# FourCastNet: A Neural Operator for Weather Forecasting

## A geometric approach to probabilistic machine-learning weather forecasting

Jikwang Kim

Seoul National University

May 31, 2026

# Outline

# Outline

# Weather Forecasting Models

- Numerical Weather Prediction (NWP)
  - Based on physical principles and mathematical equations
  - Requires significant computational resources
- Machine Learning (ML)
  - Using neural networks
  - Faster inference times and greater performance
  - Struggle with out-of-distribution events, physical consistency, and long-term stability
  - Especially for deterministic models, they often exhibit excessive smoothing
- Probabilistic ML
  - **GenCast**: using a denoising diffusion model approach, but it is computationally expensive
  - **AIFS-CRPS**: using a scoring rule (CRPS) to train a probabilistic model, but it can lead to build-up of small-scale noise

# Summary of FourCastNet 3

1. A probabilistic ML forecasting model, using hidden Markov model
2. It is based on spherical convolutions, which are more suitable for global weather forecasting
3. It uses a probabilistic loss function in the spectral domain

- A single forecast of 15 days is computed in 60 seconds on a single NVIDIA H100 GPU - a speedup of 8x over GenCast and 60x over IFS-ENS (golden standard for traditional NWP methods).
- It can retain stable predictions and accurate spectra with lead times up to 60 days.

# Outline

# Summary of FourCastNet 1

- Use the **Adaptive Fourier Neural Operator (AFNO)** architecture with a powerful **ViT** backbone.
- Learn a *resolution-invariant* operator $\mathcal{G}_\theta : X(k) \mapsto X(k+1)$ on the atmospheric state $X \in \mathbb{R}^{721\times1440\times20}$ (ERA5, $0.25^\circ$), then *roll out autoregressively*: $X_{\text{pred}}(j+i) = \mathcal{G}_\theta^{(i)}\big(X_{\text{true}}(j)\big)$.

# Neural Operator

**Standard neural network.** A map between *fixed, finite-dimensional* Euclidean spaces,

$$f_\theta : \mathbb{R}^n \to \mathbb{R}^m, \qquad v_{t+1} = \sigma\big(W v_t + b\big), \quad W \in \mathbb{R}^{d_{t+1} \times d_t}.$$

A spatial field is fed as values on a *fixed grid* of $n$ nodes; the weights are tied to that grid. Change the resolution $\Rightarrow$ $W$ no longer fits $\Rightarrow$ retrain.

**Neural operator.** A map between *infinite-dimensional function spaces*,

$$\mathcal{G}_\theta : \mathcal{A} \to \mathcal{U}, \qquad a : D \to \mathbb{R}^{d_a} \;\longmapsto\; u : D \to \mathbb{R}^{d_u},$$

that is *discretization-invariant*: *one* parameter set for *any* grid, with a well-defined continuum limit as the mesh refines.

# Neural Operator

**Replace the finite matrix product by its continuum analogue, a kernel integral:**

$$(Wv)_i = \sum_{j=1}^{n} W_{ij}\, v_j \quad \xrightarrow{n\to\infty} \quad \big(\mathcal{K}_\phi v_t\big)(x) = \int_D \kappa_\phi(x,y)\, v_t(y)\, \mathrm{d}y.$$

The kernel $\kappa_\phi$ does not depend on the grid, so the layer is meshfree. Everything else is the familiar *lift* $\to$ *iterate* $\to$ *project*:

$$\begin{aligned} v_0(x) &= P\big(a(x)\big) && \text{(pointwise lifting)} \\ v_{t+1}(x) &= \sigma\Big(W v_t(x) + \big(\mathcal{K}_\phi v_t\big)(x) + b\Big) && t = 0, \ldots, T-1 \\ u(x) &= Q\big(v_T(x)\big) && \text{(pointwise projection)} \end{aligned}$$

Then, how decide the kernel $\kappa_\phi$?

# Fourier Neural Operator (FNO)

**Key assumption.** Stationarity: $\kappa_\phi(x, y) = \kappa_\phi(x - y)$. Then the integral is a *convolution*, and by the convolution theorem

$$\big(\mathcal{K}_\phi v_t\big)(x) = \big(\kappa_\phi * v_t\big)(x) = \mathcal{F}^{-1}\Big(\mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v_t)\Big)(x) = \mathcal{F}^{-1}\Big(R_\phi \cdot \mathcal{F}(v_t)\Big)(x).$$

where $\mathcal{F}, \mathcal{F}^{-1}$ are the Fourier transform and inverse Fourier transform, respectively, and $*$ is the convolution operator, i.e. $(\kappa * v)(x) = \int \kappa(x - y)\, v(y)\, dy$.

Parameterize *directly in spectral space* by $R_\phi$ and **truncate** to the lowest $k_{\max}$ modes:

$$\big(R_\phi \cdot \mathcal{F} v_t\big)_k = \sum_{j=1}^{d_v} (R_\phi)_{k,\cdot,j}\, (\mathcal{F} v_t)_{k,j}, \qquad R_\phi \in \mathcal{C}^{k_{\max} \times d_v \times d_v},$$

with all modes $k > k_{\max}$ set to zero. One FNO layer:

$$v_{t+1}(x) = \sigma\Big(W v_t(x) + \mathcal{F}^{-1}\big(R_\phi \cdot \mathcal{F} v_t\big)(x)\Big).$$

# Aside: Convolution Theorem

**Fourier transform** and its inverse:

$$\hat{v}(\xi) = \mathcal{F}(v)(\xi) = \int v(x)\, e^{-2\pi i \langle \xi, x \rangle}\, dx, \quad v(x) = \int \hat{v}(\xi)\, e^{2\pi i \langle \xi, x \rangle}\, d\xi.$$

## Convolution Theorem

Convolution in space $\Longleftrightarrow$ *pointwise product* in frequency, i.e.:

$$\mathcal{F}(\kappa * v) = \mathcal{F}(\kappa) \cdot \mathcal{F}(v) \qquad (\text{and then } \kappa * v = \mathcal{F}^{-1}\big(\hat{\kappa} \cdot \hat{v}\big)).$$

