* excludes future observations, matching the observation availability required for operational use. Forecasts start every six hours over the model's test period. Both sets of analysis fields are interpolated from 1° to Pangu's 0.25° grid to generate ten-day forecasts. We evaluate all forecasts against 0.25° ERA5 using latitude-weighted RMSE.

Figure 7 compares Z100, T300, and Q50, representing geopotential, temperature, and specific humidity. StreamDA initialization gives lower RMSE for all three variables at all forty evaluated lead times. The differences in Z100 and Q50 persist at longer leads, while the T300 advantage is clearest at short and medium ranges.

Figure 8 extends the comparison to all variables, showing the percentage with lower RMSE under StreamDA initialization at each forecast lead. This percentage exceeds **80%** at every evaluated lead from one to ten days. It reaches **87.0%**, **91.3%**, and **92.8%** at three, five, and seven days, respectively, and remains at **85.5%** at ten days. Averaging relative reductions equally across variables gives mean RMSE reductions of **8.95%**, **5.27%**, and **3.31%** at one, three, and five days, respectively. These results show that StreamDA provides better initial conditions for Pangu across most evaluated variables, extending its analysis gains to downstream forecasting.

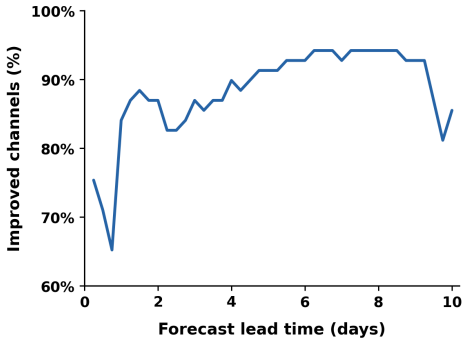


Figure 8: Percentage of the Pangu forecast variables with lower RMSE under StreamDA than HealDA* initialization across lead times.

B.3 Training and Inference Efficiency

Beyond analysis accuracy, we compare the computational costs of StreamDA, StreamDA (6 h rollout), and Window-6h to assess whether carrying states across hourly updates reduces repeated processing of observation windows. Table 5 reports parameter counts, training memory per device during steady operation, and inference time. Inference time is measured on an NVIDIA A100 GPU using a batch size of 1. The measured time includes reading input data, computing the analysis field, and writing the output.

Window-6h has 9.10% fewer parameters than StreamDA but uses **17.96%** more training memory per device. Under the measured conditions, StreamDA takes approximately **27%** of Window-6h's inference time and **21%** of StreamDA (6 h rollout)'s. Despite its larger parameter count than the window model, StreamDA is faster per output. This supports the computational benefit of carrying states forward and avoiding repeated window processing (Algorithms 2–4).

