Composing Non-Conjugate Factor Graphs with Closed-Form Variational Inference

Mykola Lukashchuk — *Eindhoven University of Technology, the Netherlands*
Kyrylo Yemets — *Lviv Polytechnic National University, Lviv, Ukraine*
Wouter M. Kouw — *Eindhoven University of Technology, the Netherlands*
Dmitry Bagaev — *Eindhoven University of Technology, the Netherlands*
İsmail Şenöz — *Lazy Dynamics, Utrecht, the Netherlands*
Jeff Beck — *Great Sky, Boulder, Colorado, USA*
Bert de Vries — *Eindhoven University of Technology, the Netherlands*

Abstract

Stacking probabilistic building blocks into deeper architectures typically breaks closed-form inference. We show that closed-form inference can be preserved. We identify five factor-graph primitives: a bilinear factor, an exponential link, a Gamma prior, a Gaussian likelihood, and an equality node, and prove that any model composed from them admits closed-form variational message passing. The construction works because each primitive preserves a small set of message families: under mean-field factorization, messages on Gaussian variables remain Gaussian and messages on precision variables remain Gamma, while the only non-conjugate interface, the exponential link, remains tractable through the Gaussian moment-generating function and the sufficient statistics of the Gamma family. We demonstrate composition at increasing depth, from static ensembles through input-dependent gating to split-branch routing, and show that stacking routing layers encodes arbitrary decision trees, establishing universal function approximation with closed-form inference. Applied to ensemble time-series forecasting, the framework yields a Bayesian mixture of experts in which gating functions are inferred rather than learned, providing calibrated uncertainty over expert selection across five benchmark datasets.

Proceedings of the 2nd International Conference on Probabilistic Numerics (ProbNum) 2026, Lappeenranta, Finland. PMLR Volume 341.

1 Introduction

In deterministic computation, building hierarchical systems is straightforward. Functions call functions, expressions nest within expressions, and complexity grows without any fundamental barrier. Languages and compilers execute such computational graphs step by step.

Probabilistic models need the same kind of hierarchy to be expressive enough. Consider ensemble forecasting: a simple model assigns a fixed precision to each expert, but what if an expert's reliability varies with the input? Then precision itself becomes a learned function of the input features, and that function should carry its own uncertainty. What if different experts are relevant in different regions of the input space? Then a routing layer must select among experts, adding another level of probabilistic structure on top. Each layer adds modeling power, and each layer should propagate uncertainty. Yet, stacking probabilistic building blocks into deeper architectures typically leads to intractable operations. The field has responded by developing increasingly powerful black-box inference methods: Markov chain Monte Carlo (Neal, 2011), black-box variational inference (Ranganath et al., 2014), amortized variational autoencoders (Kingma and Welling, 2013), normalizing flows (Rezende and Mohamed, 2015), and diffusion models (Ho et al., 2020). These approaches handle non-conjugacy by learning or sampling the inference procedure but sacrifice closed-form tractability in the process.

We take a different path. Rather than improving black-box inference to handle with compositional models, we ask: *can probabilistic models compose hierarchically while retaining closed-form inference?* Structural constraints on the approximate posterior—mean-field factorization and exponential-family form constraints—are known to reduce variational inference to closed-form updates in favorable settings. Message passing algorithms that minimize the Bethe free energy (Yedidia et al., 2005; Senöz et al., 2021) provide the execution mechanism. We show how to choose and compose fundamental probabilistic building blocks such that closed-form inference is preserved at every level of nesting.

The paper is organized around an analogy with programming languages: an *alphabet* of building blocks (Section 2), a *grammar* for composing them into models of increasing depth (Section 3), and a *runtime* in which form constraints and the Bethe free energy yield closed-form message passing derived from the model, not prescribed by the modeler (Section 4).

We make the following contributions:

1. We identify a five-letter alphabet of building blocks. The set is sufficient for universal approximation and closed under Q-conjugate variational message passing, so any composed model admits closed-form updates.

2. We demonstrate composition at increasing depth:

a static ensemble, input-dependent gating, and split-branch routing with piecewise-linear boundaries. Stacking routing layers encodes arbitrary decision trees, establishing universal function approximation.

3. We apply the framework to ensemble forecasting, yielding a Bayesian mixture of experts in which gating functions are inferred rather than learned, providing principled uncertainty over expert selection.

2 The Alphabet

We seek a set of probabilistic factors from which arbitrarily expressive models can be composed while retaining closed-form inference. We identify five such factors (Fig. 1). Sub-labels (a)–(e) match between figure and equation; edge styles encode variable roles: solid for weights and observations $(\mathbf{w}, \boldsymbol{\phi}, y, \mu)$, dashed for precisions (τ, γ, β), and dash-dotted for the latent z.

This set is *sufficient* for universal approximation (Corollary 1) and closed under Q-conjugate variational message passing (Theorems 1 and 2); we do not claim minimum cardinality.

$$f_*(z \mid \mathbf{w}, \boldsymbol{\phi}, \tau) = \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1}), \tag{1a}$$

$$f_{\exp}(\gamma \mid z) = \delta(\gamma - \exp(z)), \tag{1b}$$

$$f_{\mathcal{G}}(\gamma \mid \alpha, \beta) = \mathcal{G}(\gamma \mid \alpha, \beta), \tag{1c}$$

$$f_{\mathcal{N}}(y \mid \mu, \tau) = \mathcal{N}(y \mid \mu, \tau^{-1}), \tag{1d}$$

$$f_=(x_1, x_2, x_3) = \delta(x_1 - x_2)\,\delta(x_1 - x_3). \tag{1e}$$

When y and μ are vectors, the normal factor (1d) denotes $\mathcal{N}(\mathbf{y} \mid \boldsymbol{\mu}, \tau^{-1}\mathbf{I})$, i.e. a multivariate Gaussian whose covariance is a scalar precision τ shared across all dimensions.

In this paper, we compose these factors into Forney-style factor graphs (FFG) (Forney, 2001; Loeliger, 2004). In an FFG, square nodes represent factors and edges represent variables; each expression above becomes one node. For instance, the square labeled $*$ in Fig. 1 a is the factor f_* from (1a), and its four edges are the variables $\mathbf{w}$, $\boldsymbol{\phi}$, τ, and z. Two factors that share a variable are connected by the same edge; composing models amounts to connecting nodes via shared edges.

3 The Grammar

This section shows how to write models in the factor-graph language. We start with one simple translation exercise: write a familiar probabilistic model as a graph (Depth 0). Then we extend it by composition: first by introducing a reusable word (Depth 1), followed by composing larger words for richer sentences (Depth 2). Runtime and update rules are deferred to Section 4. Readers who prefer an executable description may consult Section D, which provides the GraphPPL.jl (Nuijten et al., 2024) code corresponding to each factor graph in this section.

3.1 From letters to a model

Given n forecasters with predictions $\hat{y}_{i,j}$ for observation y_j, the Depth-0 smoothing problem is to infer the forecaster precisions from paired observations and predictions:

$$p(\boldsymbol{\gamma} \mid \mathbf{y}, \hat{\mathbf{y}}) \propto \prod_{i=1}^{n} f_{\mathcal{G}}(\gamma_i \mid \alpha_i, \beta_i) \prod_{\substack{i=1 \\ j=1}}^{n,m} f_{\mathcal{N}}(\hat{y}_{i,j} \mid y_j, \gamma_i). \tag{2}$$

The conditional notation in (2) names the inference task, not a directed sampling order. Since the preceding section introduced local factor functions rather than a complete joint distribution, for now (2) should be read simply as "infer $\boldsymbol{\gamma}$ given observed targets and forecaster predictions." The runtime later explains how the same product of local factors induces different message-passing problems depending on which variables are observed and which are left latent. The factor graph is a direct letter-by-letter transcription of this product: one gamma prior letter $f_{\mathcal{G}}$ per expert, one normal letter $f_{\mathcal{N}}$ per expert-observation pair, and equality letters that share γ_i across observations and y_j across experts. Figure 2 shows the fragment for expert i=1 and observation j=1; dotted edges indicate repetition over the remaining indices. The executable GraphPPL.jl code for this model is given in Section D.1.

3.2 Defining a new word

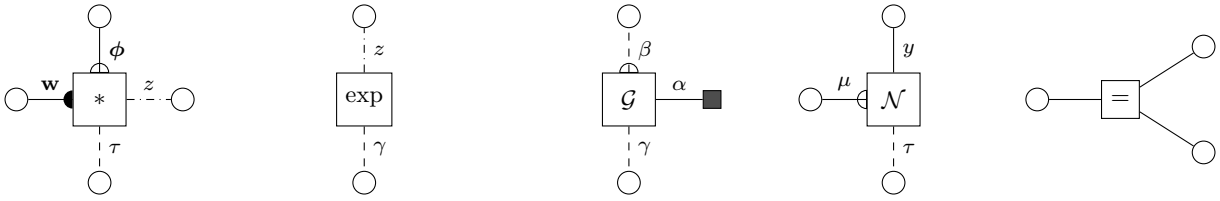
To make the precision variables γ_i input-dependent, we define one reusable word. This is composed of the softdot (1a) node with an exponential link (1b), which gives the *precision word* π (Fig. 3), mapping $(\mathbf{w}, \boldsymbol{\phi}, \tau)$ to γ.

Writing a Depth-1 sentence means replacing each Depth-0 precision letter $f_{\mathcal{G}}(\gamma_i \mid \alpha_i, \beta_i)$ with π, and adding shared parameter nodes $(\mathbf{w}_i, \tau_i)$. The right-hand likelihood part is unchanged; only the precision-generation part is replaced. Figure 3c shows this pattern. We call this sentence family *Precision-Gated Experts* (PGE).

3.3 Writing richer sentences

The Depth-1 precision word π maps $\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})$ through an exponential link, so γ grows monotonically with the linear score: wherever $\mathbf{w}^\top \boldsymbol{\phi}$ is large, γ is large too. A single π can therefore only cut the input space with one hyperplane, producing two regions. To express patterns like an exclusive-or (XOR) relation, which requires at least four regions, we need to compose a larger word from the same alphabet letters (Fig. 4).

The key idea is to create two branches that compete. A router softdot produces a score h from routing weights $\mathbf{v}_i$ and features $\boldsymbol{\phi}(\mathbf{x}_j)$. h is then taken by two switch softdots with opposing signs, followed by exponential links: when $h > 0$ the left switch produces a large activation κ^L and the right switch a small κ^R, and vice versa. Each branch has its own precision word π (with separate weights $\mathbf{w}_i$ and $\mathbf{u}_i$), whose output feeds into a normal factor gated by the switch activation κ^b. A large



(a) softdot (b) exp link (c) gamma (d) normal (e) equality

Figure 1: The building blocks as factor graph nodes. Square nodes are factors; round nodes are neighbors sharing the variable (priors, likelihoods, or other factors). Filled black squares denote *clamped* variables: fixed constants that cannot connect to other factors (e.g., the shape α of the gamma factor); this notation is used throughout. Orientation markers on the softdot (filled semi-circle on the **w** side, open on the ϕ side) identify edge roles when the node is rotated in composed graphs. Edge styles distinguish variable roles: solid for **w**, ϕ, y, μ; dashed for precisions τ, γ, β; dash-dotted for the latent z. The equality node enforces that all connected edges carry the same variable.

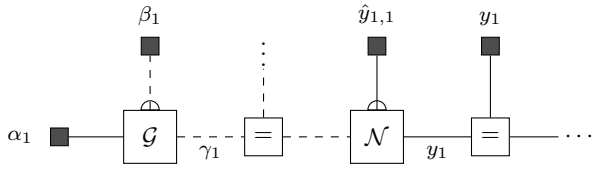


Figure 2: Depth 0 factor graph (static ensemble), shown starting from expert i=1 and observation j=1. The vertical dots ⋮ on the γ_1 equality node extend to the other observations $j = 2, \ldots, m$; the horizontal dots $\cdots$ on the y_1 equality node extends to the other $n - 1$ experts.

κ^b makes that normal factor tight, forcing the branch to pass its sub-expert score through; a small κ^b leaves the normal diffuse and the branch negligible. After a final exponential link and likelihood, both branches share the observation y_j through an equality node. The result is a piecewise-linear precision function: the router selects which sub-expert controls γ in each region of input space. Two experts with split-branch routing partition the input into four regions, which is sufficient to capture XOR. Stacking split-branch layers yields deeper sentences encoding arbitrary binary decision trees; the formal expressiveness argument is deferred to Section 5.

4 The Runtime

The inference algorithm for any model composed of the alphabet of Section 2 is not prescribed by the modeler but *emerges* from the model specification (Senöz et al., 2021; Yedidia et al., 2005). Executing inference requires two alternating operations: computing a message from a factor toward an edge, and computing the marginal on an edge from incoming messages. This section shows that both operations have closed-form solutions of known parametric form for any factor graph composed from the alphabet.

4.1 Declarative inference

The modeler specifies four things:

1. A **graph**: a Forney-style factor graph assembled from the five alphabet letters (1).
2. **Terminations**: observation or unity factors attached to half-edges (single-factor edges), determining which variables are observed or latent.
3. **Factorization constraints**: a mean-field decomposition of the approximate posterior, e.g.,
$$q(\mathbf{w}, \boldsymbol{\tau}, \mathbf{z}) = \prod_i q(\mathbf{w}_i)\, q(\tau_i) \prod_j q(z_{i,j})\,.$$
4. **Form constraints**: the exponential-family form for each marginal (Gaussian for solid and dash-dot edges, Gamma for dashed edges).

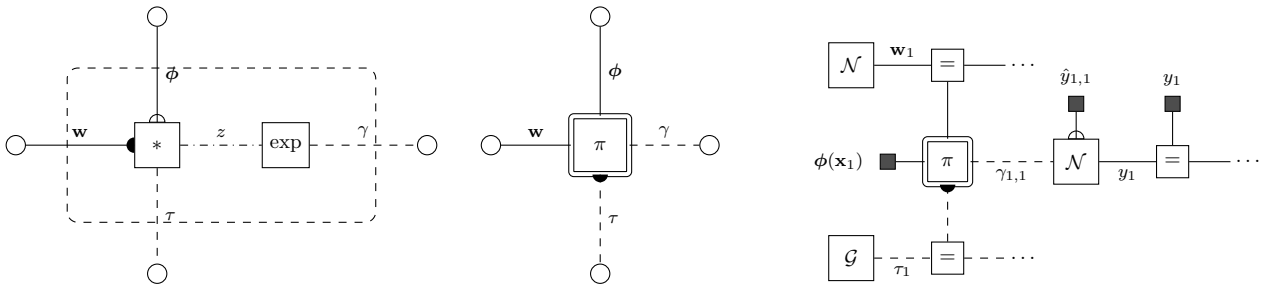
A terminated graph defines a conditional distribution after normalization; Section A.2 illustrates this on the Depth-0 model. Given these declarations, the Bethe free energy (BFE) provides the variational objective (Section A.3). Two message passing algorithms emerge from the BFE stationarity conditions (Section A.4): *variational message passing* (VMP) emerges when mean-field factorization constraints are imposed, and *belief propagation* (BP) emerges in the unconstrained case. Around deterministic factors such as the exponential link (1b), mean-field factorization cannot be imposed without degeneracy, so BP is the only option there. Both algorithms produce messages sent along edges; the difference is what information they require as input.

4.2 Computing messages

A VMP message from factor a toward edge j (Eq. (22)) requires only the *marginals* $q_i(s_i)$ on the remaining edges $i \in \mathcal{E}(a) \setminus j$. Suppose that each marginal has the parametric form indicated by its line style: Gaussian for solid and dash-dot edges, Gamma for dashed edges. Then the message computation depends only on the factor a and the *line types* of its other edges—not on the rest of the graph. The line styles introduced in Section 2 act as a type system: given marginals of the matching types, every VMP message is determined locally.