**Proof.**

$$\begin{aligned}
\mathcal{F}(\kappa * v)(\xi) &= \int \left( \int \kappa(x-y)\, v(y)\, dy \right) e^{-2\pi i \langle \xi, x \rangle}\, dx \\
&= \int v(y) \Big( \int \kappa(x-y)\, e^{-2\pi i \langle \xi, x \rangle}\, dx \Big) dy \\
&= \underbrace{\int v(y)\, e^{-2\pi i \langle \xi, y \rangle} dy}_{\mathcal{F}(v)(\xi)} \underbrace{\int \kappa(u)\, e^{-2\pi i \langle \xi, u \rangle} du}_{\mathcal{F}(\kappa)(\xi)},
\end{aligned}$$

where we substitute $u = x - y$, and so $e^{-2\pi i \langle \xi, x \rangle} = e^{-2\pi i \langle \xi, u \rangle} e^{-2\pi i \langle \xi, y \rangle}$.

# Aside: DFT and FFT

On a grid, the transform is the **Discrete Fourier Transform**. For the feature vector $v \in \mathcal{C}^N$, with $\omega = e^{-2\pi i/N}$,

$$\hat{v}_k = \sum_{n=0}^{N-1} v_n \, \omega^{kn}, \qquad v_n = \frac{1}{N} \sum_{k=0}^{N-1} \hat{v}_k \, \omega^{-kn}, \qquad k = 0, \ldots, N-1.$$

That is, we can transform with the matrix multiplication $\hat{v} = Fv$ with $F_{kn} = \omega^{kn}$ a dense (unitary up to scale) matrix $\Rightarrow$ naive cost $O(N^2)$.

The **Fast Fourier Transform** recursively splits even/odd indices using $\omega_N^2 = \omega_{N/2}$ (butterfly operation), giving $\boxed{O(N \log N)}$ instead of $O(N^2)$.

**Discrete (circular) convolution theorem:** $\widehat{(\kappa \circledast v)}_k = \hat{\kappa}_k \, \hat{v}_k$ (periodicity is natural on the globe).

# Prelim: Vision Transformer (ViT)

**Tokenization.**

1. **Patch embedding.** Split the $H \times W$ field into $P \times P$ patches, flatten each into a $P^2C$ vector, and linearly embed each into $\mathcal{R}^d$ where $d$ is model dimension (Patch = token).
2. **Positional encoding.** Add a positional encoding, since the self-attention is permutation-invariant.

$\Rightarrow$ Token matrix $X \in \mathbb{R}^{N \times d}$, $N = hw = HW/P^2$.

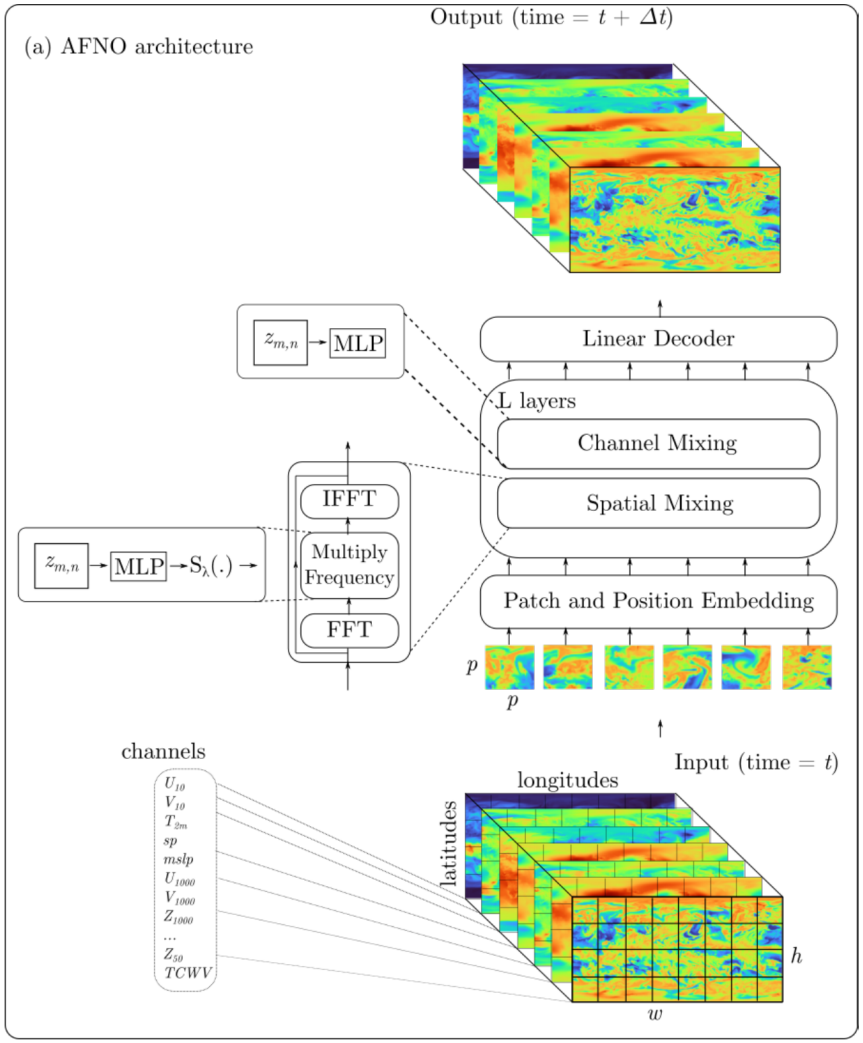
**Self-attention Encoder.** (Cost: $O(N^2 d)$ — quadratic in tokens.)
With $Q = XW_Q,\ K = XW_K,\ V = XW_V$ for learnable $W_Q, W_K, W_V \in \mathbb{R}^{d \times d_k}$, the attention is

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \qquad \big[\mathrm{softmax}(A)\big]_{ij} = \frac{e^{A_{ij}}}{\sum_r e^{A_{ir}}}.$$

Multi-head ($H$ heads), then LayerNorm + each Self-attention(Spatial Mixing) and MLP(Channel Mixing) + residuals per block.