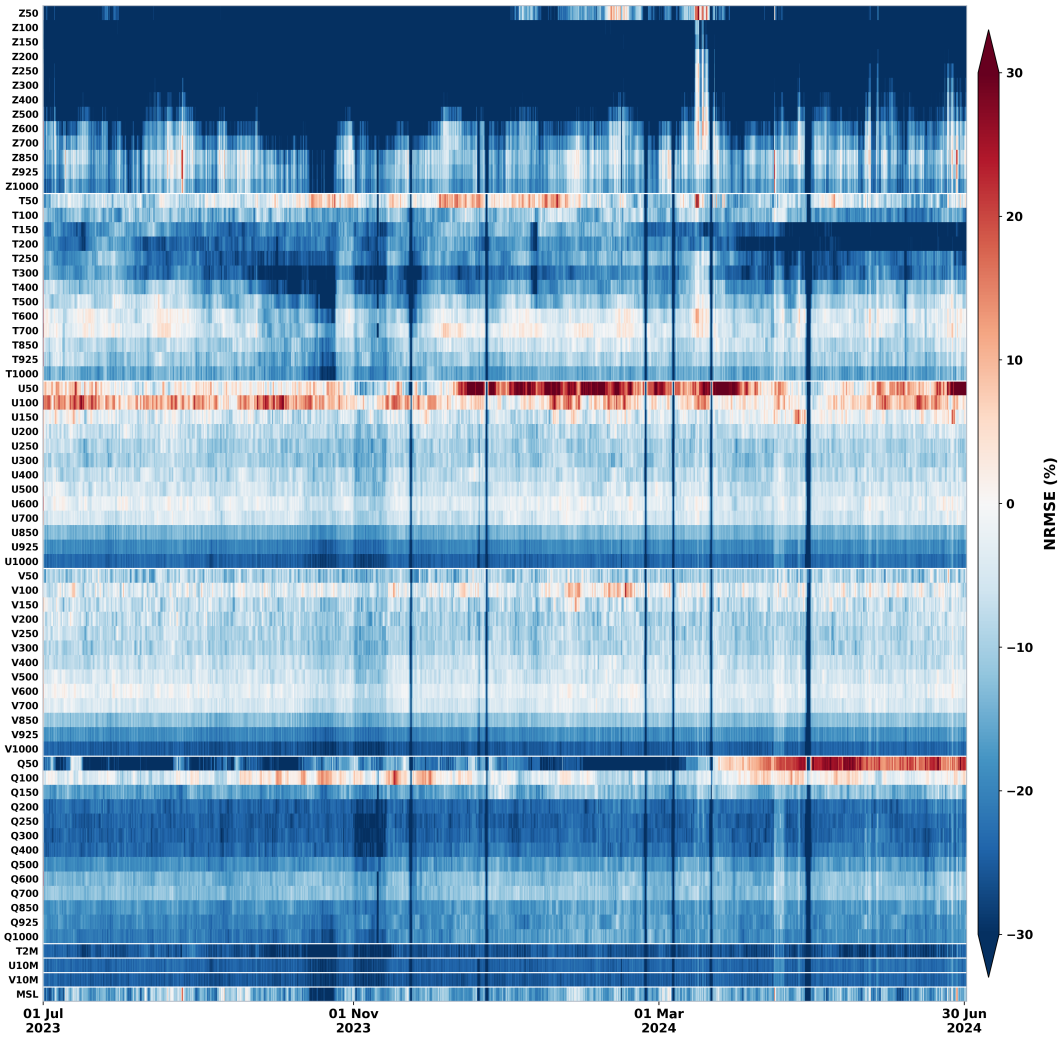


Figure 6: Relative RMSE changes of StreamDA against HealDA across 69 variables. Rows list variables, and columns show evaluation times spaced six hours apart from July 2023 through June 2024. Blue indicates lower StreamDA RMSE and red indicates higher RMSE.

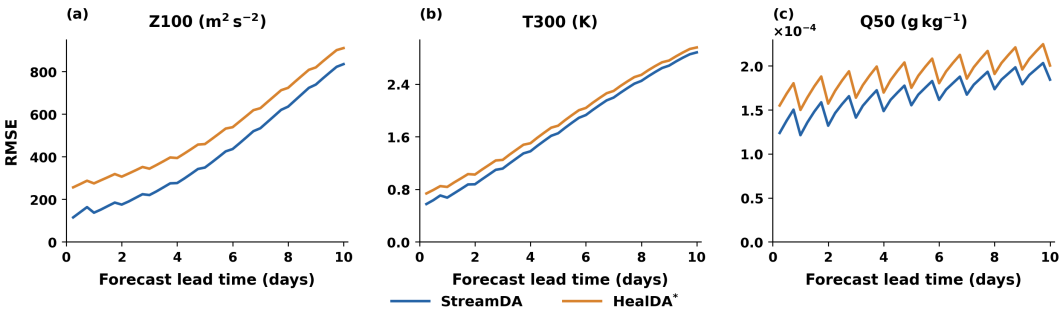


Figure 7: Pangu forecast RMSE under different analysis initializations. Panels (a)–(c) show Z100, T300, and Q50, respectively. Horizontal axes show forecast lead time in days, and vertical axes show RMSE in the units above each panel. Blue and orange curves denote StreamDA and HealDA* initializations, respectively.

Table 5: Computational costs of the three assimilation configurations. Rows list the configurations; columns give parameter counts, training GPU memory per device, and mean inference time per analysis field.

Method	Parameters (M)	Training GPU memory (GiB/device)	Inference time (s/output)
StreamDA	263.29	46.89	0.669
StreamDA (6 h rollout)	263.29	46.89	3.131
Window-6h	239.34	55.31	2.484

C Additional Ablation Studies

C.1 Memory-based Temporal Fusion

The main text evaluates MTF as a whole; here, we separate the contributions of history retrieval and the tendency term. Table 6 evaluates these terms in Equation 9. Setting $W_c = 0$ removes retrieval; setting $W_\Delta = 0$ removes tendency. All configurations retain the latent predictor and carry the latest analysis state across hourly updates. The w/o MTF configuration removes both terms and feeds the latest state directly to the predictor, with $H_{t-1} = S_{t-1}$.

Adding tendency alone lowers RMSE across all displayed variables, and adding retrieval further lowers these errors. When retrieval is retained, the tendency term gives small, mixed changes. Historical retrieval thus provides a clearer additional benefit, while the recent state difference offers a smaller contribution that depends on the variable.