A BP message from factor a toward edge j (Eq. (19)) requires the incoming *messages* $\mu_{ia}(s_i)$ on the remaining edges, not marginals. The form of these messages depends on what is connected on the other side of those edges, so the line-type argument does not apply directly.

In our alphabet, two factors require BP: the equality node (1e) and the exponential link (1b). Both are deterministic (delta functions) and cannot be mean-field factorized without degeneracy. The equality node is



(a) Expanded (b) Compact (c) Depth 1 (PGE)

Figure 3: The *precision word* π and Depth-1 model. (a) Internal structure: a softdot and exponential link connected by latent z, computing input-dependent precision γ from $\mathbf{w}$, ϕ, and τ. (b) Compact notation; double border indicates a composite word, filled semi-circle marks the τ (input precision) side. (c) Depth 1 factor graph (Precision-Gated Experts) for expert i=1, observation j=1: compared to Depth 0 (Fig. 2), the gamma prior is replaced by π, producing input-dependent $\gamma_{1,1}$. Weights $\mathbf{w}_1$ and τ_1 are shared across observations ($\cdots$).

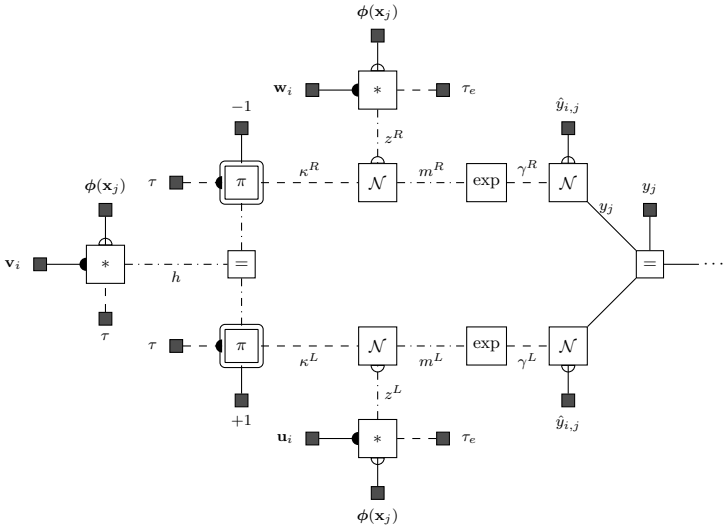


Figure 4: Depth 2 factor graph (split-branch routing) for one expert, one observation. The router softdot produces h; two switch π words with clamped weights -1 and $+1$ convert h into opposing activations κ^R, κ^L. Each branch has a sub-expert softdot producing z^b, gated by a normal factor whose precision is κ^b: the active branch (κ large) passes z^b tightly, while the suppressed branch (κ small) is diffuse. After exponential links and likelihoods, both branches share y_j via the equality node.

trivial—its BP message toward any edge is the product of incoming messages from the other edges, which preserves the message type. The exponential link is the non-trivial case: as a nonlinear deterministic factor, it cannot be mean-field factorized, so imposing $q(z)\,q(\gamma)$ with the constraint $\gamma = e^z$ would force both marginals to be degenerate.

However, every factor adjacent to the exp link is mean-field factorized, so the messages arriving at the exp link have known forms. On the z edge (Fig. 5b), the only possible neighbors are the softdot (1a), the normal factor (1d), and the equality node (1e)—and each sends a Gaussian message under the mean-field assumption. In every case, the incoming message on z is Gaussian. The BP rule through the delta function then gives

$$\mu_{\gamma \leftarrow \exp}(\gamma) = \int \delta(\gamma - e^z)\, \mu_{z \to \exp}(z)\, \mathrm{d}z, \qquad (3)$$

which is always a log-normal distribution on γ. The same enumeration on the γ edge (normal (1d), softdot (1a), gamma (1c), or equality (1e) as neighbors) shows that the incoming message is always Gamma-typed. The BP rule in the $\gamma \to z$ direction then produces a log-gamma distribution on z (Lukashchuk et al., 2024, Section 2.2).

The VMP line-type argument together with the BP enumeration above covers every factor–edge pair in the alphabet. We summarize this with the following definition and theorem.

Definition 1 (Proper factor graph)**.** *A factor graph is* proper *if it is assembled from the alphabet (1) and every shared edge respects the line types: solid and dash-dot edges connect only to Gaussian-typed ports, and dashed edges connect only to Gamma-typed ports.*

Theorem 1 (Closed-form messages of known type)**.** *Let $\mathcal{G}$ be a proper factor graph with full mean-field factorization on all non-deterministic factors, and suppose the marginals on all edges have the parametric form indicated by their line type. Then for every factor–edge pair in $\mathcal{G}$, the outgoing message exists in closed form and has a known parametric type:*

1. *At VMP factors (softdot (1a), normal (1d), gamma (1c)): the message type matches the line type of the target edge—Gaussian on solid and dash-dot edges, Gamma on dashed edges.*
2. *At the equality node (1e) (BP, trivial): the message is the product of incoming messages from the other edges and preserves their type.*

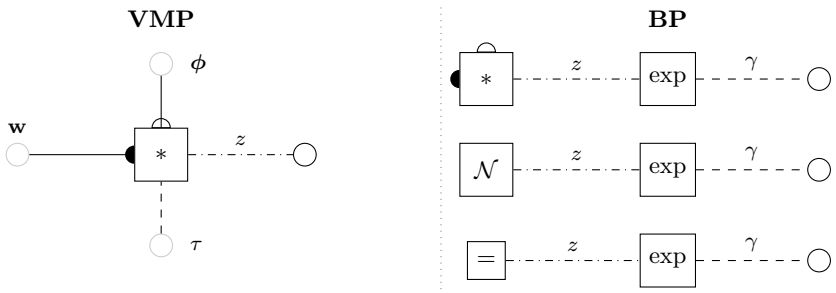


Figure 5: Two modes of message computation. **(a)** Under mean-field factorization constraints, the message from the softdot toward z depends only on the marginal *types* of its other edges: $q(\mathbf{w}), q(\phi) \in \mathcal{N}$ (solid lines), $q(\tau) \in \mathcal{G}$ (dashed line); which factors sit on the other side is irrelevant, the line styles act as a type system. **(b)** The exp link uses belief propagation (BP); the message toward γ depends on the actual message arriving on z: $\mu_{\gamma \leftarrow \exp}(\gamma) = \int \delta(\gamma - e^z)\, \mu_{z \to \exp}(z)\, \mathrm{d}z$. Enumerating all possible connections on the z edge (softdot, normal, equality), each sends a Gaussian message under mean-field, so the outgoing message on γ is always log-normal.

3. *At the exp link* (1b) *(BP, non-trivial): the message toward γ is log-normal; the message toward z is log-gamma (Lukashchuk et al., 2024, Section 2.2).*

Proof. For VMP factors, the message depends only on the factor and the marginal types on its other edges (Eq. (22)). The line types fix these marginal types, so the message is determined locally, regardless of the rest of the graph. The equality node uses BP, but its message is the product of incoming messages (Eq. (19)), which preserves the exponential family. For the exp link, mean-field factorization on all adjacent factors guaranties that the incoming message on z is always Gaussian and the incoming message on γ is always Gamma, as established by the enumeration above. The BP rule (3) then produces a log-normal message on γ and a log-gamma message on z. The full catalog of update rules is given in Section A.5. □

4.3 Computing marginals

For all edges *not adjacent to an exp-link factor*, the BFE stationarity condition (Eq. (20)) gives the marginal as the normalized product of the two incoming messages:

$$q_j^*(s_j) \propto \mu_{jb}(s_j)\, \mu_{jc}(s_j). \tag{4}$$

By Theorem 1, both messages at such edges belong to the same exponential family (Gaussian or Gamma). Their product stays in-family, and the marginal is obtained by combining natural parameters. No approximation is involved.

At the two edges of each exp-link factor, the situation is different. Theorem 1 states that on the z edge, one message is Gaussian (from the adjacent VMP factor) and the other is log-gamma (from the exp link); on the γ edge, one is Gamma and the other is log-normal. In neither case does the product (4) belong to a standard parametric family, so direct multiplication does not yield a tractable marginal.

To handle these edges, we impose a form constraint: $q(z) \in \mathcal{Q}_z = \{\mathcal{N}(z \mid m, v)\}$ with natural parameters $\boldsymbol{\eta} = (\eta_1, \eta_2)^\top = (m/v,\, -1/(2v))^\top$. Restricting the BFE to this parametric family yields a constrained objective $F[\boldsymbol{\eta}]$ that upper-bounds the unconstrained BFE, so the resulting inference is a variational bound.

We derive the stationarity condition for $q(z)$ in the concrete case where z connects the softdot (1a) on the left to the exp link (1b) on the right, with a normal factor (1d) on the far side of γ (so that e^z serves as the precision of the normal).

The exp link is a deterministic factor $f_{\exp}(z, \gamma) = \delta(\gamma - e^z)$. By (Senöz et al., 2021, Theorem 8), the node-local belief must respect the deterministic constraint, giving $q_{\exp}(z, \gamma) = q(z)\, \delta(\gamma - e^z)$ and reducing the node-local free energy to $-\mathbb{H}[q(z)]$ (for a general overview of the treatment of deterministic nodes in FFGs, see (Senöz et al., 2021, Section 5.2)). The delta absorbs γ and couples the far-side normal factor to z: the normal factor contributes $\log f_{\mathcal{N}}(y \mid \mu, e^{-z})$ evaluated at $\gamma = e^z$. Under full mean-field on the normal factor, the BFE terms that depend on $q(z)$ reduce to

$$F_z[\boldsymbol{\eta}] = -\mathbb{H}[q(z)] + \underbrace{\mathbb{E}_{q_z q_w q_\tau}[-\log f_*]}_{\text{softdot}} + \underbrace{\mathbb{E}_{q_z q_y q_\mu}\left[-\log f_{\mathcal{N}}(y \mid \mu, e^{-z}) - z\right]}_{\exp\ +\ \text{normal}}, \tag{5}$$

where the Jacobian term $-z$ arises from the change of variables $\gamma = e^z$. The softdot factor is conjugate in z: its contribution involves only Gaussian sufficient statistics of $q(z)$. The second term, originating from the exp link and the far-side normal factor, expands to

$$\mathbb{E}_{q(z)}\left[-\bar{\ell}_{\exp}(z)\right] = \tfrac{1}{2}\mathbb{E}_{q(z)}[z] - \tfrac{1}{2}\mathbb{E}_{q(y),q(\mu)}\left[(y-\mu)^2\right]\, \mathbb{E}_{q(z)}[e^z] + \text{const}, \tag{6}$$

where the coefficient of $\mathbb{E}_{q(z)}[e^z]$ depends on the marginals $q(y)$ and $q(\mu)$ of the far-side normal factor. The two contributions in (5) can be recognized as the two messages from Theorem 1: the softdot contribution is the Gaussian message $\mu_{z \leftarrow \text{softdot}}$, and the exp-link contribution (6) is the log-gamma message $\mu_{z \leftarrow \exp}$ (with its coefficients determined by the far-side normal factor). In a terminated FFG, edge z connects exactly two factors, so Theorem 1 provides exactly two messages. Their log-sum defines the stationarity target:

$$\bar{\ell}(z) = \log \mu_{z \leftarrow \text{softdot}}(z) + \log \mu_{z \leftarrow \exp}(z), \tag{7}$$

where $\mu_{z\leftarrow\text{softdot}}$ is Gaussian and $\mu_{z\leftarrow\exp}$ is log-gamma (Theorem 1). For exponential-family $q(z)$, the gradient of the entropy satisfies $\nabla_{\boldsymbol{\eta}}\mathbb{H}[q(z)] = -\mathbf{F}(\boldsymbol{\eta})\,\boldsymbol{\eta}$, where $\mathbf{F}(\boldsymbol{\eta})$ is the Fisher information matrix (Khan and Rue, 2023). Setting $\nabla_{\boldsymbol{\eta}}F_z = 0$ and rearranging gives the stationarity condition

$$\boldsymbol{\eta}^* = \mathbf{F}(\boldsymbol{\eta}^*)^{-1}\,\nabla_{\boldsymbol{\eta}^*}\mathbb{E}_{q^*(z)}\left[-\bar{\ell}(z)\right]. \tag{8}$$

The non-conjugate term in (6) requires $\mathbb{E}_{q(z)}[e^z]$ and its derivatives with respect to $\boldsymbol{\eta}$. Under the Gaussian form constraint, the moment-generating function provides this in closed form:

$$\mathbb{E}_{q(z)}[e^z] = \exp\!\left(m + \tfrac{v}{2}\right), \tag{9}$$

and its derivatives with respect to $\boldsymbol{\eta}$ are analytic expressions of (η_1, η_2). This is precisely the Q-conjugacy condition (Lukashchuk et al., 2024): the expectations under $q(z)$ of all log-factors adjacent to z are closed-form functions of $\boldsymbol{\eta}$. Combined with the closed-form Fisher information for the Gaussian in natural parameters,

$$\mathbf{F}(\boldsymbol{\eta}) = \begin{pmatrix} -\frac{1}{2\eta_2} & \frac{\eta_1}{2\eta_2^2} \\ \frac{\eta_1}{2\eta_2^2} & \frac{1}{2\eta_2^2} - \frac{\eta_1^2}{2\eta_2^3} \end{pmatrix}, \tag{10}$$

the stationarity condition (8) yields closed-form fixed-point equations for $\boldsymbol{\eta}^*$.[1]

The derivation for $q(\gamma)$ on the γ edge is analogous: the Gamma form constraint provides closed-form sufficient statistics $\mathbb{E}_{q(\gamma)}[\log\gamma] = \psi(\alpha) - \log\beta$ and $\mathbb{E}_{q(\gamma)}[\gamma] = \alpha/\beta$, where ψ denotes the digamma function and α, β are the shape and rate of the Gamma marginal $q(\gamma)$, ensuring Q-conjugacy on that edge as well. The full derivation is deferred to Section A.6.

The specific conjugate partner on the far side of the exp link—whether softdot, normal, gamma, or equality—does not affect the argument: Theorem 1 guarantees it always produces a message of the matching line type, so the coefficients in (6) change but the closed-form structure is preserved. We summarize the result.

Theorem 2 (Closed-form marginals)**.** *Let $\mathcal{G}$ be a proper factor graph (Definition 1) with full mean-field factorization on all non-deterministic factors and form constraints matching the line types. Then for every edge in $\mathcal{G}$, the BFE-stationary marginal exists in closed form: at edges not adjacent to an exp link, as the normalized product of two same-family messages; at edges adjacent to an exp link, as the solution of a closed-form fixed-point equation.*

Together, Theorems 1 and 2 establish that both operations of the runtime—computing messages and computing marginals—are closed-form for any proper factor graph. Full mean-field is the worst case; structured factorization at conjugate boundaries can only improve the approximation.

[1] "Closed-form" means every term in (8) is an analytic function of $\boldsymbol{\eta}$, with no intractable integrals; the equation is still nonlinear in $\boldsymbol{\eta}^*$ (through (9)), so it admits a closed-form *update scheme* rather than a closed-form *solution*. In practice, we solve it by natural gradient descent on the exponential family manifold (Lukashchuk et al., 2025) with Manopt.jl (Bergmann, 2022); see Section E.

5 Expressiveness

By Theorems 1 and 2, any proper factor graph (Definition 1) has closed-form inference. The question is therefore: how expressive are proper factor graphs?