**Linear decoder.** A linear layer to project back to the original dimension, then reshape to $H \times W \times C$.

# Adaptive Fourier Neural Operator with ViT Backbone




- FNO w/ ViT

  Replace self-attention ($O(N^2 d)$) as the spatial mixer with the FNO layer ($O(N \log N)$).

- Adaptive FNO w/ ViT

  In spatial mixer, apply **Weight sharing + nonlinearity**, **Block-diagonal weights** and **Soft-thresholding** to the Fourier coefficients.

# Adaptive Fourier Neural Operator with ViT Backbone

Let

$$z = \mathrm{DFT}(X) \in \mathbb{C}^{h \times w \times d}, \qquad X \in \mathbb{R}^{h \times w \times d}.$$

**Weight sharing + nonlinearity** — a 2-layer MLP, shared over all tokens $(m, n)$:

$$\tilde{z}_{m,n} = \mathrm{MLP}(z_{m,n}) = W_2\, \sigma\big(W_1 z_{m,n} + b_1\big) + b_2.$$

where $W_1, W_2 \in \mathbb{R}^{d \times d}$, and $\sigma = \mathrm{GELU}$ (old FNO: $\tilde{z}_{m,n} = R_\phi \cdot z_{m,n}$ with $R_\phi \in \mathbb{C}^{k_{\max} \times d \times d}$).

**Block-diagonal weights** (split $d$ channels into $k$ blocks):

$$W_1, W_2 = \mathrm{blockdiag}\left(W^{(1)}, \ldots, W^{(k)}\right), \quad W^{(\ell)} \in \mathbb{C}^{\frac{d}{k} \times \frac{d}{k}}.$$

**Soft-thresholding** to promote sparsity in frequency space:

$$\bar{z}_{m,n} = S_\lambda(\tilde{z}_{m,n}), \qquad S_\lambda(x) = \mathrm{sign}(x)\ \max\big(|x| - \lambda,\, 0\big),$$

not a hard truncation of the frequency coefficients (like LASSO regression, then adaptive).

# Final Architecture and Training

**Backbone.** $L = 12$ AFNO blocks, patch $p$=8, embed $d$=768, $n_b$=8 blocks, $\lambda$=$10^{-2}$, GELU. Linear decoder reconstructs the next frame:

$$\mathcal{G}_\theta : \; X(k) \;\longmapsto\; X(k{+}1), \qquad \Delta t = 6 \text{ h}.$$

**Two-stage objective.**

$$\mathcal{L}_{\text{pre}}(\theta) = \mathbb{E}_k \left\| \mathcal{G}_\theta\big(X(k)\big) - X_{\text{true}}(k{+}1) \right\|^2,$$

$$\mathcal{L}_{\text{ft}}(\theta) = \mathbb{E}_k \Big[ \left\| \mathcal{G}_\theta(X(k)) - X_{\text{true}}(k{+}1) \right\|^2 + \left\| \mathcal{G}_\theta^{(2)}(X(k)) - X_{\text{true}}(k{+}2) \right\|^2 \Big],$$

where $\mathcal{G}_\theta^{(2)} = \mathcal{G}_\theta \circ \mathcal{G}_\theta$ (feeds its own output back).

**Precipitation Model.** A *separate* AFNO head $\mathcal{G}_\theta^p$ diagnoses total precipitation from the (frozen) backbone output:

$$p(k{+}1) = \mathcal{G}_\theta^p\big(X(k{+}1)\big), \qquad \text{where } p = \log\big(1 + p/\epsilon\big), \quad \epsilon = 10^{-5}.$$

Trained on the log-transformed target (sparse, zero-inflated field); a final ReLU enforces $p \geq 0$.

**Ensembles (Monte-Carlo over initial conditions).** Perturb the standardized IC and roll out:

$$X^{(e)}(k) = \hat{X}_{\text{true}}(k) + \sigma\, \xi^{(e)}, \quad \xi^{(e)} \sim \mathcal{N}(0,1),\ \sigma = 0.3,\ e = 1, \ldots, E.$$

The ensemble mean beats the control at longer lead times.

# Outline

# Summary of FourCastNet 2

- Replace the FNO with a **Spherical Fourier Neural Operator (SFNO)**, which is more suitable for spherical data settings.
- Respecting spherical geometry ($SO(3)$-equivariance via the SHT) yields **stable autoregressive rollouts for a full year** (1,460 steps), vs. $\sim$25 days (100 steps) for the FFT-based FNO that distorts at the poles.

# Aside: Group Theory

**Motivation:** FourCastNet's FFT secretly assumes a *flat* world. The natural symmetry on a flat domain is **translation**; on the globe it is **rotation**.

**Definition:**

- **Group** $G$: a set of transformations closed under composition, with identity $e$ and inverses.
  e.g.: translations $T_a\, x = x + a,\ a \in \mathbb{R}^2$, and rotations $Rx,\ R \in \mathrm{SO}(3)$
- Group **action** $\Phi : G \times X \to X,\ (g, x) \mapsto g \cdot x$: "applies" $g$ to a point, obeying $e \cdot x = x$ and $(gh) \cdot x = g \cdot (h \cdot x)$.
- **Symmetry** of a space $X$: a transformation $g$ preserving its structure (distances $d$), i.e., $d(g \cdot x,\ g \cdot y) = d(x, y)$, and it forms a **symmetry group**.
  e.g. translations $(\mathbb{R}^2, +)$ for $\mathbb{R}^2$, rotation $\mathrm{SO}(3)$ for $\mathbb{S}^2$.
- **Commutativity**: $g \cdot h = h \cdot g$ for all $g, h \in G$. If so, $G$ is **abelian**.
  e.g. translations commute ($T_a\, T_b = T_b\, T_a$), but rotations do not ($R_1 R_2 \neq R_2 R_1$).