Table 6: MTF component ablations. Rows compare StreamDA with settings that remove the retrieval term, the tendency term, or both (w/o MTF). Columns report mean analysis RMSE for nineteen variables. Pressure-level entries follow 300/500/850 hPa, with units below variable names. Lower values are better.

Setting	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
w/o MTF	66.79/50.53/37.42	0.52/0.57/0.81	1.97/1.79/1.25	2.00/1.73/1.24	0.049/0.28/0.70	0.84	0.69	0.69	52.60
w/o retrieval term ($W_c = 0$)	63.34/49.12/36.56	0.50/0.55/0.80	1.94/1.75/1.23	1.97/1.71/1.23	0.048/0.28/0.69	0.83	0.67	0.68	51.21
w/o tendency term ($W_\Delta = 0$)	61.20/47.65/36.22	0.50/0.54/0.79	1.92/1.74/1.22	1.96/1.70/1.22	0.048/0.28/0.68	0.82	0.67	0.68	50.93
StreamDA	61.99/47.59/36.04	0.50/0.55/0.79	1.92/1.74/1.22	1.95/1.69/1.22	0.048/0.27/0.68	0.82	0.67	0.68	50.80

History Aggregation. We next examine whether our retrieval method improves analysis accuracy compared with alternative ways of using historical information. We replace the retrieved context with either the mean of the stored states (Mean pooling) or their concatenation followed by an MLP that maps back to the latent state dimension (Concatenation + MLP). Table 7 compares these alternatives with StreamDA. Retrieval yields lower mean RMSE across all nineteen displayed variables. Across all 69 variables, StreamDA reduces RMSE by **1.10%** relative to Mean pooling and **0.84%** relative to Concatenation + MLP, with lower errors in 69 and 67 variables, respectively. These percentages average the relative reductions in mean hourly RMSE equally across variables. Conditioning retrieval on the latest state therefore gives a modest accuracy benefit over both alternative aggregation methods.

C.2 CMOF Component Ablations

The main text compares CMOF with uniform fusion. Here, we distinguish the roles of local coverage and learnable source scaling. The w/o CMOF configuration retains observation feature construction, including observation counts and accumulated weights, and replaces the fusion in Equation 6 with equal source weights and fixed normalization.

Table 8 tests coverage modulation and source scaling separately. Removing coverage sets $C_t^s = 1$ in Equation 6, while removing source scaling fixes $\alpha_s = 1$. Each configuration retains the other fusion component and leaves observation feature construction unchanged. Removing coverage produces

Table 7: MTF history aggregation ablations. Rows compare retrieval in StreamDA with averaging stored states and concatenating them before MLP projection. Columns report mean analysis RMSE for nineteen variables over all hourly test outputs. Pressure-level entries follow 300/500/850 hPa, with units below variable names. Lower values are better.

Method	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
StreamDA	61.99/47.59/36.04	0.50/0.55/0.79	1.92/1.74/1.22	1.95/1.69/1.22	0.048/0.27/0.68	0.82	0.67	0.68	50.80
Mean pooling	64.81/49.07/36.43	0.50/0.55/0.80	1.93/1.75/1.22	1.96/1.71/1.22	0.048/0.28/0.69	0.83	0.67	0.68	51.14
Concatenation + MLP	62.26/47.79/36.41	0.50/0.55/0.80	1.94/1.76/1.23	1.97/1.71/1.23	0.048/0.28/0.69	0.83	0.68	0.68	51.19

substantially larger errors than removing source scaling, particularly in geopotential and mean sea level pressure. Removing source scaling gives smaller changes relative to StreamDA and improves some variables. The comparison identifies local coverage as the more influential fusion component in these experiments: source weights shared across locations and times do not fully replace local coverage in the tested configurations.

Table 8: CMOF component ablations. Rows compare StreamDA with configurations that remove coverage modulation (C_t^s) or learnable source scaling (α_s) individually, followed by w/o CMOF using uniform fusion. Columns report mean analysis RMSE for nineteen variables. Pressure-level entries follow 300/500/850 hPa, with units below variable names. Lower values are better.