Proposition 1 (Decision tree encoding)**.** *For any binary decision tree T of depth d with axis-aligned splits and linear leaf functions, there exists a proper factor graph that implements T in the limit $\tau \to \infty$.*

Proof. By induction on d. The base case (d=0) is a single softdot factor. For $d > 0$, the root split is implemented by a depth-2 router with opposing switches (Section 3.3); each child subtree of depth $d-1$ is implemented by the inductive hypothesis. □

Consider the output variable y_j in the depth-2 construction (Fig. 4): it is shared across all branches via equality nodes, and its posterior mean $\mathbb{E}_q[y_j](\mathbf{x})$ is determined by the precision-weighted combination of the expert predictions $\hat{y}_i$. In the sharp-routing limit ($\tau \to \infty$), each routing decision becomes deterministic and the dominant expert's prediction is selected in each region. Stacking d layers of splits produces 2^d regions, and each region selects one expert's constant prediction $\hat{y}_i$. The posterior mean is then a so-called simple function (a finite sum of constants times indicator functions of measurable sets), which can approximate any measurable function to arbitrary accuracy (Rudin, 2013, Ch. 1, Theorem 1.17). Universal approximation follows.

Corollary 1 (Universal approximation)**.** *For any continuous $f\colon \mathcal{X} \to \mathbb{R}$ on compact $\mathcal{X} \subset \mathbb{R}^p$ and any $\epsilon > 0$, there exists a proper factor graph built by stacking split-branch routing layers such that $\|\mathbb{E}_q[y_j](\mathbf{x}) - f(\mathbf{x})\|_\infty < \epsilon$ in the sharp-routing limit, with all inference updates being closed-form.*

For finite τ, the posterior over y_j carries calibrated uncertainty about the routing decisions (Fig. 6). Small τ yields wide posteriors over the router score h, so both branches contribute, and the model hedges across experts; large τ concentrates the posterior on a single branch. This uncertainty is a direct output of inference, not an add-on; it can be propagated downstream to any quantity that depends on y_j. The connection to classical neural universal approximation theorems is discussed in Section C.

6 Application

Time series forecasting is a natural testbed for the framework. Multiple forecasters with different inductive biases (linear, convolutional, recurrent) are trained independently on the same data, and their relative reliability varies with the temporal context: one forecaster may excel during trend regimes while another is more accurate around seasonal peaks. The task of predicting the next time window is a standard regression problem, but the interesting question is *which expert to trust given the current context*. Our framework answers this question

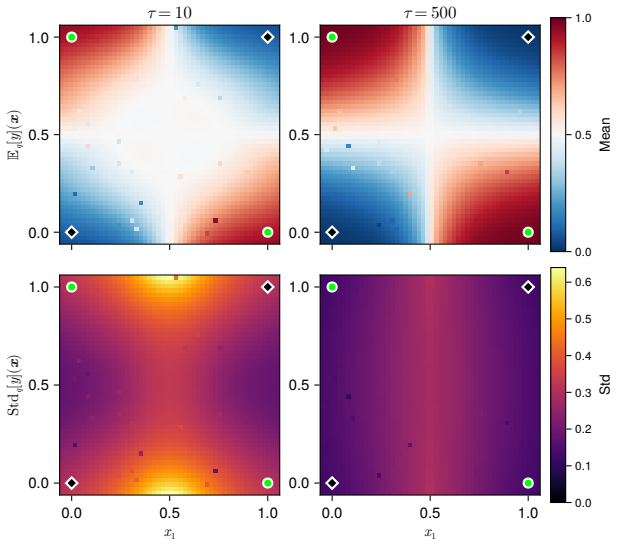


Figure 6: Posterior prediction for the XOR encoding with two depth-2 experts at different routing precisions τ. **Top:** the posterior mean $\mathbb{E}_q[y_j](\mathbf{x})$ transitions from a smooth blend (τ=10) to sharp classification regions (τ=500). **Bottom:** the posterior standard deviation shows routing uncertainty: large everywhere under soft routing, concentrated at the decision boundary under sharp routing. Explicit parameter values are given in Section B.

probabilistically: the precision-gated mechanism learns input-dependent trust from the recent history.

6.1 Models and inference variants

We evaluate three model structures at increasing depth. The Static ensemble (Depth 0, Fig. 2) learns a single precision γ_i per forecaster, recovering Cochran's inverse-variance weighting (Cochran, 1937). The Dynamic model (Depth 1, Fig. 3c) replaces each gamma prior with a precision word π, making trust input-dependent. The Noisy model (Depth 1) augments the Dynamic model with an additional normal factor with precision κ between the expert-combined variable and the observation, separating forecaster disagreement from irreducible predictive noise; the factor graph and GraphPPL.jl code are given in Section D.3.

For the Dynamic and Noisy models, inference maintains a posterior $q(\mathbf{w}_i)$ over the weight vectors. We consider two inference variants that share the model but differ in the covariance constraint: full covariance $q(\mathbf{w}_i) = \mathcal{N}(\mathbf{w}_i \mid \mathbf{m}_i, \mathbf{V}_i)$, or diagonal covariance $q(\mathbf{w}_i) = \mathcal{N}(\mathbf{w}_i \mid \mathbf{m}_i, \mathrm{diag}(\mathbf{v}_i))$, reducing parameters from $O(d^2)$ to $O(d)$ per expert. This gives five configurations: Static, Dynamic, Dynamic Diagonal, Noisy, and Noisy Diagonal. All use the closed-form message passing of Section 4.

6.2 Setup

We combine $n = 7$ forecasters (five neural: CNN, DLinear, NLinear (Zeng et al., 2023), LSTM (Hochreiter and Schmidhuber, 1997), NConv; two constant: 10th and 90th quantiles) on five benchmarks (ETTh1, ETTh2 (Zhou et al., 2021), Exchange Rate, Electricity (Trindade, 2015), Traffic) with horizons $H \in \{96, 192, 336, 720\}$. Feature vectors $\boldsymbol{\phi}(\mathbf{x}_j)$ come from a VAE with a latent dimension of 64. For comparison, we include softmax-based Mixture-of-Experts (MoE) (Jacobs et al., 1991) baselines with single-layer and two-layer gating networks. Full experimental details, dataset statistics, and parameter counts are in the appendix (Section E and Tables 2 and 3). We report MSE and negative log-likelihood (NLL; see Section E.3 for MoE baseline NLL computation). Our framework provides calibrated predictive distributions, not just point forecasts.

6.3 Results

Full results—including a Dynamic comparison with a state-space Matérn-3/2 GP (Hartikainen and Sarkka, 2010)—are in Tables 5 to 7; summary plots in Figs. 7, 9 and 10.

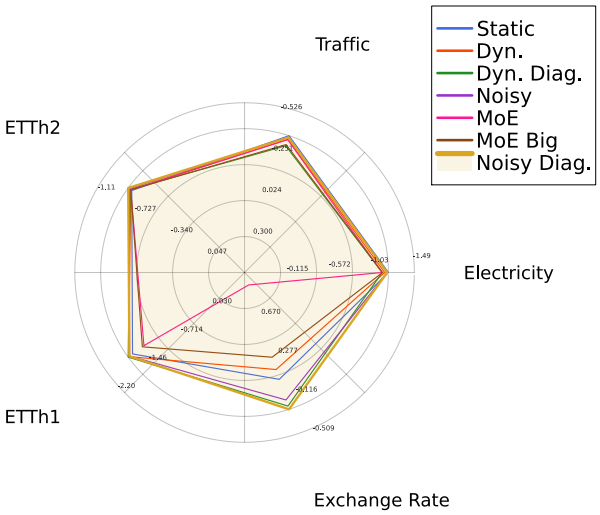
The Static ensemble already matches or surpasses the best individual forecaster on multiple dataset–horizon pairs. The Dynamic variants adapt expert trust to the input, achieving the best MSE across all ETTh1 horizons with substantially lower NLL. The Diagonal variants are particularly strong on multivariate datasets: on Exchange Rate, Noisy Diagonal achieves the best NLL while reducing the parameter count from 15 512 to 952 (d=65). The softmax-based MoE baselines are occasionally competitive in MSE but often have non-finite NLL, indicating poor calibration. Overall, Noisy Diagonal provides the best accuracy–calibration–efficiency trade-off (Fig. 9). Code: repository.

7 Related Work

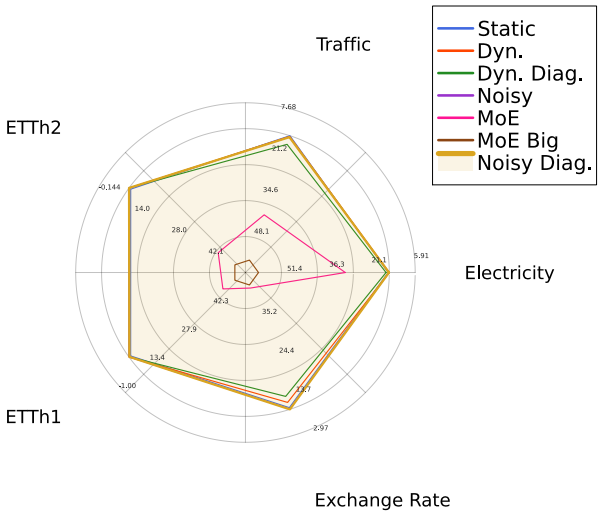
The depth-0 instance of our framework recovers the inverse-variance weighting of Cochran (1937); our framework generalizes it by learning precisions from data, making them input-dependent and composing them hierarchically.

Our universality result (Corollary 1) parallels classical universal approximation theorems for neural networks (Cybenko, 1989; Hornik, 1991): the core mechanism—a sufficiently fine partition of the input space—is the same; however, our framework replaces gradient-based point estimation with Bethe free energy optimization, yielding marginal posterior distributions.

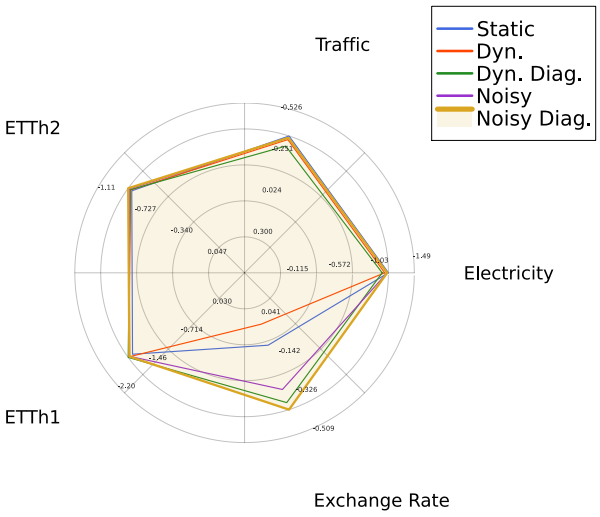
The current alphabet covers $\times$, exp, and $+$ (the last is not needed for universality but is easy to include); these suffice so that a model shares the topology of the corresponding computational graph while carrying uncertainty on every edge. Unlike a deterministic compiler, which executes instructions without an objective, the probabilistic runtime *optimizes* the Bethe free energy at every iteration. This separates the framework from linearization-based approaches such as Laplace propagation (Smola et al., 2003), which approximate the nonlinearity and optimize no bound; here, the nonlinearity is exact, and only the posterior family is constrained. We do not have a formal lower bound on alphabet size, but we conjecture that at least one non-conjugate factor is required: a purely linear–Gaussian alphabet remains



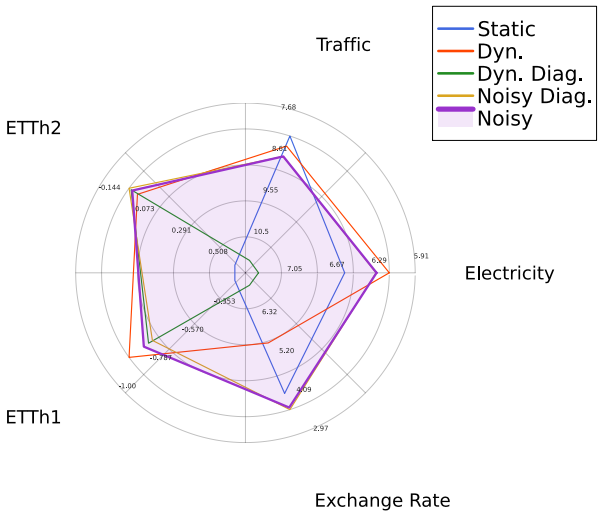
(a) All methods: Log MSE



(b) All methods: Log NLL



(c) Our framework: Log MSE



(d) Our framework: Log NLL

Figure 7: Radar charts of log-transformed MSE and NLL averaged over all horizons; each axis is a dataset, larger area is better. (a, b) All methods: Noisy Diagonal attains the largest area; larger neural MoE models improve MSE but become overconfident, with poor NLL. (c, d) Our framework only: Noisy Diagonal has the best MSE coverage; both noisy variants provide comparable NLL.

linear–Gaussian end-to-end and cannot reach universal approximation; the exponential link plays that role.

The per-edge stationarity condition (8) has the same form as the global optimality condition of Khan (2025, Eq. 5); our contribution is that it arises *locally* at each non-conjugate edge of the factor graph, so that marginal computation decomposes into independent per-edge problems solved by message passing.

Concurrent work on the generalized HGF (Weber et al., 2026) shares this philosophy but relies on Laplace approximation updates and does not provide expressiveness guaranties; our construction provides both.

8 Conclusion

The practical success of deep learning rests not only on its expressive power (LeCun et al., 2015) but also on its *accessibility*: layers, backpropagation, and an optimizer suffice to compose powerful models from modular building blocks. The factor graph approach follows a similar compositional philosophy (Loeliger, 2004), yet in practice demands considerably more: deriving message rules, selecting message types, and mixing algorithmic techniques for each new problem.

Our contribution is to narrow this gap. A finite alphabet of Q-conjugate factors composes freely, bringing layer-like modularity to probabilistic inference: softdot experts, exponential-link gates, and split-branch routers stack to arbitrary depth with guaranteed closed-form variational message passing. The alphabet is the language, the Bethe free energy is the compiler, and message passing is the runtime (van de Laar et al., 2018; Bagaev and De Vries, 2023); no new derivations are needed. The tradeoff is explicit: the practitioner trades arbitrary differentiable operations for closed-form marginal inference. This limitation is mild—the alphabet is a universal approximator—and should diminish as the library of reusable composed subgraphs grows.

Acknowledgements

We gratefully acknowledge financial support by the Dutch Ministry of Economic Affairs (PPS funding), by the

Dutch Research Council (NWO) and by hearing aid manufacturer GN Hearing, under contracts TKI-HTSM/21.0161/2112P09 (project: Auto-AR) and KICH3.LTP.20.006 (Project: ROBUST).

Bibliography

D. Bagaev and B. De Vries. Reactive Message Passing for Scalable Bayesian Inference. *Scientific Programming*, 2023:1–26, May 2023. ISSN 1875-919X, 1058-9244. doi: 10.1155/2023/6601690. URL https://www.hindawi.com/journals/sp/2023/6601690/.

R. Bergmann. Manopt.jl: Optimization on Manifolds in Julia. *Journal of Open Source Software*, 7(70):3866, 2022. doi: 10.21105/joss.03866.

W. G. Cochran. Problems arising in the analysis of a series of similar experiments. *Supplement to the Journal of the Royal Statistical Society*, 4(1):102–118, 1937.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, Dec. 1989. ISSN 1435-568X. doi: 10.1007/BF02551274. URL https://doi.org/10.1007/BF02551274.

J. Dauwels. On Variational Message Passing on Factor Graphs. In *IEEE International Symposium on Information Theory*, pages 2546–2550, Nice, France, June 2007. doi: 10.1109/ISIT.2007.4557602. URL http://ieeexplore.ieee.org/abstract/document/4557602.

G. Forney. Codes on graphs: normal realizations. *IEEE Transactions on Information Theory*, 47(2):520–548, Feb. 2001. ISSN 0018-9448. doi: 10.1109/18.910573. URL https://ieeexplore.ieee.org/abstract/document/910573.