$\Rightarrow$ This mismatch (flat FFT vs. curved geometry) is why FFT-based models distort at the poles.

# Aside: Group Theory

**Equivariance.** (What we actually want)
The forecast operator $F$ (current state $\mapsto$ next state) should commute with rotations:

$$\Phi_R\big(F[u]\big) = F\big(\Phi_R[u]\big) \qquad \forall R \in SO(3).$$

*In words:* rotating the globe **then** forecasting one step = forecasting one step **then** rotating. The physics does not depend on how we orient the planet.

$\Rightarrow$ Replace the FFT with a **Spherical Harmonic Transform (SHT)**.

# Spherical Harmonic Transform (SHT)

**Setting.** We work with square-integrable functions on the sphere, equipped with the area-weighted inner product

$$\langle u, v \rangle_{L^2(\mathbb{S}^2)} = \int_{\mathbb{S}^2} \overline{u}\, v \, d\Omega = \int_0^{2\pi} \int_0^{\pi} \overline{u(\theta, \varphi)}\, v(\theta, \varphi)\, \sin\theta \, d\theta \, d\varphi.$$

- $\theta \in [0, \pi]$: **colatitude** (north pole $\theta = 0$, south pole $\theta = \pi$).
- $\varphi \in [0, 2\pi]$: **longitude**.
- $\overline{u}$: complex conjugate (functions may be complex-valued).
- $d\Omega = \sin\theta \, d\theta \, d\varphi$: Lebesgue measure on the sphere, — near the poles ($\sin\theta \to 0$), the true area shrinks.

This induces the norm $\|u\| = \sqrt{\langle u, u \rangle}$ and the Hilbert space $L^2(\mathbb{S}^2)$.

# Spherical Harmonic Transform (SHT)

$$Y_l^m(\theta,\varphi) = \underbrace{(-1)^m c_l^m \, P_l^m(\cos\theta)}_{\text{latitude } (\theta)} \cdot \underbrace{e^{im\varphi}}_{\text{longitude } (\varphi)}, \qquad c_l^m = \sqrt{\tfrac{2l+1}{4\pi}\tfrac{(l-m)!}{(l+m)!}}.$$

Reading the two factors is the key:

- **Longitude** $e^{im\varphi}$: $\varphi$ wraps around a circle, so this is just a **plane wave** — handled by the ordinary **FFT**.
- **Latitude** $P_l^m(\cos\theta)$: associated Legendre polynomial; the genuinely non-flat part, needing a separate **quadrature**, not an FFT.
- $c_l^m$: a normalization constant making each $Y_l^m$ unit-size.

- **degree** $l \in \{0, 1, 2, \dots\}$: the *frequency* w.r.t. the latitude.
- **order** $m \in \{-l, \dots, l\}$: zonal wavenumber (oscillations around longitude); $2l+1$ values per $l$.

**Orthonormality** (the decisive property)

$$\langle Y_l^m,\, Y_{l'}^{m'} \rangle_{L^2(\mathbb{S}^2)} = \delta_{ll'}\, \delta_{mm'}.$$

$\Rightarrow \{Y_l^m\}$ is an **orthonormal basis** of $L^2(\mathbb{S}^2)$.

**Proof.** With $Y_l^m = (-1)^m c_l^m P_l^m(\cos\theta)\, e^{im\varphi}$, the integrand *separates*:

$$\langle Y_l^m, Y_{l'}^{m'} \rangle = (-1)^{m+m'} c_l^m c_{l'}^{m'} \underbrace{\int_0^{2\pi} e^{i(m'-m)\varphi} d\varphi}_{I_\varphi} \underbrace{\int_0^{\pi} P_l^m P_{l'}^{m'} \sin\theta\, d\theta}_{I_\theta},$$

$$\text{(circle Fourier)} \quad I_\varphi = \begin{cases} 2\pi & m = m' \\ \left[ e^{i(m'-m)\varphi}/i(m'-m) \right]_0^{2\pi} = 0 & m \neq m'. \end{cases} = 2\pi\, \delta_{mm'},$$

$$I_\theta \overset{m=m'}{=} \int_0^{\pi} P_l^m(\cos\theta)\, P_{l'}^m(\cos\theta)\, \sin\theta\, d\theta$$

$$\overset{x=\cos\theta}{=} \int_{-1}^{1} P_l^m(x)\, P_{l'}^m(x)\, dx = \frac{2}{2l+1} \frac{(l+m)!}{(l-m)!}\, \delta_{ll'}. \qquad \text{(Legendre orthogonality)}$$

# Spherical Fourier Neural Operators (SFNO)

Hence the **Spherical Harmonic Transform (SHT)**:

$$u(\theta,\varphi)=\sum_{l\geq 0}\sum_{|m|\leq l}\hat{u}(l,m)\,Y_l^m,\qquad \hat{u}(l,m)=\langle Y_l^m,u\rangle=\int_{\mathbb{S}^2}\overline{Y_l^m}\,u\,d\Omega.$$

and the **spherical convolution**:

$$(\kappa * u)(x)=\int_{R\in \mathrm{SO}(3)}\kappa(Rn)\cdot u(R^{-1}x)\,dR=\int_{\mathbb{S}^2}u(x')\,\overline{\kappa(R_x^{-1}x')}\,d\mu(x').$$

where $n=(0,0,1)^T$ is the northpole, and $dR, d\mu(x')$ are the Haar measure on $\mathrm{SO}(3)$ and $\mathbb{S}^2$, respectively.