Method	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
StreamDA	61.99/47.59/36.04	0.50/0.55/0.79	1.92/1.74/1.22	1.95/1.69/1.22	0.048/0.27/0.68	0.82	0.67	0.68	50.80
w/o coverage ($C_t^s = 1$)	75.60/55.73/42.88	0.53/0.57/0.82	1.97/1.78/1.25	2.00/1.73/1.25	0.048/0.28/0.70	0.85	0.70	0.70	59.88
w/o source scaling ($\alpha_s = 1$)	60.00/47.67/36.16	0.50/0.55/0.80	1.93/1.74/1.23	1.96/1.70/1.23	0.048/0.28/0.69	0.83	0.68	0.68	51.17
w/o CMOF	73.17/54.71/41.78	0.52/0.57/0.81	1.95/1.77/1.24	1.98/1.72/1.24	0.048/0.28/0.69	0.85	0.69	0.69	58.40

C.3 Sensitivity to the Auxiliary Prior Loss Weight

We examine how strongly background supervision affects the accuracy of the final analysis field. Table 9 varies the auxiliary prior loss weight while holding all other settings fixed. Relative to the default $\lambda_{\text{prior}} = 0.1$, a weight of 0.01 lowers RMSE by 0.28% on average across all 69 variables. Weights of 0.5 and 1.0 increase RMSE by 1.75% and 2.81% on average, with higher errors in 68 and 69 variables, respectively. Small weights give similar overall accuracy, whereas stronger background supervision increases analysis errors in most variables. This comparison measures sensitivity to the loss weight; it does not test whether the auxiliary loss is necessary.

Table 9: Mean analysis RMSE for different auxiliary prior loss weights. Rows compare four values of λ_{prior}, with 0.1 as the default; columns correspond to nineteen evaluated variables. Pressure-level entries follow 300/500/850 hPa, with units below variable names. Lower values are better.

λ_{prior}	Z ($m^2 \cdot s^{-2}$)	T (K)	U ($m \cdot s^{-1}$)	V ($m \cdot s^{-1}$)	Q ($g \cdot kg^{-1}$)	T2M (K)	U10M ($m \cdot s^{-1}$)	V10M ($m \cdot s^{-1}$)	MSL (Pa)
0.01	62.70/47.81/35.97	0.50/0.54/0.78	1.91/1.72/1.21	1.95/1.69/1.21	0.048/0.28/0.68	0.82	0.67	0.67	50.75
0.1 (Default)	61.99/47.59/36.04	0.50/0.55/0.79	1.92/1.74/1.22	1.95/1.69/1.22	0.048/0.27/0.68	0.82	0.67	0.68	50.80
0.5	64.27/48.46/36.47	0.51/0.56/0.80	1.94/1.76/1.23	1.98/1.71/1.23	0.049/0.28/0.69	0.83	0.68	0.68	51.43
1.0	65.09/48.66/36.75	0.52/0.57/0.81	1.96/1.78/1.24	1.99/1.73/1.24	0.049/0.28/0.70	0.84	0.68	0.69	51.75

D Evaluation Under Observation Gaps

We evaluate StreamDA and w/o CMOF under recorded observation gaps and controlled source removal. Both evaluations use relative RMSE change (NRMSE), with the reference specified below. Negative values indicate lower error than the reference, and positive values indicate higher error.

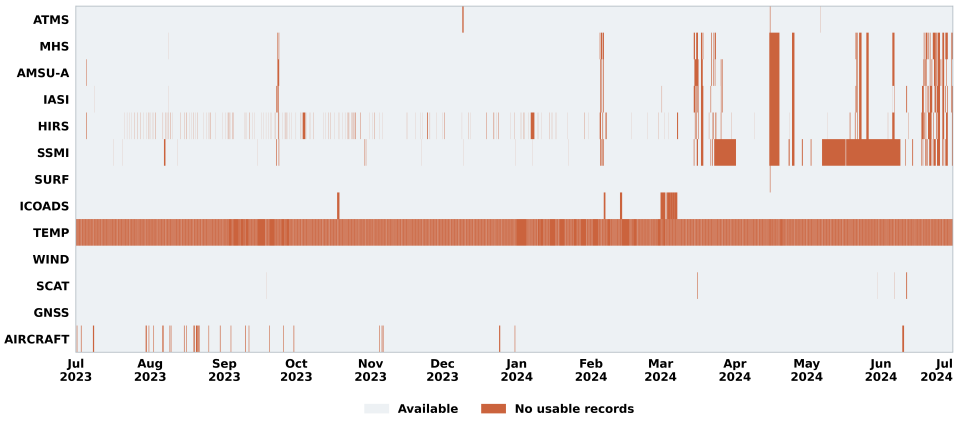


Figure 9: Hourly availability of all observation sources from July 2023 through June 2024. Rows list sources, and each column represents one hour. Gray indicates at least one usable observation, and orange indicates none. Date ticks are shown monthly.