J. Hartikainen and S. Sarkka. Kalman filtering and smoothing solutions to temporal Gaussian process regression models. In *2010 IEEE International Workshop on Machine Learning for Signal Processing*, pages 379–384, Kittila, Finland, Aug. 2010. IEEE. ISBN 978-1-4244-7875-0. doi: 10.1109/MLSP.2010.5589113. URL http://ieeexplore.ieee.org/document/5589113/.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(91)90009-T. URL https://www.sciencedirect.com/science/article/pii/089360809190009T.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

M. E. Khan. Information Geometry of Variational Bayes. *Information Geometry*, 8(S1):275–289, Nov. 2025. ISSN 2511-2481, 2511-249X. doi: 10.1007/s41884-025-00174-3. URL https://link.springer.com/10.1007/s41884-025-00174-3.

M. E. Khan and H. Rue. The Bayesian learning rule. *Journal of Machine Learning Research*, 24(281):1–46, 2023.

D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, Dec. 2013. URL http://arxiv.org/abs/1312.6114. arXiv: 1312.6114.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001. doi: 10.1109/18.910572. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=910572.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature14539. URL https://www.nature.com/articles/nature14539.

H.-A. Loeliger. An introduction to factor graphs. *Signal Processing Magazine, IEEE*, 21(1):28–41, Jan. 2004. doi: 10.1109/MSP.2004.1267047. URL https://ieeexplore.ieee.org/document/1267047.

H.-A. Loeliger. Factor Graphs and Message Passing Algorithms – Part 1: Introduction, 2007. URL http://www.crm.sns.it/media/course/1524/Loeliger_A.pdf.

I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

M. Lukashchuk, I. Senöz, and B. de Vries. Q-conjugate message passing for efficient bayesian inference. In *International conference on probabilistic graphical models*, pages 295–311. PMLR, 2024.

M. Lukashchuk, D. Bagaev, A. Podusenko, İ. Şenöz, and B. de Vries. ExponentialFamilyManifolds.jl: Representing exponential families as Riemannian manifolds. *Proceedings of the JuliaCon Conferences*, 7(70):179, 2025. doi: 10.21105/jcon.00179. URL https://doi.org/10.21105/jcon.00179.

R. M. Neal. *MCMC using Hamiltonian dynamics*. May 2011. doi: 10.1201/b10905. URL http://arxiv.org/abs/1206.1901. arXiv:1206.1901 [physics, stat].

W. W. L. Nuijten, D. Bagaev, and B. de Vries. GraphPPL.jl: A Probabilistic Programming Language for Graphical Models. *Entropy*, 26(11), 2024. ISSN 1099-4300. doi: 10.3390/e26110890. URL https://www.mdpi.com/1099-4300/26/11/890.

R. Ranganath, S. Gerrish, and D. Blei. Black Box Variational Inference. In S. Kaski and J. Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 814–822, Reykjavik, Iceland, Apr. 2014. PMLR. URL https://proceedings.mlr.press/v33/ranganath14.html.

D. J. Rezende and S. Mohamed. Variational Inference with Normalizing Flows. *arXiv:1505.05770 [cs, stat]*, May 2015. URL http://arxiv.org/abs/1505.05770. arXiv: 1505.05770.

W. Rudin. *Real and complex analysis*. McGraw-Hill international editions Mathematics series. McGraw-Hill, New York, NY, 3. ed., internat. ed., [nachdr.] edition, 2013. ISBN 978-0-07-100276-9 978-0-07-054234-1. OCLC: 957461070.

I. Senöz, T. van de Laar, D. Bagaev, and B. de Vries. Variational Message Passing and Local Constraint Manipulation in Factor Graphs. *Entropy*, 23(7): 807, July 2021. ISSN 1099-4300. doi: 10.3390/e23070807. URL https://www.mdpi.com/1099-4300/23/7/807.

A. Smola, S. Vishwanathan, and E. Eskin. Laplace propagation. In *Advances in neural information processing systems*, volume 16. MIT Press, 2003. URL https://proceedings.neurips.cc/paper_files/paper/2003/file/7fd804295ef7f6a2822bf4c61f9dc4a8-Paper.pdf.

A. Trindade. ElectricityLoadDiagrams20112014. *UCI Machine Learning Repository*, 10:C58C86, 2015.

T. van de Laar, M. Cox, I. Senoz, I. Bocharov, and B. de Vries. ForneyLab: A Toolbox for Biologically Plausible Free Energy Minimization in Dynamic Neural Models. In *Conference on Complex Systems (CCS)*, Thessaloniki, Greece, Sept. 2018.

L. A. Weber, P. T. Waade, N. Legrand, A. H. Møller, K. E. Stephan, and C. Mathys. The generalized Hierarchical Gaussian Filter. Mar. 2026. doi: 10.7554/elife.110174.1. URL http://dx.doi.org/10.7554/eLife.110174.1.

J. Winn and C. M. Bishop. Variational Message Passing. *Journal of Machine Learning Research*, 6(23): 661–694, 2005. ISSN 1533-7928. URL http://jmlr.org/papers/v6/winn05a.html.

J. S. Yedidia, W. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, July 2005. ISSN 0018-9448. doi: 10.1109/TIT.2005.850085. URL http://ieeexplore.ieee.org/abstract/document/1459044.

A. Zeng, M. Chen, L. Zhang, and Q. Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11121–11128, 2023.

H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pages 11106–11115. AAAI Press, 2021.

A Forney-style Factor Graphs and Message Passing

This appendix reviews Forney-style factor graphs and shows how message passing algorithms arise from minimizing the Bethe free energy. We follow the presentation of Senöz et al. (2021), to which we refer for proofs and further details.

A.1 Forney-style factor graphs

A Forney-style factor graph (FFG) is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ (Forney, 2001; Loeliger, 2004). We denote the neighboring edges of a node $a \in \mathcal{V}$ by $\mathcal{E}(a)$. Vice versa, for an edge $i \in \mathcal{E}$, the notation $\mathcal{V}(i)$ collects all neighboring nodes. As a notational convention, we index nodes by a, b, c and edges by i, j, k, unless stated otherwise.

An FFG represents a factorized positive function

$$f(\mathbf{s}) = \prod_{a \in \mathcal{V}} f_a(\mathbf{s}_a), \tag{11}$$

where $\mathbf{s}$ is the collection of all variables, $\mathbf{s}_a$ collects the argument variables of factor f_a, and a node $a \in \mathcal{V}$ corresponds to a factor f_a. The neighboring edges $\mathcal{E}(a)$ correspond to the variables $\mathbf{s}_a$ that are the arguments of f_a.

In the FFG representation, a node can be connected to an arbitrary number of edges, while an edge can only be connected to at most two nodes. An edge that connects to only one node (e.g., for an observed variable or a fixed hyperparameter) is called a *half-edge*. When a variable must be shared among more than two factors, FFGs introduce explicit *equality nodes* with factor function

$$f_a(s_i, s_j, s_k) = \delta(s_j - s_i)\, \delta(s_j - s_k), \tag{12}$$

which constrain connected edges to carry identical values.

A.2 Terminated factor graphs

If every edge in the FFG has exactly two connected nodes (including equality nodes), we call the graph a *terminated* FFG (TFFG). A half-edge can be closed by an observation factor, e.g. $f_{\text{obs}}(s_i) = \delta(s_i - s_i^{\text{obs}})$, or by a unity factor $f_1(s_i) = 1$. The latter leaves the variable uninformative; ordinary priors, such as the Gamma factor on γ_i, are just regular factors in the graph. All FFGs in this paper are assumed to be terminated before inference is run.

For the Depth-0 model, the same unnormalized factor product

$$f(\mathbf{y}, \hat{\mathbf{y}}, \boldsymbol{\gamma}) = \prod_{i=1}^{n} f_{\mathcal{G}}(\gamma_i \mid \alpha_i, \beta_i) \prod_{\substack{i=1 \\ j=1}}^{n,m} f_{\mathcal{N}}(\hat{y}_{i,j} \mid y_j, \gamma_i) \tag{13}$$

supports different directed graphical-model readings after termination. In smoothing, y_j and $\hat{y}_{i,j}$ are observed, so the terminated graph defines

$$p(\boldsymbol{\gamma} \mid \mathbf{y}, \hat{\mathbf{y}}) \propto f(\mathbf{y}, \hat{\mathbf{y}}, \boldsymbol{\gamma}).$$

In prediction, only the forecaster predictions are observed; y_* is left latent (closed by a unity factor), giving

$$p(y_*, \boldsymbol{\gamma} \mid \hat{\mathbf{y}}_*) \propto \prod_i f_{\mathcal{G}}(\gamma_i \mid \alpha_i, \beta_i) \prod_i f_{\mathcal{N}}(\hat{y}_{i,*} \mid y_*, \gamma_i).$$

Thus the FFG fixes the local factorization, while termination specifies which conditional distribution is represented.

A.3 Bethe free energy

Given a TFFG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ for a factorized function $f(\mathbf{s}) = \prod_{a \in \mathcal{V}} f_a(\mathbf{s}_a)$, the Bethe free energy (BFE) is defined as (Yedidia et al., 2005)

$$\begin{aligned} F[q, f] \triangleq \sum_{a \in \mathcal{V}} \underbrace{\int q_a(\mathbf{s}_a) \log \frac{q_a(\mathbf{s}_a)}{f_a(\mathbf{s}_a)} \, \mathrm{d}\mathbf{s}_a}_{F[q_a, f_a]} \\ + \sum_{i \in \mathcal{E}} \underbrace{\int q_i(s_i) \log \frac{1}{q_i(s_i)} \, \mathrm{d}s_i}_{H[q_i]}, \end{aligned} \tag{14}$$

such that the factorized beliefs

$$q(\mathbf{s}) = \prod_{a \in \mathcal{V}} q_a(\mathbf{s}_a) \prod_{i \in \mathcal{E}} q_i(s_i)^{-1} \tag{15}$$

satisfy the following constraints:

$$\int q_a(\mathbf{s}_a) \, \mathrm{d}\mathbf{s}_a = 1, \quad \text{for all } a \in \mathcal{V}, \tag{16a}$$

$$\int q_a(\mathbf{s}_a) \, \mathrm{d}\mathbf{s}_{a \setminus i} = q_i(s_i), \quad \text{for all } a \in \mathcal{V} \text{ and all } i \in \mathcal{E}(a). \tag{16b}$$

The normalization constraint (16a) and marginalization constraint (16b) together imply that the edge marginals $q_i(s_i)$ are also normalized. The BFE includes a local free energy term $F[q_a, f_a]$ for each node $a \in \mathcal{V}$ and an entropy term $H[q_i]$ for each edge $i \in \mathcal{E}$. In a TFFG every edge connects exactly two nodes; the edge entropy corrects for the double-counting that would arise from the node-local free energies alone.

The Bethe factorization (15) and constraints (16) are summarized by the *local polytope*

$$\begin{aligned} \mathcal{L}(\mathcal{G}) = \big\{ & q_a \text{ for all } a \in \mathcal{V} \text{ s.t. (16a)}, \\ & \text{and } q_i \text{ for all } i \in \mathcal{E}(a) \text{ s.t. (16b)} \big\}, \end{aligned} \tag{17}$$

which defines the constrained search space for the factorized variational distribution. The problem is then to find the beliefs in the local polytope that minimize the BFE:

$$q^*(\mathbf{s}) = \arg\min_{q \in \mathcal{L}(\mathcal{G})} F[q, f]. \tag{18}$$

A.4 Message passing from stationarity conditions

Instead of solving the global optimization problem (18) directly, we employ the factorization of the BFE and the local polytope to subdivide it into interdependent local objectives. The resulting fixed-point equations can be interpreted as messages on the edges of the FFG. We state the two results needed in this paper; proofs can be found in (Kschischang et al., 2001; Yedidia et al., 2005; Dauwels, 2007; Senöz et al., 2021).

Sum-product message passing (belief propagation). The stationary points of the Lagrangian associated with (18) yield the sum-product message update. The message from node b to edge j is

$$\mu_{jb}^{(k+1)}(s_j) = \int f_b(\mathbf{s}_b) \prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} \mu_{ib}^{(k)}(s_i) \, \mathrm{d}\mathbf{s}_{b \setminus j}, \tag{19}$$

where k is an iteration index and μ_{ib} denotes the incoming message on edge i towards node b. The edge belief is proportional to the product of the two incoming messages:

$$q_j^*(s_j) = \frac{\mu_{jb}^*(s_j) \, \mu_{jc}^*(s_j)}{\int \mu_{jb}^*(s_j) \, \mu_{jc}^*(s_j) \, \mathrm{d}s_j}. \tag{20}$$

On tree-structured graphs the sum-product algorithm recovers the exact posterior; on graphs with cycles, convergence is not guaranteed in general (Yedidia et al., 2005).

Naive mean-field variational message passing. When a naive mean-field factorization constraint is imposed at node b, i.e., $q_b(\mathbf{s}_b) = \prod_{i \in \mathcal{E}(b)} q_i(s_i)$, the local free energy for factor b becomes

$$\begin{aligned} F[q_b, f_b] = \sum_{i \in \mathcal{E}} \int q_i(s_i) \log q_i(s_i) \, \mathrm{d}s_i \\ - \int \Big\{ \prod_{i \in \mathcal{E}(b)} q_i(s_i) \Big\} \log f_b(\mathbf{s}_b) \, \mathrm{d}\mathbf{s}_b. \end{aligned} \tag{21}$$

The variational message passing update (Winn and Bishop, 2005) from node b towards edge j is then given by

$$\mu_{jb}^{(k+1)}(s_j) = \exp\left(\int \Big\{ \prod_{\substack{i \in \mathcal{E}(b) \\ i \neq j}} q_i^{(k)}(s_i) \Big\} \log f_b(\mathbf{s}_b) \, \mathrm{d}\mathbf{s}_{b \setminus j} \right), \tag{22}$$

which is the exponentiated expected log-factor under the current beliefs of all other edges. We use naive mean-field factorization constraints at all non-conjugate factor boundaries in this paper.

A.5 Message rules for the alphabet

These rules, combined with a valid schedule, are sufficient to run inference on any factor graph composed from the alphabet.