| | **FNO** (plane) | **SFNO** (sphere) |
|---|---|---|
| Basis | plane waves $e^{i2\pi\langle k,x\rangle}$ | spherical harmonics $Y_l^m$ |
| Frequency | $k$ (one index) | degree $l$ + order $m$ |
| Computation | FFT both directions | longitude: FFT; latitude: Legendre quad. |
| Convolution | $\mathcal{F}[\kappa * v](k)=\mathcal{F}[\kappa](k)\mathcal{F}[v](k)$ | $\mathcal{F}[\kappa * u](l,m)=2\pi\sqrt{\frac{4\pi}{2l+1}}\mathcal{F}[u](l,m)\mathcal{F}[\kappa](l,0)$ |
| Filters | $\mathcal{F}[\mathcal{K}_\vartheta[u]]=\tilde{\kappa}_\vartheta(k)\cdot\mathcal{F}[u](k)$ | $\mathcal{F}[\mathcal{K}_\vartheta[u]]=\tilde{\kappa}_\vartheta(l)\cdot\mathcal{F}[u](l,m)$ |
| Symmetry | translation-equivariant | rotation-equivariant $(SO(3))$ |
| Real signal | $\hat{u}(-k)=\overline{\hat{u}(k)}$ | $\overline{Y_l^m}=(-1)^m Y_l^{-m}$ |

**Theorem: Rotation equivariance of SHT**

For $\mathcal{K}_\vartheta[u] = \mathcal{F}^{-1}[\tilde{\kappa}_\vartheta \cdot \mathcal{F}u]$ with $\tilde{\kappa}_\vartheta = \tilde{\kappa}_\vartheta(l)$ (degree-only filter):

$$\mathcal{K}_\vartheta \circ \Phi_R = \Phi_R \circ \mathcal{K}_\vartheta \quad \Longleftrightarrow \quad \mathcal{K}_\vartheta[\Phi_R\, u] = \Phi_R[\mathcal{K}_\vartheta u] \quad \forall R \in SO(3).$$

**cf. FNO** (with translation $\Phi_T u(x,y) = u(x-a, y-b)$). Translation $\rightarrow$ phase in Fourier space:

$$\mathcal{F}[\Phi_T u](k) = \int u(x-a, y-b) e^{-i2\pi\langle k,x\rangle} dx = e^{-i2\pi\langle k,a\rangle}\, e^{-i2\pi\langle l,b\rangle}\, \mathcal{F}[u](k)$$
$$\equiv \rho_a(k)\, \mathcal{F}[u](k).$$

Then,

$$\mathcal{K}_\vartheta[\Phi_T u] = \mathcal{F}^{-1}\left[\tilde{\kappa}_\vartheta(k)\, \rho_a(k)\, \mathcal{F}u\right]$$
$$= \rho_a\, \mathcal{F}^{-1}\left[\mathcal{F}(\mathcal{K}_\vartheta u)\right] = \Phi_T[\mathcal{K}_\vartheta u] \quad (\rho_a \text{ is a scalar}).$$

**proof.** Sphere: replace scalar phase $\rho_a(k)$ by the Wigner matrix $D^l(R)$ (acting within each degree $l$). Since $\tilde{\kappa}_\vartheta(l)$ is $m$-independent, it commutes with $D^l(R)$ — giving equivariance. (The nonlinear filters break it like AFNO-style.)

# Final Architecture and Training

**Backbone.** $\mathcal{G}_\theta : \; u(k) \mapsto u(k{+}1)$ (encoder $\to$ $N$ SFNO blocks $\to$ decoder), $\Delta t = 6\,\mathrm{h}$.

- **SFNO block:** spherical Fourier layer (*linear*, $SO(3)$-equivariant) + point-wise MLP (channel mixing):

$$\mathcal{K}_\vartheta[u] = \mathcal{F}^{-1}\big[\tilde{\kappa}_\vartheta(l)\cdot \mathcal{F}u\big], \qquad u \;\mapsto\; \mathrm{MLP}\big(\mathcal{K}_\vartheta[u]\big).$$

- **Equivariant rescaling:** up/down-scaling done *by the SFNO block itself* — truncate SHT frequencies (down) / evaluate inverse SHT at higher resolution (up), *replacing* FourCastNet's non-equivariant patching / pixel-shuffle.

**Multi-stage objective** (latitude-weighted $L^2$, $F_\theta^{(s)} = F_\theta \circ \cdots \circ F_\theta$):

$$\mathcal{L}_{\text{pre}}(\theta) = \mathbb{E}_k \left\| \mathcal{G}_\theta\big(u(k)\big) - u_{\text{true}}(k{+}1) \right\|^2,$$

$$\mathcal{L}_{\text{ft}}(\theta) = \mathbb{E}_k \; \frac{1}{n_s} \sum_{s=1}^{n_s} \left\| \mathcal{G}_\theta^{(s)}\big(u(k)\big) - u_{\text{true}}(k{+}s) \right\|^2,$$

a single-step pre-train, then a multi-step fine-tune (increasing $n_s$, starting at $n_s = 2$) that feeds the model its own output back.

# Outline

# Summary of FourCastNet 3

- Augment the SFNO with **local DISCO convolutions** — anisotropic, locally-supported filters that the spectral route cannot represent.
- Turn it **probabilistic** via a hidden Markov model: condition on a spherical-diffusion latent $z_n$ to generate ensembles in one forward pass.
- Train with a **combined spatial + spectral CRPS** objective — sharp, calibrated members with faithful spectra.
- Scale to **1024+ GPUs** via spatial domain decomposition (model + data parallel), inspired by classical NWP.

# Revisit: SFNO filters

**Recall the spherical convolution theorem (SFNO block):**

$$\mathcal{F}[\kappa * u](l,m) = 2\pi\sqrt{\tfrac{4\pi}{2l+1}}\,\mathcal{F}[u](l,m)\,\mathcal{F}[\kappa](l,0).$$

$\Rightarrow$ Such a filter $\tilde{\kappa} = \tilde{\kappa}(l)$ is **isotropic** (not depends on the longitude $m$).