D.1 Recorded Observation Availability and Analysis Accuracy

We examine whether the advantage under missing observations shown in the main text's individual cases also holds across the full test period. Figure 9 shows hourly availability for all thirteen sources. We exclude TEMP from the missingness criterion because its input interval is six hours. An hour has incomplete observations if at least one of the other twelve sources provides no usable observations.

We group hours by whether all twelve assessed sources are available. For each group and variable, we compute NRMSE relative to w/o CMOF:

$$\mathrm{NRMSE} = \left(\frac{\overline{\mathrm{RMSE}}_{\mathrm{StreamDA}}}{\overline{\mathrm{RMSE}}_{\text{w/o CMOF}}} - 1 \right) \times 100\%. \tag{18}$$

Here, $\overline{\mathrm{RMSE}}$ is averaged over the same selected hours for both models before the ratio is computed. We then average NRMSE equally across variables.

At least one of the twelve assessed sources is unavailable in 26.78% of evaluated hours. In these hours, mean NRMSE is **-8.91%**, with lower StreamDA errors in **68 of 69** variables. When all twelve sources are available, mean NRMSE is **-1.96%**. The larger advantage during incomplete hours is therefore present across the recorded test period, supporting the value of coverage modulation beyond the displayed cases.

D.2 Controlled Observation Removal

The recorded-gap evaluation above compares performance under naturally incomplete observations. To further assess sensitivity to observation loss, we deliberately remove sources and compare each model with its reference trajectory using the original inputs. We evaluate StreamDA and w/o CMOF on the twelfth day of each month from July 2023 through June 2024, giving twelve dates within the test period.

For each model and date, we first assimilate recorded observations for 24 h. We then copy its state and memory into a reference trajectory and thirteen independent removal trajectories. The reference keeps the recorded inputs; each removal trajectory disables the first k sources in a random ordering, for $k = 1, \ldots, 13$. Each date has a different ordering, shared by both models. Sources are disabled for eight hours (06–13 UTC), then restored for 24 h of further assimilation without resetting states.

For each model m, we compare its removal and reference trajectories at each variable and hour:

$$\mathrm{NRMSE}_m = \left(\frac{\mathrm{RMSE}_{m,\mathrm{removed}}}{\mathrm{RMSE}_{m,\mathrm{reference}}} - 1 \right) \times 100\%. \tag{19}$$

At each removal level, we average these ratios equally across all variables, the eight removal hours, and the twelve dates.

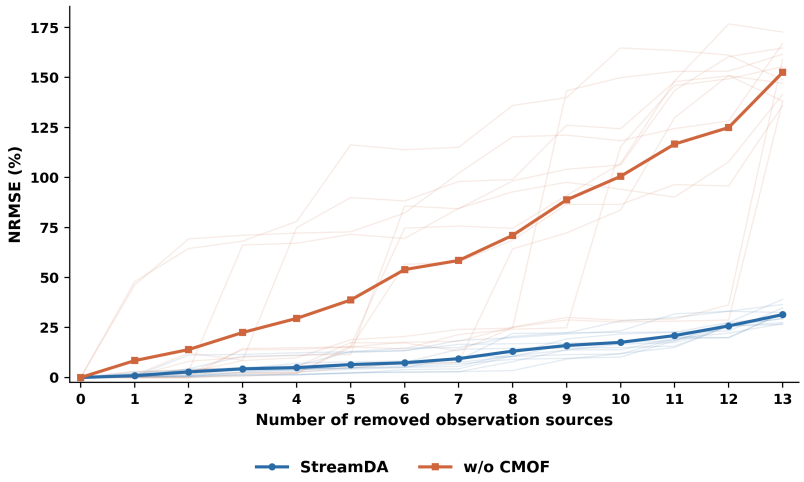


Figure 10: Analysis sensitivity to cumulative source removal. The horizontal axis gives the number of disabled observation sources; the vertical axis gives mean NRMSE (%) during the removal hours, averaged over all variables and the eight removal hours. Thin curves show twelve monthly cases, and thick curves with markers show their mean. Blue circles denote StreamDA and orange squares denote w/o CMOF.

StreamDA has lower mean NRMSE at all thirteen removal levels and in **144 of 156** date and removal combinations (Figure 10). Removing six sources raises RMSE by **7.37%** for StreamDA and 53.96% for w/o CMOF; removing all thirteen raises it by **31.44%** and 152.52%, respectively. With four or more sources removed, StreamDA has a smaller increase on every date. Under the tested removal combinations, CMOF limits the relative error increase as fewer observation sources remain available, complementing the evidence from recorded gaps.