Soft-dot The soft-dot factor node has the form:

$$f_*(z \mid \mathbf{w}, \boldsymbol{\phi}, \tau) = \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1}) .$$

We assume the node's incoming messages are marginals of the form:

$$\begin{aligned} q(z) &= \mathcal{N}(z \mid m_z, s_z^2) \\ q(\mathbf{w}) &= \mathcal{N}(\mathbf{w} \mid m_{\mathbf{w}}, s_{\mathbf{w}}^2) \\ q(\boldsymbol{\phi}) &= \mathcal{N}(\boldsymbol{\phi} \mid m_{\boldsymbol{\phi}}, s_{\boldsymbol{\phi}}^2) \\ q(\tau) &= \Gamma(\tau \mid \alpha_\tau, \beta_\tau) . \end{aligned}$$

The forward message towards z is:

$$\begin{aligned} \mu(z) &\propto \exp\big(\mathbb{E}_q\big[\log \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1})\big]\big) \\ &\propto \exp\big(\mathbb{E}_q\big[-\frac{\tau}{2}(z^2 - 2z\mathbf{w}^\top \boldsymbol{\phi} + \mathbf{w}^\top \boldsymbol{\phi}\boldsymbol{\phi}^\top \mathbf{w})\big]\big) \\ &\propto \exp\big(-\frac{\tau}{2}(z^2 - 2z m_{\mathbf{w}}^\top m_{\boldsymbol{\phi}} + m_{\mathbf{w}}^\top m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top m_{\mathbf{w}})\big]\big) \\ &\propto \mathcal{N}(z \mid m_{\mathbf{w}}^\top m_{\boldsymbol{\phi}}, \tau^{-1}) \end{aligned}$$

The backwards message towards $\mathbf{w}$ is:

$$\begin{aligned} \mu(\mathbf{w}) &\propto \exp\big(\mathbb{E}_q\big[\log \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1})\big]\big) \\ &\propto \exp\big(-\frac{\tau}{2}(m_z^2 - 2m_z \mathbf{w}^\top m_{\boldsymbol{\phi}} + \mathbf{w}^\top (m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top + S_{\boldsymbol{\phi}})\mathbf{w})\big) \\ &\propto \mathcal{N}(\mathbf{w} \mid m_{\mathbf{w}}, S_{\mathbf{w}}) , \end{aligned} \tag{23}$$

whose parameters are

$$\begin{aligned} m_{\mathbf{w}} &= \tau(m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top + S_{\boldsymbol{\phi}})(m_z m_{\boldsymbol{\phi}}) \\ S_{\mathbf{w}} &= \tau^{-1}(m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top + S_{\boldsymbol{\phi}})^{-1} . \end{aligned}$$

The backwards message towards $\boldsymbol{\phi}$ is:

$$\begin{aligned} \mu(\boldsymbol{\phi}) &\propto \exp\big(\mathbb{E}_q\big[\log \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1})\big]\big) \\ &\propto \exp\big(-\frac{\tau}{2}(m_z^2 - 2m_z m_{\mathbf{w}}^\top \boldsymbol{\phi} + \boldsymbol{\phi}^\top (m_{\mathbf{w}} m_{\mathbf{w}}^\top + S_{\mathbf{w}})\boldsymbol{\phi})\big) \\ &\propto \mathcal{N}(\boldsymbol{\phi} \mid m_{\boldsymbol{\phi}}, S_{\boldsymbol{\phi}}) , \end{aligned}$$

whose parameters are

$$\begin{aligned} m_{\boldsymbol{\phi}} &= \tau(m_{\mathbf{w}} m_{\mathbf{w}}^\top + S_{\mathbf{w}})(m_z m_{\mathbf{w}}) \\ S_{\boldsymbol{\phi}} &= \tau^{-1}(m_{\mathbf{w}} m_{\mathbf{w}}^\top + S_{\mathbf{w}})^{-1} . \end{aligned}$$

The backwards message towards τ is:

$$\begin{aligned} \mu(\tau) &\propto \exp\big(\mathbb{E}_q\big[\log \mathcal{N}(z \mid \mathbf{w}^\top \boldsymbol{\phi}, \tau^{-1})\big]\big) \\ &\propto \exp\big(\mathbb{E}_q\big[\frac{1}{2}\log\tau - \frac{\tau}{2}(z - \mathbf{w}^\top \boldsymbol{\phi})^2\big]\big) \\ &\propto \tau^{1/2} \exp\Big(-\frac{\tau}{2}\big((m_z - m_{\mathbf{w}}^\top m_{\boldsymbol{\phi}})^2 + s_z^2 \\ &\qquad + \mathrm{tr}[S_{\mathbf{w}}(m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top + S_{\boldsymbol{\phi}})]\big)\Big) \\ &\propto \mathcal{G}(\tau \mid \alpha_\tau, \beta_\tau) , \end{aligned} \tag{24}$$

whose parameters are:

$$\begin{aligned} &\alpha_\tau = 3/2 \\ &\beta_\tau = (m_z - m_{\mathbf{w}}^\top m_{\boldsymbol{\phi}})^2 + s_z^2 + \mathrm{tr}[S_{\mathbf{w}}(m_{\boldsymbol{\phi}} m_{\boldsymbol{\phi}}^\top + S_{\boldsymbol{\phi}})]. \end{aligned}$$

LogGamma The LogGamma distribution is a probability distribution on the real line with density

$$\mathcal{LG}(x \mid a, b) = \frac{e^{bx} e^{-e^x/a}}{a^b \Gamma(b)}, \quad -\infty < x < \infty,\ a > 0,\ b > 0 .$$

It arises as the change of variable $x = \log\gamma$ applied to a Gamma-distributed γ.

Exponential The exponential link factor node is:

$$f_{\exp}(\gamma \mid z) = \delta\big(\gamma - \exp(z)\big) .$$

The node's incoming messages are of the form:

$$\mu(\gamma) = \mathcal{G}(\gamma \mid \alpha_\gamma, \beta_\gamma) , \quad \mu(z) = \mathcal{N}(z \mid m_z, s_z^2) .$$

The forwards message to γ is:

$$\begin{aligned} \mu(\gamma) &= \int \delta\big(\gamma - \exp(z)\big)\mu(z)\mathrm{d}z \\ &= |\frac{\partial \log(\gamma)}{\partial \gamma}| \int \delta\big(\log(\gamma) - z\big)\mathcal{N}(z \mid m_z, s_z^2)\mathrm{d}z \\ &= \frac{1}{\gamma}\mathcal{N}(\log(\gamma) \mid m_z, s_z^2) \\ &= \mathcal{LN}(\gamma \mid m_z, s_2^2) . \end{aligned}$$

The backwards message towards z is obtained by an exact change of variable. Substituting $\gamma = \exp(z)$ with Jacobian $|\partial \exp(z)/\partial z| = \exp(z)$:

$$\begin{aligned} \mu(z) &= \int \delta\big(\gamma - \exp(z)\big)\mu(\gamma)\,\mathrm{d}\gamma \\ &= \Big|\frac{\partial \exp(z)}{\partial z}\Big| \int \delta\big(\exp(z) - \gamma\big)\,\mathcal{G}(\gamma \mid \alpha_\gamma, \beta_\gamma)\,\mathrm{d}\gamma \\ &= \exp(z) \cdot \mathcal{G}\big(\exp(z) \mid \alpha_\gamma, \beta_\gamma\big) \\ &= \exp(z) \cdot \frac{\beta_\gamma^{\alpha_\gamma}}{\Gamma(\alpha_\gamma)} \exp(z)^{\alpha_\gamma - 1} \exp\big(-\beta_\gamma \exp(z)\big) \\ &= \frac{\beta_\gamma^{\alpha_\gamma}}{\Gamma(\alpha_\gamma)} \exp(\alpha_\gamma z) \exp\big(-\beta_\gamma \exp(z)\big) \\ &= \mathcal{LG}\big(z \mid \beta_\gamma^{-1}, \alpha_\gamma\big) . \end{aligned}$$

Gamma The Gamma distribution factor node is:

$$f_{\mathcal{G}}(\gamma \mid \alpha, \beta) = \mathcal{G}(\gamma \mid \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \gamma^{\alpha - 1} \exp(-\beta\gamma) .$$

We assume incoming messages of the form:

$$q(\gamma) = \mathcal{G}(\gamma \mid \eta_\gamma, \zeta_\gamma) , \quad q(\beta) = \mathcal{G}(\beta \mid \eta_\beta, \zeta_\beta) ,$$

where η is a shape and ζ a rate parameter. The forward message towards γ is:

$$\begin{aligned} \mu(\gamma) &\propto \exp\big(\mathbb{E}_q\big[\log \Gamma(\gamma \mid \alpha, \beta)\big]\big) \\ &\propto \exp\big(\mathbb{E}_q\big[(\alpha - 1)\log(\gamma) - \beta\gamma\big]\big) \\ &\propto \gamma^{\alpha - 1} \exp\big(-\frac{\eta_\beta}{\zeta_\beta}\gamma\big) \\ &\propto \mathcal{G}(\gamma \mid \alpha, \frac{\eta_\beta}{\zeta_\beta}) \end{aligned}$$

The backwards message towards β is:

$$\begin{aligned}\mu(\beta) &\propto \exp\big(\mathbb{E}_q\big[\log\Gamma(\gamma\,|\,\alpha,\beta)\big]\big)\\ &\propto \exp\big(\mathbb{E}_q\big[\alpha\log\beta-\beta\gamma\big]\big)\\ &\propto \beta^{\alpha}\exp\big(-\beta\frac{\eta_\gamma}{\zeta_\gamma}\big)\\ &\propto \mathcal{G}(\beta\,|\,\alpha,\frac{\eta_\gamma}{\zeta_\gamma})\,.\end{aligned}$$

Normal The Normal factor node has the form:

$$f_{\mathcal{N}}(y\,|\,\mu,\tau)=\mathcal{N}(y\,|\,\mu,\tau^{-1})\,.$$

We assume the incoming messages are:

$$\begin{aligned}q(y,\mu)&=q(y\,|\,\mu,s_y^2)q(\mu\,|\,m_\mu,s_\mu^2)\\ &=\mathcal{N}\left(\begin{bmatrix}y\\ \mu\end{bmatrix}\,|\,\begin{bmatrix}m_\mu\\ m_\mu\end{bmatrix},\begin{bmatrix}s_y^2+s_\mu^2 & s_\mu^2\\ s_\mu^2 & s_\mu^2\end{bmatrix}\right)\\ q(\tau)&=\mathcal{G}(\tau\,|\,\alpha_\tau,\beta_\tau)\end{aligned}$$

Then the forward message to y is:

$$\begin{aligned}\mu(y)&\propto\exp\big(\mathbb{E}_q\big[\log\mathcal{N}(y\,|\,\mu,\tau^{-1})\big]\big)\\ &\propto\exp\big(\mathbb{E}_q\big[\frac{1}{2}\log\tau-\frac{\tau}{2}(y^2-2y\mu+\mu^2)\big]\big)\\ &\propto\exp\big(-\frac{\alpha_\tau}{2\beta_\tau}(y^2-2ym_\mu+m_\mu^2)\big)\\ &\propto\mathcal{N}\big(y\,|\,m_\mu,\frac{\alpha_\tau}{\beta_\tau}\big)\,,\end{aligned}$$

where $\psi(\cdot)$ is the digamma function. The backwards message towards μ is:

$$\begin{aligned}\mu(\mu)&\propto\exp\big(\mathbb{E}_q\big[\frac{1}{2}\log\tau-\frac{\tau}{2}(y^2-2y\mu+\mu^2)\big]\big)\\ &\propto\exp\big(-\frac{\alpha_\tau}{2\beta_\tau}(m_\mu^2-2m_\mu\mu+\mu^2)\big)\\ &\propto\mathcal{N}\big(\mu\,|\,m_\mu,\frac{\alpha_\tau}{\beta_\tau}\big)\end{aligned}$$

The backwards message towards τ is:

$$\begin{aligned}\mu(\tau)&\propto\exp\big(\mathbb{E}_q\big[\frac{1}{2}\log\tau-\frac{\tau}{2}(y^2-2y\mu+\mu^2)\big]\big)\\ &\propto\tau^{1/2}\exp\big(-\frac{\tau}{2}s_y^2\big)\\ &\propto\mathcal{G}(\tau\,|\,\frac{3}{2},s_y^2)\,.\end{aligned}$$

Equality The equality node has the following form:

$$f_{=}(x_1,x_2,x_3)=\delta\big(x_1-x_2\big)\delta\big(x_2-x_3\big)$$

The forward message to x_1 is:

$$\begin{aligned}\mu(x_1)&=\iint\delta\big(x_1-x_2\big)\delta\big(x_1-x_3\big)\mu_{x_2}(x_2)\mu_{x_3}(x_3)\mathrm{d}x_2\mathrm{d}x_3\\ &=\int\delta\big(x_1-x_3\big)\mu_{x_2}(x_1)\mu_{x_3}(x_3)\mathrm{d}x_3\\ &=\mu_{x_2}(x_1)\mu_{x_3}(x_1)\,.\end{aligned}$$

In words, the outgoing message is the product of incoming messages. In fact, the equality node is symmetric; the outgoing message towards any edge is the product of incoming messages on other edges (Loeliger, 2007).

A.6 Q-conjugacy on the γ edge

This section derives the closed-form stationarity condition for the marginal $q(\gamma)$ on an edge adjacent to the exponential link, under the Gamma form constraint $q(\gamma)=\mathcal{G}(\gamma\mid\alpha,\beta)$. The derivation parallels the Gaussian z-edge treatment in the main text but uses the Gamma exponential family structure.

Local BFE terms. On the γ edge, two messages arrive from the two adjacent factors. From the exponential link, the forward message is log-Normal (Section A.5):

$$\mu_{\gamma\leftarrow\exp}(\gamma)=\mathcal{LN}(\gamma\mid m_z,s_z^2)\,,\tag{25}$$

whose log-density is

$$\log\mathcal{LN}(\gamma\mid m_z,s_z^2)=-\frac{(\log\gamma-m_z)^2}{2s_z^2}-\log\gamma-\tfrac{1}{2}\log(2\pi s_z^2)\,.$$

From the far-side conjugate factor (e.g., the normal or gamma factor), the message is Gamma:

$$\mu_{\gamma\leftarrow\mathrm{far}}(\gamma)=\mathcal{G}(\gamma\mid a,b)\,,\tag{26}$$

whose log-density is $(a-1)\log\gamma-b\gamma+a\log b-\log\Gamma(a)$. The stationarity target is

$$\bar{\ell}(\gamma)=\log\mu_{\gamma\leftarrow\exp}(\gamma)+\log\mu_{\gamma\leftarrow\mathrm{far}}(\gamma)\,.\tag{27}$$

Gamma as exponential family. The Gamma distribution $\mathcal{G}(\gamma\mid\alpha,\beta)$ with shape α and rate β is an exponential family with natural parameters

$$\boldsymbol{\eta}=\begin{pmatrix}\eta_1\\ \eta_2\end{pmatrix}=\begin{pmatrix}\alpha-1\\ -\beta\end{pmatrix},\quad \eta_1>-1,\ \eta_2<0\,,\tag{28}$$

sufficient statistics $\mathbf{T}(\gamma)=(\log\gamma,\ \gamma)^\top$, and log-partition function

$$A(\boldsymbol{\eta})=\log\Gamma(\eta_1+1)-(\eta_1+1)\log(-\eta_2)\,.\tag{29}$$

Closed-form expectations. The expected sufficient statistics under $q(\gamma)=\mathcal{G}(\gamma\mid\alpha,\beta)$ are given by the gradient of the log-partition:

$$\nabla_{\boldsymbol{\eta}}A(\boldsymbol{\eta})=\begin{pmatrix}\mathbb{E}_{q(\gamma)}[\log\gamma]\\ \mathbb{E}_{q(\gamma)}[\gamma]\end{pmatrix}=\begin{pmatrix}\psi(\eta_1+1)-\log(-\eta_2)\\ -(\eta_1+1)/\eta_2\end{pmatrix},\tag{30}$$

where ψ denotes the digamma function. Both are analytic functions of $\boldsymbol{\eta}$.

The log-Normal message (25) additionally requires the second moment $\mathbb{E}_q\left[(\log\gamma)^2\right]$. By the variance identity,

$$\begin{aligned}\mathbb{E}_{q(\gamma)}\left[(\log\gamma)^2\right]&=\mathrm{Var}_q[\log\gamma]+\big(\mathbb{E}_{q(\gamma)}[\log\gamma]\big)^2\\ &=\psi'(\eta_1+1)+\big(\psi(\eta_1+1)-\log(-\eta_2)\big)^2,\end{aligned}\tag{31}$$

where ψ' is the trigamma function. This is again an analytic function of $\boldsymbol{\eta}$.

Writing $\bar{g} \triangleq \mathbb{E}_q[\log\gamma]$, $\bar{g}_2 \triangleq \mathbb{E}_q[(\log\gamma)^2]$, and $\bar{\gamma} \triangleq \mathbb{E}_q[\gamma]$, the expected energy under $q(\gamma)$ is

$$\mathbb{E}_q\left[-\bar{\ell}(\gamma)\right] = \tfrac{1}{2s_z^2}\,\bar{g}_2 - \tfrac{m_z}{s_z^2}\,\bar{g} + \bar{g} - (a-1)\,\bar{g} + b\,\bar{\gamma} + \text{const}\,, \tag{32}$$

where every expectation is a closed-form function of $\boldsymbol{\eta}$ via (30)–(31). This is the Q-conjugacy condition: $\mathbb{E}_q\left[-\bar{\ell}(\gamma)\right]$ and its gradient with respect to $\boldsymbol{\eta}$ are analytic functions of $\boldsymbol{\eta}$.