**Why?** $\mathbb{S}^2$ **is not a group.** A rotation needs three angles, but a point on $\mathbb{S}^2$ needs only two. Apply $R = R_z(\varphi)R_y(\theta)R_z(\psi)$ to the north pole $n = (0,0,1)^\top$, right to left:

$$R_z(\psi)\,n = n, \qquad R_y(\theta)\,n = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \sin\theta \\ 0 \\ \cos\theta \end{pmatrix},$$

$$R\,n = R_z(\varphi) \begin{pmatrix} \sin\theta \\ 0 \\ \cos\theta \end{pmatrix} = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \sin\theta \\ 0 \\ \cos\theta \end{pmatrix} = \begin{pmatrix} \cos\varphi\sin\theta \\ \sin\varphi\sin\theta \\ \cos\theta \end{pmatrix}.$$

$\Rightarrow\ \mathbb{S}^2 = SO(3)/SO(2).$

- Leftover $\psi$ = filter's **in-plane orientation**, but ambiguous (no global "up").
- Well-defined only if $\psi$-invariant = **axisymmetric** $\Rightarrow$ no anisotropic structure.

# DISCO convolution

**Idea.** Don't go through the spectral domain. Approximate the continuous spherical convolution with a **quadrature rule**.

$$(u \circledast \kappa)(x) = \int_{\mathbb{S}^2} u(x')\,\overline{\kappa(R_x^{-1}x')}\,d\mu(x') \;\approx\; \sum_{j=1}^{n_{\text{lat}}n_{\text{lon}}} \overline{\kappa(R_x^{-1}x_j)}\,u(x_j)\,\omega_j.$$

- $\{x_j\}$: grid points, $\{\omega_j\}$: spherical quadrature weights.

**DISCO : Discrete-Continuous convolution.**

1. **Precompute rotations (continuous).** For any output point $x_i$ and input grid point $x_j$, compute every rotation:

$$K_{ij} := \overline{\kappa(R_i^{-1}x_j)} \quad \Rightarrow \quad (u \circledast \kappa)(x_i) = \sum_j K_{ij}\,u(x_j)\,\omega_j.$$

   Compact support $\Rightarrow$ $K_{ij}$ is sparse $\Rightarrow$ $\mathcal{O}(n_{\text{lat}}n_{\text{lon}})$ (linear, no SHT).

2. **Learnable filter (basis expansion).** Parametrize $\kappa$ as a linear combination of fixed basis functions $\tilde{\kappa}_b$ with learnable weights $w_b$:

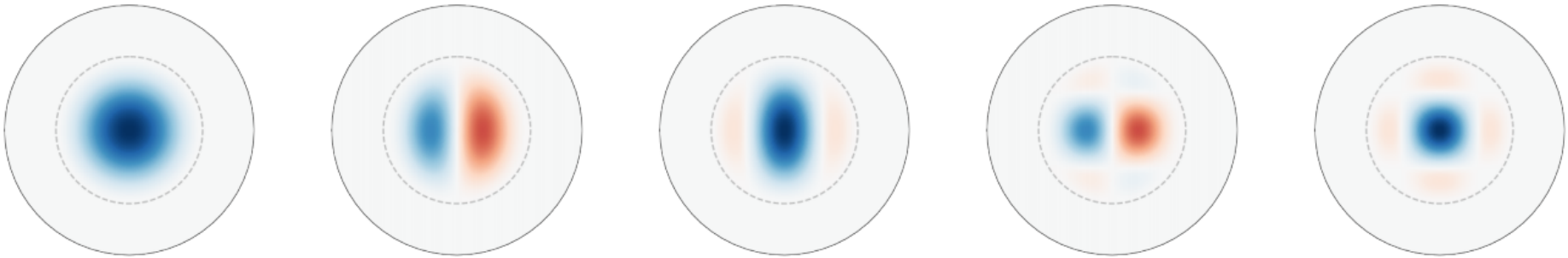
$$\kappa(x) = \sum_{b=1}^{n_{\text{basis}}} w_b\,\tilde{\kappa}_b(x).$$

# DISCO convolution

1. **Morlet wavelet basis.** On a compact disk centered at the North Pole, with normalized radius $r = \theta/\theta_{\text{cutoff}} \in [0, 1]$ and angle $\phi \in [0, 2\pi)$:

$$\tilde{\kappa}_{\ell m}(r, \phi) = h(r) \underbrace{e^{i\pi \ell\, r \sin\phi}}_{\text{freq. along } r\sin\phi} \underbrace{e^{i\pi m\, r \cos\phi}}_{\text{freq. along } r\cos\phi}, \qquad h(r) = \cos^2\left(\frac{\pi}{2} r\right).$$

   - **Hann window** : compact support (keeps $K_{ij}$ sparse) + smoothness (no high-freq. ringing).
   - **Two independent indices** $\ell, m$ set the oscillation along *two orthogonal disk axes* $r\sin\phi$ and $r\cos\phi$.



(a) $\ell = 0, m = 0$ (b) $\ell = 0, m = 1$ (c) $\ell = 0, m = 2$ (d) $\ell = 2, m = 1$ (e) $\ell = 2, m = 2$

# DISCO convolution

**Two key freedoms unlocked vs. the spectral (SFNO) route:**

- **Locally supported** (a small stencil), not globally smooth.
- **Anisotropic** — orientation is fixed analytically *before* discretizing, so the $\psi$-ambiguity never arises.

$$\underbrace{\hat{u}(l,m)\,\hat{\kappa}(l,0)}_{\text{global spectral conv (SFNO)}} \qquad \underbrace{\sum_j K_{ij}\,u(x_j)\,\omega_j}_{\text{local DISCO conv}}$$

isotropic, whole sphere at once

anisotropic, small neighborhood

| | **Global (spectral)** | **Local (DISCO)** |
|---|---|---|
| Support | global, smooth | compact stencil |
| Filter dep. | degree $l$ only | degree $\ell$ + order $m$ |
| Symmetry | isotropic | anisotropic |
| Captures | planetary waves | fronts, orographic / zonal flow |
| Classical analogue | pseudo-spectral (IFS) | finite differencing |
| Cost | SHT, $\mathcal{O}(n_{\text{lat}}^2 n_{\text{lon}} \log n_{\text{lon}})$ | $\mathcal{O}(n_{\text{lat}} n_{\text{lon}})$ |

FCN3 uses both, interleaved at a 4:1 local:global ratio — spanning planetary waves down to local fronts. (A transposed variant $u \circledast^{\dagger} \kappa$ serves as transposed conv. for upsampling.)