Fisher information matrix. The Fisher information matrix of the Gamma distribution in natural parameters is the Hessian of the log-partition (29):

$$\mathbf{F}(\boldsymbol{\eta}) = \nabla^2_{\boldsymbol{\eta}} A(\boldsymbol{\eta}) = \begin{pmatrix} \psi'(\eta_1+1) & -1/\eta_2 \\ -1/\eta_2 & (\eta_1+1)/\eta_2^2 \end{pmatrix}. \tag{33}$$

Stationarity condition. Setting the gradient of the local BFE contribution at the γ edge to zero and rearranging (as in the main text for the z edge) yields the fixed-point equation

$$\boldsymbol{\eta}^* = \mathbf{F}(\boldsymbol{\eta}^*)^{-1}\,\nabla_{\boldsymbol{\eta}^*}\mathbb{E}_{q^*(\gamma)}\left[-\bar{\ell}(\gamma)\right]\,. \tag{34}$$

Since $\mathbf{F}(\boldsymbol{\eta})$ (33) and $\nabla_{\boldsymbol{\eta}}\mathbb{E}_{q(\gamma)}\left[-\bar{\ell}(\gamma)\right]$ (32) are both closed-form functions of $\boldsymbol{\eta}$, the right-hand side of (34) is a closed-form expression of $\boldsymbol{\eta}^*$. This gives a closed-form fixed-point equation for the Gamma marginal $q^*(\gamma)$, analogous to the Gaussian fixed-point equation for $q^*(z)$ in the main text.

Differentiating (32) with respect to $\boldsymbol{\eta}$ and writing $\bar{g} = \psi(\eta_1+1) - \log(-\eta_2)$ and $c_1 = 1 - m_z/s_z^2 - (a-1)$, the gradient is

$$\nabla_{\boldsymbol{\eta}}\mathbb{E}_q\left[-\bar{\ell}(\gamma)\right] = \begin{pmatrix} \frac{\psi'(\eta_1+1)}{s_z^2}\left[\bar{g} + \frac{\psi''(\eta_1+1)}{2\,\psi'(\eta_1+1)}\right] + c_1\,\psi'(\eta_1+1) \\ \frac{\bar{g}}{s_z^2(-\eta_2)} + \frac{c_1}{(-\eta_2)} - \frac{b}{\eta_2^2} \end{pmatrix}, \tag{35}$$

where ψ'' is the polygamma function of order two. Every term is an analytic function of $\boldsymbol{\eta}$. Substituting (33) and (35) into (34) yields a closed-form fixed-point equation for $\boldsymbol{\eta}^*$, which we solve by natural gradient descent on the Gamma manifold as described in Section E.

B XOR Encoding Parameters

This appendix gives explicit parameter values that realize the XOR encoding of Fig. 6. We use features $\boldsymbol{\phi}(\mathbf{x}) = (x_1, x_2, 1)^\top$ with a bias term and two experts ($i \in \{1,2\}$) with depth-2 routing, each producing a log-precision $m_i(\mathbf{x})$ that feeds into a likelihood $y \sim \mathcal{N}(\hat{y}_i, e^{-m_i})$. Expert 1 predicts $\hat{y}_1$=0 and expert 2 predicts $\hat{y}_2$=1.

Router weights. Each expert has a routing vector $\mathbf{v}_i$ and shared sub-expert weight vectors $\mathbf{w}^L$, $\mathbf{w}^R$:

$$\mathbf{v}_1 = (14,\ 0,\ -7)^\top, \qquad \mathbf{v}_2 = (-14,\ 0,\ 7)^\top,$$
$$\mathbf{w}^L = (0,\ 10,\ 0)^\top, \qquad \mathbf{w}^R = (0,\ -10,\ 10)^\top.$$

The softdot precision is $\tau = 2000$ (approximating the sharp-routing limit).

Routing mechanism. For expert i at input $\mathbf{x}$, the router score is $h_i = \mathbf{v}_i^\top \boldsymbol{\phi}(\mathbf{x})$. Two switches with clamped weights ± 1 produce opposing activations: $\kappa_i^R = \exp(h_i)$, $\kappa_i^L = \exp(-h_i)$. The blended log-precision is

$$m_i(\mathbf{x}) \approx \frac{\kappa_i^R \cdot (\mathbf{w}^R)^\top \boldsymbol{\phi}(\mathbf{x}) + \kappa_i^L \cdot (\mathbf{w}^L)^\top \boldsymbol{\phi}(\mathbf{x})}{\kappa_i^R + \kappa_i^L},$$

which selects the sub-expert from the active branch.

Trace through all four inputs. Table 1 shows the routing scores and resulting log-precisions. Expert 1 achieves high $\gamma_1 = e^{m_1}$ (and thus high influence for its prediction $\hat{y}_1$=0) precisely at the XOR=0 inputs; expert 2 achieves high $\gamma_2 = e^{m_2}$ (high influence for $\hat{y}_2$=1) at the XOR=1 inputs.

Table 1: XOR encoding trace. For each input, the active branch selects the sub-expert whose output z yields a large log-precision m, giving that expert high influence $\gamma = e^m$.

x_1	x_2	XOR	h_1	m_1	h_2	m_2	Dominant
0	0	0	-7	≈ 10	$+7$	≈ 0	Exp. 1 ($\hat{y}$=0)
0	1	1	-7	≈ 0	$+7$	≈ 10	Exp. 2 ($\hat{y}$=1)
1	0	1	$+7$	≈ 0	-7	≈ 10	Exp. 2 ($\hat{y}$=1)
1	1	0	$+7$	≈ 10	-7	≈ 0	Exp. 1 ($\hat{y}$=0)

C Comparison with Neural Universal Approximation

The universal approximation result of Corollary 1 parallels classical theorems for neural networks. Cybenko (1989) proved that single-hidden-layer networks with sigmoidal activation functions are universal approximators. Hornik (1991) extended this to arbitrary bounded nonconstant activation functions, showing that the multilayer feedforward architecture itself—not the specific choice of activation—is the source of universality (Theorems 1 and 2 therein).

The structural analogy with our framework is direct. In the sharp-routing limit ($\tau \to \infty$), the split-branch construction of Section 3.3 partitions the input space into regions separated by axis-aligned hyperplanes, selecting a different expert prediction in each region. This is the same piecewise-constant mechanism that underlies the neural proofs: a sufficiently fine partition of the input space, combined with a constant approximation in each cell, can approximate any continuous function on a compact domain to arbitrary accuracy.

The key difference lies in the inference mechanism:

- Neural networks are trained via gradient descent on a loss function, producing point estimates of all parameters.
- Our framework optimizes the Bethe free energy with closed-form message passing, producing marginal posterior distributions over all parameters and latent variables.

As a consequence, epistemic uncertainty is available at every level of the hierarchy without additional cost—no ensembles-of-ensembles, no MC dropout, no Laplace approximation. Figure 6 illustrates this concretely: the posterior standard deviation reveals where the model is uncertain about its routing decisions, information that is absent from any point-estimate approach.

D Models in GraphPPL.jl

The models from Section 3 can be written directly in GraphPPL.jl (Nuijten et al., 2024), a probabilistic programming language that compiles model specifications into Forney-style factor graphs for inference in RxInfer.jl (Bagaev and De Vries, 2023).

In the `@model` block, each tilde statement creates one factor node. The mapping to the alphabet (1) is as follows:

GraphPPL	Alphabet letter
`NormalMeanPrecision`	$f_{\mathcal{N}}$ (1d)
`GammaShapeRate`	$f_{\mathcal{G}}$ (1c)
`softdot`	f_* (1a)
`Log`	$f_{\exp}$ (1b)

Equality nodes $f_=$ (1e) are inserted automatically whenever a variable appears in more than two factors.

D.1 Depth 0: static ensemble

The following listing implements the static ensemble model introduced in Section 3.1. The outer loop over i corresponds to the $\cdots$ in Fig. 2 (repetition over experts), and the inner loop over j corresponds to the $\vdots$ (repetition over observations).

```
@model function static_ensemble(
    n_forecasters, X, y, priors
)
  for i = 1:n_forecasters
    gamma[i] ~ priors[:gamma][i]
  end
  for i = 1:n_forecasters
    for j = 1:length(y)
      y[j] ~ NormalMeanPrecision(
        X[i,j], gamma[i]
      )
    end
  end
end
```

D.2 Depth 1: Precision-Gated Experts

The following listing implements the Precision-Gated Experts model introduced in Section 3.2. The loops over i and j carry the same meaning as in Section D.1 ($\cdots$ and $\vdots$ in Fig. 3c).

```
@model function pge(n_forecasters,
    n_obs, features, predictions,
    y, priors
)
  for i = 1:n_forecasters
    w[i]    ~ priors[:w][i]
    tau[i]  ~ priors[:tau][i]
    beta[i] ~ priors[:beta][i]
  end
  for j = 1:n_obs
    for i = 1:n_forecasters
      z[i,j] ~ softdot(
        features[j], w[i], tau[i]
      )
      gamma[i,j] ~ GammaShapeRate(
        1.0, beta[i]
      )
      z[i,j] ~ Log(gamma[i,j])
      y[j] ~ NormalMeanPrecision(
        predictions[i,j], gamma[i,j]
      )
    end
  end
end
```

The constraints and initialization blocks declare the factorization and form constraints from Section 4.1:

```
@constraints function pge_constraints()
  q(w, z, gamma, tau, beta) =
    q(w)q(z, gamma)q(tau)q(beta)
  q(z)     :: ProjectedTo(
    NormalMeanVariance)
  q(gamma) :: ProjectedTo(Gamma)
end

@initialization function pge_init(priors)
  q(w)     = priors[:w]
  q(z)     = NormalMeanVariance(0.0, 1.0)
  q(gamma) = GammaShapeScale(1.0, 1.0)
  q(tau)   = priors[:tau]
  q(beta)  = priors[:beta]
end
```

Given these three blocks, RxInfer.jl constructs the factor graph, derives the message passing schedule, and runs inference automatically — no update equations are written by the user.

D.3 Noisy Experts

The noisy experts model (Fig. 8) augments the Precision-Gated Experts (Section D.2) with an additional noise layer for each expert. Instead of treating each forecaster prediction $\hat{y}_{i,j}$ as an exact input, the model introduces a latent variable $\text{pred}_{i,j}$ connected to the observed prediction through a Normal factor with learned precision κ_i.

Training model. During training, y is a data variable (observed). The factorization between $q(y, \text{pred})$ is not required: since y is observed, the structured dependence between y and pred is absorbed by the data and does not affect the variational updates.

```
@model function noisy_experts(
    n_forecasters, n_obs,
    features, predictions, y, priors
)
  local w, z, gamma, tau, beta, kappa, pred
  for i = 1:n_forecasters
    w[i]     ~ priors[:w][i]
    tau[i]   ~ priors[:tau][i]
    beta[i]  ~ priors[:beta][i]
    kappa[i] ~ priors[:kappa][i]
  end
  for j = 1:n_obs
    for i = 1:n_forecasters
      z[i,j] ~ softdot(
        features[j], w[i], tau[i]
      ) where {meta = LowRankMeta()}
      gamma[i,j] ~ GammaShapeRate(
        1.0, beta[i]
```

```
    )
    z[i,j] ~ Log(gamma[i,j])
    pred[i,j] ~ NormalMeanPrecision(
      predictions[i,j], kappa[i]
    )
    y[j] ~ NormalMeanPrecision(
      pred[i,j], gamma[i,j]
    )
  end
 end
end
```

Prediction model. For prediction, y is no longer observed but is instead a latent variable with an uninformative prior. In this setting, the choice between mean-field and belief propagation on the pred $\rightarrow y$ edge becomes significant. Under the mean-field variational rule the outgoing message towards y from $f_{\mathcal{N}}(\text{pred}, \gamma)$ uses only $\mathbb{E}_q[\text{pred}]$:

```
# Mean-field rule: ignores Var(pred)
@rule NormalMeanPrecision(:out, Marginalisation) (
    q_mu::Any, q_tau::Any
) = NormalMeanPrecision(
    mean(q_mu), mean(q_tau)
)
```

This sets the predictive precision to $\mathbb{E}[\gamma_{i,j}]$ alone, discarding the forecaster noise κ_i^{-1} entirely. By contrast, the belief propagation rule propagates the full uncertainty in pred:

```
# BP rule: additive variance 1/kappa + 1/gamma
@rule NormalMeanPrecision(:out, Marginalisation) (
    m_mu::NormalDistributionsFamily,
    m_tau::PointMass
) = begin
    m_mu_mean, m_mu_cov = mean_cov(m_mu)
    NormalMeanPrecision(m_mu_mean,
      inv(m_mu_cov + inv(mean(m_tau))))
end
```

Here $\text{Var}(\text{pred}_{i,j}) = \kappa_i^{-1}$, so the predictive variance becomes $\kappa_i^{-1} + \gamma_{i,j}^{-1}$. The prediction model activates this rule by keeping $q(y, \text{pred})$ in a single structured factor and marking y as uninformative:

```
@model function noisy_experts_prediction(
    n_forecasters, n_obs,
    features, predictions, priors
)
  local w, z, gamma, tau, beta, kappa, pred, y
  for i = 1:n_forecasters
    w[i]     ~ priors[:w][i]
    tau[i]   ~ priors[:tau][i]
    beta[i]  ~ priors[:beta][i]
    kappa[i] ~ priors[:kappa][i]
  end
  for j = 1:n_obs
    for i = 1:n_forecasters
      z[i,j] ~ softdot(
        features[j], w[i], tau[i]
      ) where {meta = LowRankMeta()}
      gamma[i,j] ~ GammaShapeRate(
        1.0, beta[i]
      )
      z[i,j] ~ Log(gamma[i,j])
      pred[i,j] ~ NormalMeanPrecision(
        predictions[i,j], kappa[i]
      )
      y[j] ~ NormalMeanPrecision(
        pred[i,j], gamma[i,j]
      )
    end
    y[j] ~ Uninformative()
  end
end
```

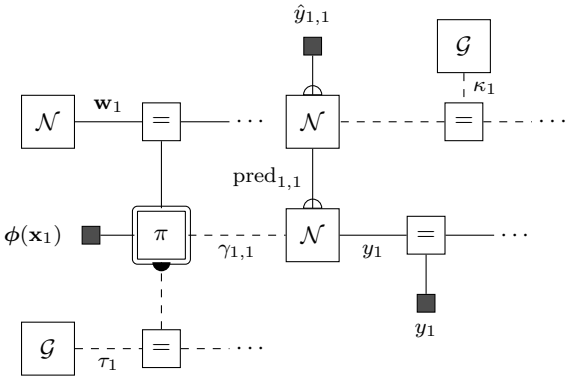


Figure 8: Noisy experts factor graph, shown for expert i=1 and observation j=1. Compared to Depth 1 (Fig. 3c), the clamped prediction $\hat{y}_{1,1}$ is replaced by a latent variable $\text{pred}_{1,1}$, connected to the observed forecaster output through an additional Normal factor with learned precision κ_1. This separates forecaster disagreement from irreducible predictive uncertainty.