# Hidden Markov Model: the probabilistic forecasting framework

Not a single deterministic $u_{n+1} = F_\theta(u_n, t_n)$, but the conditional distribution $p(u_{n+1} \mid u_n, t_n)$.

**Hidden Markov model (HMM).** A Markov chain of hidden states $z_n$ drives the observable dynamics.

$$z_n \to z_{n+1} \quad \text{(transition)}, \qquad z_n \to u_n \quad \text{(emission)},$$

that is, we have defined $u_{n+1} = F_\theta(u_n, z_n, t_n)$.

If we can choose the proper latent distribution (or hidden dynamics) $p(z_{n+1} \mid z_n)$, then

$$F_\theta(u_n, z_n, t_n) \;\sim\; p(u_{n+1} \mid u_n, t_n).$$

Different noise draws $z_{n,e} \Rightarrow$ different ensemble members $\{u_{n+1,e}\}_{e=1}^{N_{\text{ens}}}$, all in one forward pass (no iterative denoising, unlike diffusion / GenCast).

# Hidden Markov Model: hidden dynamics

The latent field $z_n$ follows plausible dynamics in the (spherical) spectral domain, which carries the right spatio-temporal correlations. Assume an **AR(1) process**:

$$z_n = z(x, t_n) = \underbrace{\phi\, z(x, t_{n-1})}_{\text{temporal memory}} + \sum_{\ell} \sum_{|m| \le \ell} \underbrace{\sigma_\ell\, \eta_\ell\, Y_\ell^m(x)}_{\text{spatially correlated innovation}} \quad , \quad \eta_\ell \sim \mathcal{N}(0, 1).$$

- $\phi = e^{-\lambda}$ : the temporal correlation
- $\sigma_\ell = F_0\, e^{-\frac{k_T}{2}\ell(\ell+1)}$ : the spatial length scale (decay over degree $\ell$).
  - $\ell(\ell+1)$ : the eigenvalue of the spherical Laplacian (smoothing by diffusion).

  $\Rightarrow$ $\ell \Uparrow$ (high freq., small scale) $\rightarrow$ $\sigma_\ell \Downarrow$ $\rightarrow$ small-scale noise $\Downarrow$ (smooth field).
  - $k_T$ : spatial scale of the noise.

  $\Rightarrow$ $k_T \Uparrow$ $\rightarrow$ smoother noise (only large scales survive).
- FCN3 draws from 8 such processes with different $k_T$ — from very smooth to relatively rough noise — supplying atmospheric uncertainty across many spatial scales.

# Loss function

$$\theta^* = \arg\min_\theta \sum_n \mathcal{L}_{\text{ens}}\Big(\{F_\theta(u_n, z_{n,e}, t_n)\}_{e=1}^{N_{\text{ens}}},\ u^*_{n+1}\Big).$$

**Choice of $\mathcal{L}_{\text{ens}}$:** minimize the distance between the estimated distribution $F = p(u)$ and the true observation $u^*$. We use the CRPS:

$$\text{CRPS}(F, u^*) = \int_{\mathbb{R}} \big(F(u) - \mathbf{1}(u^* \le u)\big)^2 du = \underbrace{\mathbb{E}_F|U - u^*|}_{\text{accuracy}} - \underbrace{\tfrac{1}{2}\mathbb{E}_F|U - U'|}_{\text{spread}}.$$

- **vs. the old RMSE objective (SFNO):** a single deterministic prediction trained by

$$\mathcal{L}_{\text{RMSE}} = \mathbb{E}_n \big\| F_\theta(u_n) - u^*_{n+1} \big\|^2.$$

  RMSE rewards only the mean, so a blurry forecast (average of many futures) scores well.

**Trained end-to-end as an ensemble model:** $N_{\text{ens}}$ members are sampled and scored *jointly* every step — so the model learns the right spread, not just the mean.

# Combined CRPS loss

The naive objective averages CRPS pointwise over the sphere:

$$\mathcal{L}_{\text{spatial}} = \frac{1}{4\pi} \int_{\mathbb{S}^2} \text{CRPS}\big(\{u_e(x)\}, u^*(x)\big)\, d\mu(x).$$

Pointwise CRPS sees only the marginal CDF at each point, so can't catch the spatial correlation structure.

**Fix: add a CRPS term in the spectral domain.**

$$\mathcal{L}_{\text{spectral}} = \sum_{\ell=1}^{n_{\text{lat}}/2} \sum_{|m|\le\ell} \text{CRPS}\big(\{\hat{u}_\ell^m\},\, \hat{u}_\ell^{m,*}\big).$$

$$\mathcal{L}_{\text{ens}} = \sum_n \sum_c w_c\, w_{\Delta t,c}\, w_n \left(\mathcal{L}_{\text{spatial}} + \lambda_{\text{spectral}}\, \mathcal{L}_{\text{spectral}}\right).$$

- Channel weight $w_c$ + temporal weight $w_{\Delta t,c}$ (inverse std of 1-hourly differences) balance variables of very different magnitudes/timescales.
- Rollout weight $w_n$ averages over autoregressive lead times during fine-tuning.

# Final Architecture and Training

**(1) Direct state prediction.**
Not predict the tendency $u_{n+1} - u_n$ (a large skip connection). FCN3 predicts $u_{n+1}$ directly, since the skip connection amplifies high-freq. artifacts in long rollouts.

**(2) Channel-separated encoder / decoder.**
Variables (atmospheric, surface, auxiliary+noise) have very different spectral statistics — mixing them early entangles unrelated signals.

**(3) No layer normalization.**
Absolute magnitudes are physically meaningful, instead use He initialization + LayerScale to keep activations bounded.

**(4) Distributed training via domain decomposition.**
High-dimensional fields ($721 \times 1440$/variable) break LLM-style FSDP (Fully Sharded Data Parallel). Inspired by classical NWP, split both model and data spatially (lat $\times$ lon; distributed SHT & DISCO), plus ensemble/batch parallelism — scaling to 1024+ H100 GPUs.

# Results

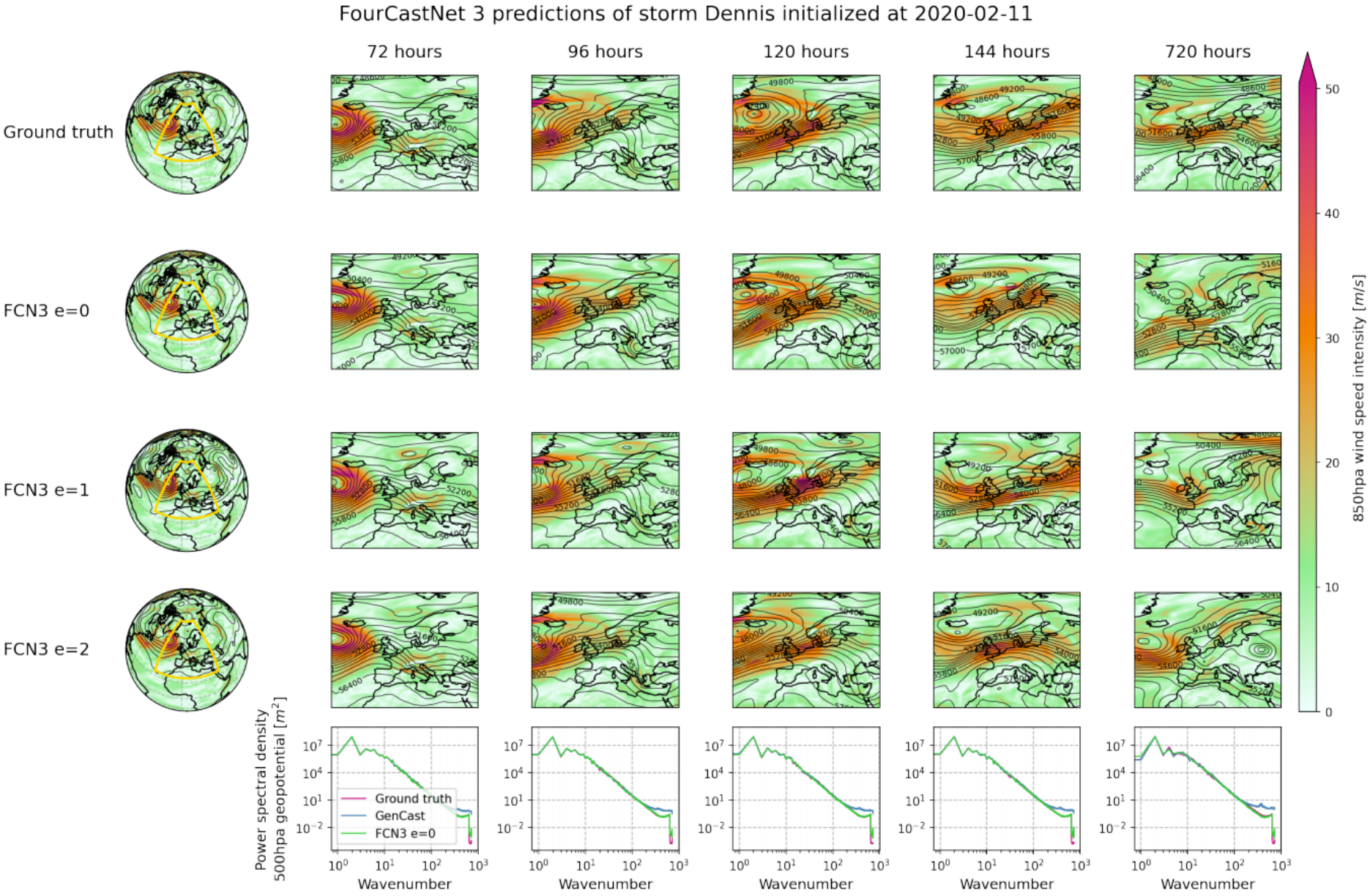
Evaluated on out-of-sample 2020, 50-member ensemble, against IFS-ENS and GenCast (WeatherBench 2).

- Beats IFS-ENS (the gold-standard physics ensemble) on CRPS & ensemble-mean RMSE.
- Matches GenCast (SOTA diffusion model) on 15/16 channels — at double the temporal resolution (6-hourly vs 12-hourly), despite training only up to 2016.

| **15-day forecast** | **Hardware** | **Time** |
|---|---|---|
| FCN3 | 1× H100 | ∼60 s |
| GenCast | Cloud TPU v5 | ∼8 min (8× slower) |
| IFS-ENS | 96 CPUs | ∼1 hr (60× slower) |

Also, well-calibrated, and guaranteed the spectral fidelity.

# Results


FourCastNet 3 predictions of storm Dennis initialized at 2020-02-11
72 hours
96 hours
120 hours
144 hours
720 hours
Ground truth
FCN3 e=0
FCN3 e=1
FCN3 e=2
850hpa wind speed intensity [m/s]
0
10
20
30
40
50
Power spectral density
500hpa geopotential [m²]
Ground truth
GenCast
FCN3 e=0
Wavenumber

# References I


[1] Boris Bonev, Thorsten Kurth, Christian Hundt, Jaideep Pathak, Maximilian Baust, Karthik Kashinath, and Anima Anandkumar.

Spherical fourier neural operators: Learning stable dynamics on the sphere.

In *International conference on machine learning*, pages 2806–2823. PMLR, 2023.

[2] Boris Bonev, Thorsten Kurth, Ankur Mahesh, Mauro Bisson, Jean Kossaifi, Karthik Kashinath, Anima Anandkumar, William D Collins, Michael S Pritchard, and Alexander Keller.

Fourcastnet 3: A geometric approach to probabilistic machine-learning weather forecasting at scale.

*arXiv preprint arXiv:2507.12144*, 2025.

[3] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al.

Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators.

*arXiv preprint arXiv:2202.11214*, 2022.