The constraints and initialization blocks for both models:

```
@constraints function noisy_constraints(
    priors, prediction
)
  if prediction
    q(w, z, gamma, tau, beta,
      kappa, pred, y) =
      q(w)q(z, gamma)q(tau)
      q(beta)q(kappa)q(y, pred)
  else
    q(w, z, gamma, tau, beta,
      kappa, pred) =
      q(w)q(z, gamma)q(tau)
      q(beta)q(kappa)q(pred)
  end
  q(z) :: ProjectedTo(
    NormalMeanVariance)
  q(gamma) :: ProjectedTo(Gamma)
end

@initialization function noisy_init(priors)
  q(w)     = deepcopy(priors[:w])
  q(z)     = NormalMeanVariance(0.0, 1.0)
  q(gamma) = GammaShapeScale(1.0, 1.0)
  q(tau)   = priors[:tau]
  q(beta)  = priors[:beta]
  q(kappa) = priors[:kappa]
  q(pred)  = NormalMeanVariance(0.0, 1.0)
end
```

E Experimental Details

E.1 Base forecasters

We train $n = 7$ forecasters independently using a fixed input sequence length of 96 time steps.

- **DLinear** (Zeng et al., 2023): decomposes the input series into trend and seasonal components, then applies separate linear layers to forecast each part. Channel-independent.
- **NLinear** (Zeng et al., 2023): applies a linear forecasting layer after normalizing the input sequence. Channel-independent.
- **LSTM** (Hochreiter and Schmidhuber, 1997): recurrent neural network for temporal dependencies.

- **CNN**: cross-channel one-dimensional convolutional model applied across time-series channels.
- **NConv**: similar to NLinear, but replaces the linear layer with a one-dimensional convolution.
- **Quantile 10 / Quantile 90**: constant forecasters corresponding to the 10th and 90th quantiles.

All neural networks are trained with the AdamW optimizer (Loshchilov and Hutter, 2019), learning rate 0.001, for 50 epochs; the best checkpoint is selected by validation performance. For ETTh1 and ETTh2, forecasting is performed on a univariate target; for Exchange Rate, Electricity, and Traffic, all dimensions are predicted.

E.2 Feature extraction

To obtain a compact representation of multivariate time series with input length 96, we train a variational autoencoder (VAE) (Kingma and Welling, 2013) that reconstructs the input sequence through a latent representation of dimension 64.

E.3 Baseline gating methods

For comparison we consider two gating variants based on the classical Mixture-of-Experts architecture (Jacobs et al., 1991): **MoE**, a single-layer neural network with softmax output, and **MoE Big**, a two-layer fully connected neural network with softmax output. Both are trained with early stopping: training halts when the loss decrease falls below $\delta = 10^{-5}$, with 100 epochs and patience 50.

NLL computation for the softmax baselines. The softmax gating weights $w_i = e^{z_i}/\sum_j e^{z_j}$ are positive and sum to one, so the predictive mean is $\hat{y} = \sum_i w_i \hat{y}_i$. To obtain a predictive variance we interpret the unnormalized scores e^{z_i} as precisions: setting $\gamma_i = e^{z_i}$ recovers the same weighted average $\hat{y} = \sum_i \gamma_i \hat{y}_i / \sum_i \gamma_i$ and yields a predictive precision $\sum_j e^{z_j}$, giving a Gaussian predictive distribution $\mathcal{N}(\hat{y}, (\sum_j e^{z_j})^{-1})$ from which NLL is computed.

This interpretation is inherently limited: the softmax is shift-invariant, so replacing $z_i \to z_i + c$ leaves the weights (and hence the predictive mean) unchanged but scales the implied precision by e^c. No additional term in the training objective can resolve this ambiguity, since the gradient of the softmax output with respect to a global shift is zero. Consequently, the NLL values reported for the MoE baselines depend on an unidentified degree of freedom. In PGE, the precisions are explicit random variables equipped with priors, so no such ambiguity arises and the predictive uncertainty is fully determined by the model.

Table 2: Characteristics of the benchmark datasets, including dimensionality, length, split ratio (train:val:test), and sampling frequency.

Dataset	Dims	Length	Split ratio	Frequency
ETTh1	7	14307	6:2:2	15 min
ETTh2	7	14307	6:2:2	15 min
Electricity	321	26211	7:1:2	Hourly
Traffic	862	17451	7:1:2	Hourly
Exchange	8	7207	7:1:2	Daily

E.4 Our framework

All configurations of our framework are trained for 5 variational message passing iterations and use 3 iterations during prediction.

Solving the non-conjugate fixed point. The stationarity condition (8) is a nonlinear fixed-point equation whose right-hand side is available in closed form but whose solution is not. We solve it by casting it as the minimization of the local free energy $F_z[\boldsymbol{\eta}]$ (5) over the natural-parameter space of the Gaussian, which carries the Fisher information metric and forms a Riemannian manifold. Optimization on this manifold is performed with Manopt.jl (Bergmann, 2022), using the exponential family manifold interface of Lukashchuk et al. (2025). We use an Armijo line search with initial step size 10^{-2} and a bounded-norm update rule (maximum step norm 0.5), running up to 50 iterations with a tolerance of 10^{-6}.

F Full Results

F.1 Dataset info

The statistical characteristics of the datasets include the dataset names, the number of input dimensions, corresponding to the number of individual time series in each dataset, the length of each time series, the training, validation, and test splits adopted for expert and ensemble training, and the sampling frequency of the temporal index.

F.2 Methods parameter count

Table 3 reports the trainable parameter counts for all considered ensemble methods in the VAE-based feature setting with input dimensionality $d = 65$ and number of experts $K = 7$. Here, $d = 65$ consists of a 64-dimensional latent representation produced by the VAE together with one additional constant feature. The table presents both the general formulas and the corresponding numerical values for the experimental setup used in this work.

As expected, the Static model has the smallest number of parameters, since it only learns global coefficients for each expert. The MoE model is also relatively compact due to its single-layer softmax gating mechanism, whereas MoE Big is substantially larger because

Table 3: Parameter-count formulas and counts for the VAE-feature setting ($d = 65$, $K = 7$).

Method	Formula for N_{total}	Count ($d = 65$)
Static	$2K$	14
MoE	$K(d+1)$	462
MoE Big	$H(d+1) + K(H+1)$	14023
Dyn. Diag.	$2K(d+2)$	938
Noisy Diag.	$K(2d+6)$	952
Dynamic	$K\left(d + \frac{d(d+1)}{2} + 4\right)$	15498
Noisy	$K\left(d + \frac{d(d+1)}{2} + 6\right)$	15512

it employs a deeper gating network. The Dynamic and Noisy variants require the largest number of parameters because they model full covariance structures, whose complexity grows quadratically with d. By contrast, the diagonal variants remain much more parameter-efficient, since they restrict uncertainty modeling to diagonal terms only.

These parameter counts are later used to analyze the trade-off between forecasting performance and model complexity.

F.3 Pareto Frontier

The Pareto frontier was constructed for the Static, Dynamic, Dynamic Diagonal, Noisy, and Noisy Diagonal models. It compares the radar-plot area percentage shown in Figures 7a and 7b against the number of trainable parameters used by each model, with parameter counts reported in Table 3.

F.4 Experts Performance

This table presents the MAE and MSE results for each expert model on all datasets described in Table 2, evaluated over the forecasting horizons $96, 192, 336, 720$. All experts were trained with an input sequence length of 96. The considered expert models are CNN, DLinear, LSTM, NLinear, NConv, as well as the quantile predictors q10 and q90.

F.5 Performance of ensembles

The MSE and NLL values are presented for the Static, Dynamic, Dynamic Diagonal, Noisy, Noisy Diagonal, MoE, and MoE Big ensemble methods in Tables 6 and 7. The MoE model is implemented as a single-layer neural network with softmax-based weighting, whereas MoE Big is a two-layer softmax-based neural network with a higher number of trainable parameters, as detailed in Table 3. All ensemble approaches were evaluated on the same five datasets listed in Table 2. In all cases, the ensemble input consisted of sequences of length 96, which were compressed into a 64-dimensional latent state using a variational autoencoder trained to reconstruct all dataset features over the same input length.

GP comparison. We additionally compare the full-covariance Dynamic ensemble with univariate temporal GP regression in its exact two-state linear-Gaussian representation. Each GP conditions on the preceding 96 standardized OT observations and predicts one target at horizon H. The signal variance is fixed to one; a validation-MSE grid search selects $\ell = 128$ days in every setting and selects the observation-noise variance separately for each dataset and horizon.

This is not a fully like-for-like comparison. Dynamic is a second-stage regressor: it receives the forecasts of seven pretrained experts and uses a multivariate latent context to decide how to combine them. The GP is a standalone autoregressor: it produces its forecast directly from only the preceding values of the target series. Both methods are evaluated on the same standardized targets, horizons, and test splits with the same metrics, but they do not receive equivalent inputs or training resources. Accordingly, Table 5 should be read as a comparison with a classical probabilistic forecasting reference, not as a controlled ablation that attributes the performance difference solely to the model class. Metrics and confidence intervals follow the same per-origin convention as Table 6.

Table 5: Dynamic ensemble versus Matérn-3/2 GP state-space regression on the standardized ETTh targets. Values are mean ± 95% confidence interval; **bold** identifies the lower mean in each row.

Dataset	H	Metric	Dynamic	GP–SSM
ETTh1	96	MSE	**0.128 ± 0.007**	0.154 ± 0.008
		NLL	**0.412 ± 0.019**	0.799 ± 0.006
	192	MSE	**0.115 ± 0.006**	0.138 ± 0.007
		NLL	**0.370 ± 0.019**	0.793 ± 0.006
	336	MSE	**0.096 ± 0.005**	0.127 ± 0.006
		NLL	**0.314 ± 0.015**	1.004 ± 0.003
	720	MSE	**0.112 ± 0.005**	0.166 ± 0.008
		NLL	**0.376 ± 0.015**	0.880 ± 0.006
ETTh2	96	MSE	**0.346 ± 0.016**	0.428 ± 0.021
		NLL	**0.934 ± 0.033**	1.025 ± 0.034
	192	MSE	**0.336 ± 0.016**	0.390 ± 0.020
		NLL	**0.924 ± 0.036**	0.995 ± 0.016
	336	MSE	**0.353 ± 0.017**	0.401 ± 0.022
		NLL	**0.961 ± 0.036**	1.377 ± 0.005
	720	MSE	**0.321 ± 0.016**	0.655 ± 0.034
		NLL	**0.870 ± 0.032**	1.257 ± 0.040

F.6 Forecasting Comparison

Figure 10 compares Static, Noisy Diagonal, and MoE forecasts with confidence intervals on Electricity at $H = 720$. It shows the Static expert contributions, the Noisy Diagonal contributions and Top Share, and the MoE softmax gate weights. Figure 11 illustrates that MoE can be overconfident and collapse onto an expert that is not best for the data.

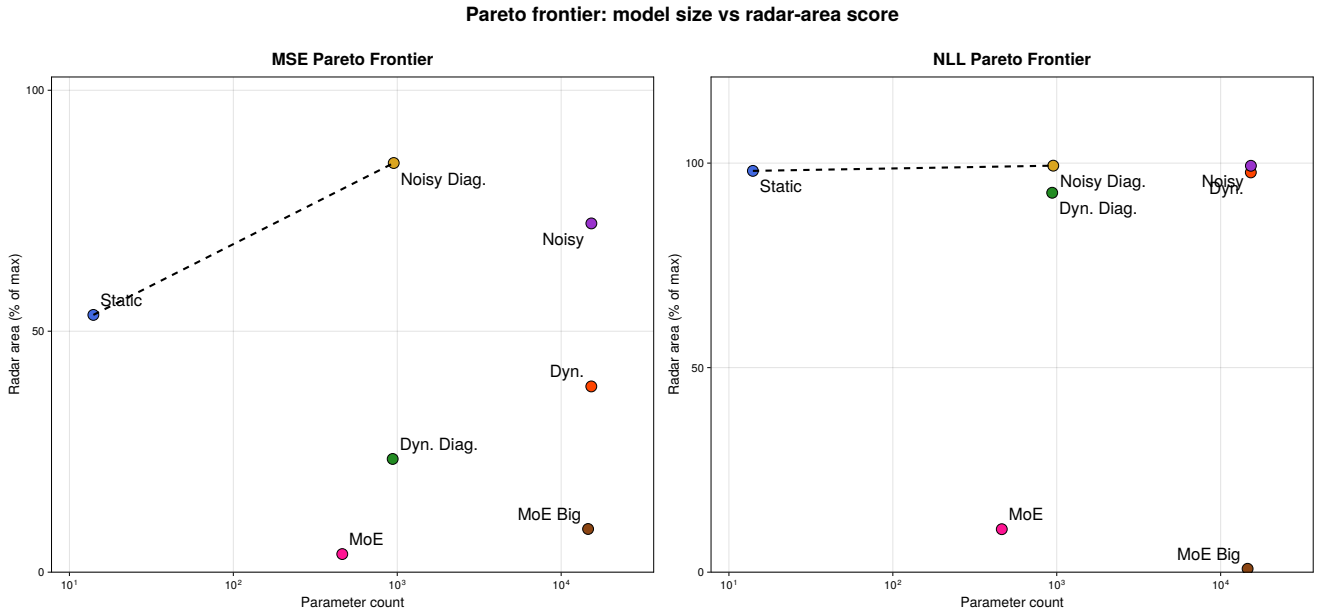


Figure 9: Pareto frontier relating model size to radar-chart area. Static and Noisy Diagonal offer the strongest trade-off between complexity and predictive performance.

Table 4: Comparison of MSE and MAE for baseline models and quantile-based predictors with input sequence length 96.

Dataset	H	CNN		DLinear		LSTM		NLinear		NConv		q10		q90	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Exchange_rate	96	2.455	1.187	0.157	0.306	1.150	0.881	0.183	0.317	0.180	0.314	1.439	0.958	3.423	1.411
	192	3.559	1.448	0.325	0.446	2.589	1.337	0.400	0.478	0.410	0.479	1.379	0.962	3.584	1.447
	336	2.295	1.184	0.558	0.606	1.536	1.023	0.804	0.680	0.820	0.680	1.407	0.974	3.751	1.491
	720	4.902	1.706	1.495	0.984	1.982	1.147	2.187	1.166	2.426	1.225	2.269	1.262	2.313	1.125
	Avg	3.303	1.381	0.634	0.585	1.814	1.097	0.894	0.660	0.959	0.674	1.623	1.039	3.268	1.368
ETTh1	96	0.456	0.586	0.156	0.312	0.412	0.536	0.147	0.301	0.181	0.333	0.611	0.680	0.601	0.675
	192	0.248	0.398	0.162	0.318	0.278	0.419	0.137	0.295	0.178	0.323	0.538	0.630	0.647	0.708
	336	0.298	0.449	0.137	0.309	0.231	0.377	0.135	0.291	0.174	0.328	0.501	0.605	0.708	0.750
	720	1.006	0.905	0.290	0.462	0.275	0.409	0.216	0.361	0.310	0.440	0.465	0.583	0.799	0.809
	Avg	0.502	0.585	0.187	0.350	0.299	0.435	0.159	0.312	0.211	0.356	0.529	0.624	0.689	0.736
ETTh2	96	1.796	1.114	0.313	0.448	0.851	0.752	0.339	0.465	0.427	0.505	3.160	1.561	1.086	0.825
	192	1.017	0.772	0.270	0.414	0.855	0.751	0.320	0.455	0.414	0.504	2.943	1.496	1.081	0.823
	336	1.109	0.853	0.269	0.413	0.848	0.715	0.345	0.470	0.423	0.508	2.787	1.449	1.081	0.824
	720	1.957	1.228	0.385	0.516	1.232	0.966	0.681	0.661	0.685	0.655	2.683	1.420	1.198	0.875
	Avg	1.470	0.992	0.309	0.448	0.947	0.796	0.421	0.513	0.487	0.543	2.893	1.482	1.111	0.837
Electricity	96	0.350	0.432	0.176	0.267	0.396	0.443	0.178	0.266	0.348	0.363	2.778	1.335	1.838	1.095
	192	0.388	0.462	0.251	0.326	0.335	0.403	0.260	0.324	0.307	0.345	2.786	1.337	1.872	1.105
	336	0.364	0.434	0.223	0.314	0.337	0.407	0.229	0.310	0.248	0.319	2.800	1.340	1.922	1.119
	720	0.418	0.462	0.344	0.404	0.425	0.462	0.370	0.407	0.462	0.449	2.832	1.347	2.049	1.155
	Avg	0.380	0.448	0.249	0.336	0.373	0.429	0.259	0.341	0.341	0.369	2.799	1.340	1.920	1.119
Traffic	96	0.743	0.422	0.475	0.308	0.845	0.469	0.478	0.305	1.205	0.596	2.738	1.139	3.308	1.442
	192	0.737	0.423	0.710	0.431	0.817	0.443	0.716	0.430	0.831	0.453	2.746	1.140	3.161	1.425
	336	0.730	0.403	0.547	0.345	0.740	0.407	0.556	0.337	0.622	0.344	2.752	1.141	3.109	1.437
	720	0.800	0.447	0.814	0.473	1.055	0.576	0.820	0.466	1.191	0.588	2.772	1.144	3.073	1.426
	Avg	0.753	0.424	0.637	0.389	0.864	0.474	0.643	0.385	0.962	0.496	2.752	1.141	3.163	1.433

Table 6: MSE for Static, Dynamic (Dyn.), Dynamic Diagonal (Dyn. Diag.), Noisy Experts (Noisy), Noisy Diagonal (Noisy Diag.), Mixture of Experts (MoE), and MoE Big ensembles, compared against the best baseline model selected by lowest baseline MSE for each dataset and horizon (Best). Rows are organized by dataset and horizon; the Avg row averages the four horizons. **Bold** indicates the best result, blue underlined the second best. The ± values denote 95% confidence intervals. NLL for the same runs is reported in Table 7.

Dataset	H	Static	Dyn.	Dyn. Diag.	Noisy	Noisy Diag.	MoE	MoE Big	Best
ETTh1	96	0.132 ± 0.01	0.128 ± 0.01	**0.123 ± 0.01**	0.131 ± 0.01	0.129 ± 0.01	0.147 ± 0.01	0.156 ± 0.01	0.147
	192	0.114 ± 0.01	0.115 ± 0.01	**0.112 ± 0.01**	0.116 ± 0.01	0.115 ± 0.01	0.137 ± 0.01	0.137 ± 0.01	0.137
	336	0.105 ± 0.01	0.096 ± 0.00	**0.095 ± 0.00**	0.098 ± 0.01	0.097 ± 0.00	0.172 ± 0.01	0.231 ± 0.01	0.135
	720	0.145 ± 0.01	**0.112 ± 0.01**	0.113 ± 0.01	0.115 ± 0.01	0.116 ± 0.01	0.215 ± 0.01	0.216 ± 0.01	0.216
	Avg	0.124 ± 0.01	0.113 ± 0.01	**0.111 ± 0.01**	0.115 ± 0.01	0.114 ± 0.01	0.168 ± 0.01	0.185 ± 0.01	0.159
ETTh2	96	0.325 ± 0.02	0.346 ± 0.02	0.344 ± 0.02	0.340 ± 0.02	0.337 ± 0.02	0.339 ± 0.02	**0.313 ± 0.02**	0.313
	192	0.296 ± 0.02	0.336 ± 0.02	0.332 ± 0.02	0.327 ± 0.02	0.321 ± 0.02	0.320 ± 0.02	0.320 ± 0.02	**0.270**
	336	0.304 ± 0.02	0.353 ± 0.02	0.350 ± 0.02	0.337 ± 0.02	0.333 ± 0.02	0.269 ± 0.01	0.269 ± 0.01	**0.269**
	720	0.480 ± 0.02	**0.321 ± 0.02**	0.323 ± 0.02	0.326 ± 0.02	0.322 ± 0.02	0.385 ± 0.02	0.385 ± 0.02	0.385
	Avg	0.351 ± 0.02	0.339 ± 0.02	0.337 ± 0.02	0.333 ± 0.02	0.328 ± 0.02	0.328 ± 0.02	0.322 ± 0.02	**0.309**
Exchange rate	96	0.326 ± 0.01	0.398 ± 0.01	0.239 ± 0.01	0.306 ± 0.01	0.262 ± 0.01	2.455 ± 0.03	0.183 ± 0.01	**0.157**
	192	0.820 ± 0.03	0.821 ± 0.03	0.521 ± 0.02	0.509 ± 0.02	0.453 ± 0.02	3.559 ± 0.11	3.559 ± 0.11	**0.325**
	336	1.161 ± 0.03	1.420 ± 0.03	1.071 ± 0.02	0.854 ± 0.02	0.749 ± 0.02	2.295 ± 0.07	2.295 ± 0.07	**0.558**
	720	1.675 ± 0.04	1.754 ± 0.03	**1.140 ± 0.01**	1.515 ± 0.03	1.472 ± 0.03	1.982 ± 0.05	1.982 ± 0.05	1.495
	Avg	0.995 ± 0.02	1.098 ± 0.03	0.743 ± 0.01	0.796 ± 0.02	0.734 ± 0.02	2.573 ± 0.07	2.005 ± 0.06	**0.634**
Electricity	96	0.182 ± 0.01	0.197 ± 0.01	0.198 ± 0.01	0.188 ± 0.01	0.188 ± 0.01	0.176 ± 0.01	**0.176 ± 0.01**	0.176
	192	**0.225 ± 0.01**	0.232 ± 0.01	0.248 ± 0.01	0.228 ± 0.01	0.228 ± 0.01	0.251 ± 0.01	0.251 ± 0.01	0.251
	336	**0.207 ± 0.01**	0.216 ± 0.01	0.229 ± 0.01	0.210 ± 0.01	0.209 ± 0.01	0.224 ± 0.01	0.230 ± 0.01	0.223
	720	**0.308 ± 0.01**	0.321 ± 0.01	0.320 ± 0.01	0.311 ± 0.01	0.311 ± 0.01	0.340 ± 0.01	0.370 ± 0.01	0.344
	Avg	**0.231 ± 0.01**	0.242 ± 0.01	0.249 ± 0.01	0.234 ± 0.01	0.234 ± 0.01	0.248 ± 0.01	0.257 ± 0.01	0.248
Traffic	96	0.503 ± 0.02	0.528 ± 0.02	0.556 ± 0.03	0.540 ± 0.03	0.540 ± 0.03	**0.472 ± 0.02**	0.477 ± 0.02	0.475
	192	0.629 ± 0.03	0.648 ± 0.03	0.684 ± 0.03	0.622 ± 0.03	**0.622 ± 0.03**	0.677 ± 0.03	0.690 ± 0.03	0.710
	336	**0.515 ± 0.02**	0.528 ± 0.03	0.567 ± 0.03	0.517 ± 0.02	0.517 ± 0.02	0.552 ± 0.02	0.556 ± 0.03	0.547
	720	0.748 ± 0.03	0.757 ± 0.04	0.782 ± 0.04	0.741 ± 0.03	**0.741 ± 0.03**	0.766 ± 0.03	1.036 ± 0.04	0.800
	Avg	**0.599 ± 0.03**	0.615 ± 0.03	0.647 ± 0.03	0.605 ± 0.03	0.605 ± 0.03	0.617 ± 0.03	0.690 ± 0.03	0.633

Table 7: NLL for Static, Dynamic (Dyn.), Dynamic Diagonal (Dyn. Diag.), Noisy Experts (Noisy), Noisy Diagonal (Noisy Diag.), Mixture of Experts (MoE), and MoE Big ensembles. Rows are organized by dataset and horizon; the Avg row averages the four horizons. **Bold** indicates the best result, blue underlined the second best. The ± values denote 95% confidence intervals. The Best baseline column is omitted here because the deterministic baselines do not provide predictive distributions, and non-finite values are shown as ×. MSE for the same runs is reported in Table 6.

Dataset	H	Static	Dyn.	Dyn. Diag.	Noisy	Noisy Diag.	MoE	MoE Big
ETTh1	96	1.138 ± 0.10	**0.412 ± 0.02**	0.451 ± 0.01	0.447 ± 0.02	0.468 ± 0.01	$(2.1\pm1.7)\times10^{25}$	$(2.3\pm1.3)\times10^{24}$
	192	0.759 ± 0.08	**0.370 ± 0.02**	0.425 ± 0.01	0.408 ± 0.02	0.433 ± 0.01	$(1.1\pm1.1)\times10^{22}$	$(1.2\pm0.9)\times10^{19}$
	336	0.610 ± 0.07	**0.314 ± 0.01**	0.384 ± 0.01	0.362 ± 0.01	0.393 ± 0.01	$(1.1\pm0.7)\times10^{22}$	$(2.5\pm1.9)\times10^{14}$
	720	0.801 ± 0.07	**0.376 ± 0.02**	0.439 ± 0.01	0.424 ± 0.01	0.454 ± 0.01	$(9.1\pm6.9)\times10^{11}$	$(1.5\pm1.1)\times10^{17}$
	Avg	0.827 ± 0.08	**0.368 ± 0.02**	0.424 ± 0.01	0.410 ± 0.01	0.437 ± 0.01	$(5.1\pm4.3)\times10^{24}$	$(5.7\pm3.3)\times10^{23}$
ETTh2	96	2.333 ± 0.14	0.934 ± 0.03	0.885 ± 0.02	0.897 ± 0.03	**0.874 ± 0.02**	$(3.1\pm2.5)\times10^{23}$	$(1.1\pm1.1)\times10^{31}$
	192	1.736 ± 0.12	0.924 ± 0.04	0.872 ± 0.02	0.882 ± 0.03	**0.856 ± 0.02**	$(2.9\pm2.7)\times10^{23}$	$(7.1\pm6.6)\times10^{24}$
	336	1.661 ± 0.11	0.961 ± 0.04	0.902 ± 0.02	0.904 ± 0.03	**0.878 ± 0.02**	$(5.8\pm4.1)\times10^{19}$	$(1.4\pm1.3)\times10^{23}$
	720	1.976 ± 0.11	0.870 ± 0.03	0.857 ± 0.02	0.863 ± 0.03	**0.854 ± 0.02**	$(2.1\pm1.5)\times10^{19}$	×
	Avg	1.926 ± 0.12	0.922 ± 0.03	0.879 ± 0.02	0.886 ± 0.03	**0.866 ± 0.02**	$(1.5\pm1.3)\times10^{23}$	$(3.8\pm3.6)\times10^{30}$
Exchange rate	96	18.757 ± 0.68	$(1.7\pm0.1)\times10^{2}$	$(6.2\pm0.5)\times10^{2}$	10.912 ± 0.33	**9.705 ± 0.32**	$(2.5\pm0.2)\times10^{19}$	$(1.5\pm0.2)\times10^{13}$
	192	40.131 ± 1.35	$(1.7\pm0.1)\times10^{2}$	$(1.6\pm0.1)\times10^{3}$	17.519 ± 0.73	**16.568 ± 0.63**	$(2.3\pm0.2)\times10^{14}$	×
	336	43.942 ± 1.04	$(2.4\pm0.1)\times10^{2}$	$(1.9\pm0.1)\times10^{3}$	27.064 ± 0.65	**25.050 ± 0.56**	$(5.9\pm1.3)\times10^{20}$	$(1.1\pm0.2)\times10^{21}$
	720	36.109 ± 0.78	$(1.3\pm0.0)\times10^{2}$	$(8.6\pm0.5)\times10^{2}$	36.573 ± 0.61	**35.816 ± 0.72**	$(4.3\pm1.1)\times10^{21}$	$(2.8\pm0.8)\times10^{32}$
	Avg	34.735 ± 0.96	$(1.8\pm0.1)\times10^{2}$	$(1.2\pm0.1)\times10^{3}$	23.017 ± 0.58	**21.785 ± 0.56**	$(1.2\pm0.3)\times10^{21}$	$(9.4\pm2.8)\times10^{31}$
Electricity	96	$(5.2\pm0.2)\times10^{2}$	$\mathbf{(3.1\pm0.1)\times10^{2}}$	$(1.4\pm0.1)\times10^{3}$	$(3.3\pm0.1)\times10^{2}$	$(3.3\pm0.1)\times10^{2}$	$(6.5\pm5.4)\times10^{14}$	$(5.7\pm4.3)\times10^{19}$
	192	$(5.6\pm0.2)\times10^{2}$	$\mathbf{(3.6\pm0.1)\times10^{2}}$	$(1.3\pm0.0)\times10^{3}$	$(4.2\pm0.1)\times10^{2}$	$(4.2\pm0.1)\times10^{2}$	$(1.1\pm0.3)\times10^{13}$	$(1.4\pm1.3)\times10^{22}$
	336	$(5.8\pm0.2)\times10^{2}$	$\mathbf{(3.6\pm0.1)\times10^{2}}$	$(1.3\pm0.0)\times10^{3}$	$(4.0\pm0.2)\times10^{2}$	$(4.0\pm0.2)\times10^{2}$	$(4.3\pm1.5)\times10^{10}$	$(4.7\pm1.4)\times10^{12}$
	720	$(7.0\pm0.2)\times10^{2}$	$\mathbf{(4.6\pm0.1)\times10^{2}}$	$(1.9\pm0.0)\times10^{3}$	$(5.7\pm0.1)\times10^{2}$	$(5.7\pm0.1)\times10^{2}$	$(1.6\pm0.2)\times10^{5}$	$(3.5\pm2.2)\times10^{23}$
	Avg	$(5.9\pm0.2)\times10^{2}$	$\mathbf{(3.7\pm0.1)\times10^{2}}$	$(1.5\pm0.0)\times10^{3}$	$(4.3\pm0.1)\times10^{2}$	$(4.3\pm0.1)\times10^{2}$	$(1.7\pm1.4)\times10^{14}$	$(9.2\pm5.7)\times10^{22}$
Traffic	96	$\mathbf{(1.9\pm0.1)\times10^{3}}$	$(2.5\pm0.1)\times10^{3}$	$(6.3\pm0.6)\times10^{4}$	$(2.9\pm0.1)\times10^{3}$	$(2.9\pm0.1)\times10^{3}$	$(9.7\pm7.1)\times10^{13}$	×
	192	$\mathbf{(2.2\pm0.1)\times10^{3}}$	$(3.0\pm0.2)\times10^{3}$	$(6.8\pm0.8)\times10^{4}$	$(4.3\pm0.2)\times10^{3}$	$(4.3\pm0.2)\times10^{3}$	$(1.4\pm1.1)\times10^{21}$	×
	336	$\mathbf{(2.1\pm0.1)\times10^{3}}$	$(2.7\pm0.2)\times10^{3}$	$(5.1\pm0.5)\times10^{4}$	$(3.4\pm0.2)\times10^{3}$	$(3.4\pm0.2)\times10^{3}$	$(5.5\pm2.2)\times10^{14}$	$(9.9\pm4.2)\times10^{21}$
	720	$\mathbf{(2.3\pm0.1)\times10^{3}}$	$(3.2\pm0.2)\times10^{3}$	$(8.0\pm0.9)\times10^{4}$	$(4.8\pm0.2)\times10^{3}$	$(4.8\pm0.2)\times10^{3}$	$(2.9\pm0.9)\times10^{12}$	×
	Avg	$\mathbf{(2.2\pm0.1)\times10^{3}}$	$(2.9\pm0.2)\times10^{3}$	$(6.6\pm0.7)\times10^{4}$	$(3.8\pm0.2)\times10^{3}$	$(3.8\pm0.2)\times10^{3}$	$(3.5\pm2.6)\times10^{20}$	$(9.9\pm4.2)\times10^{21}$

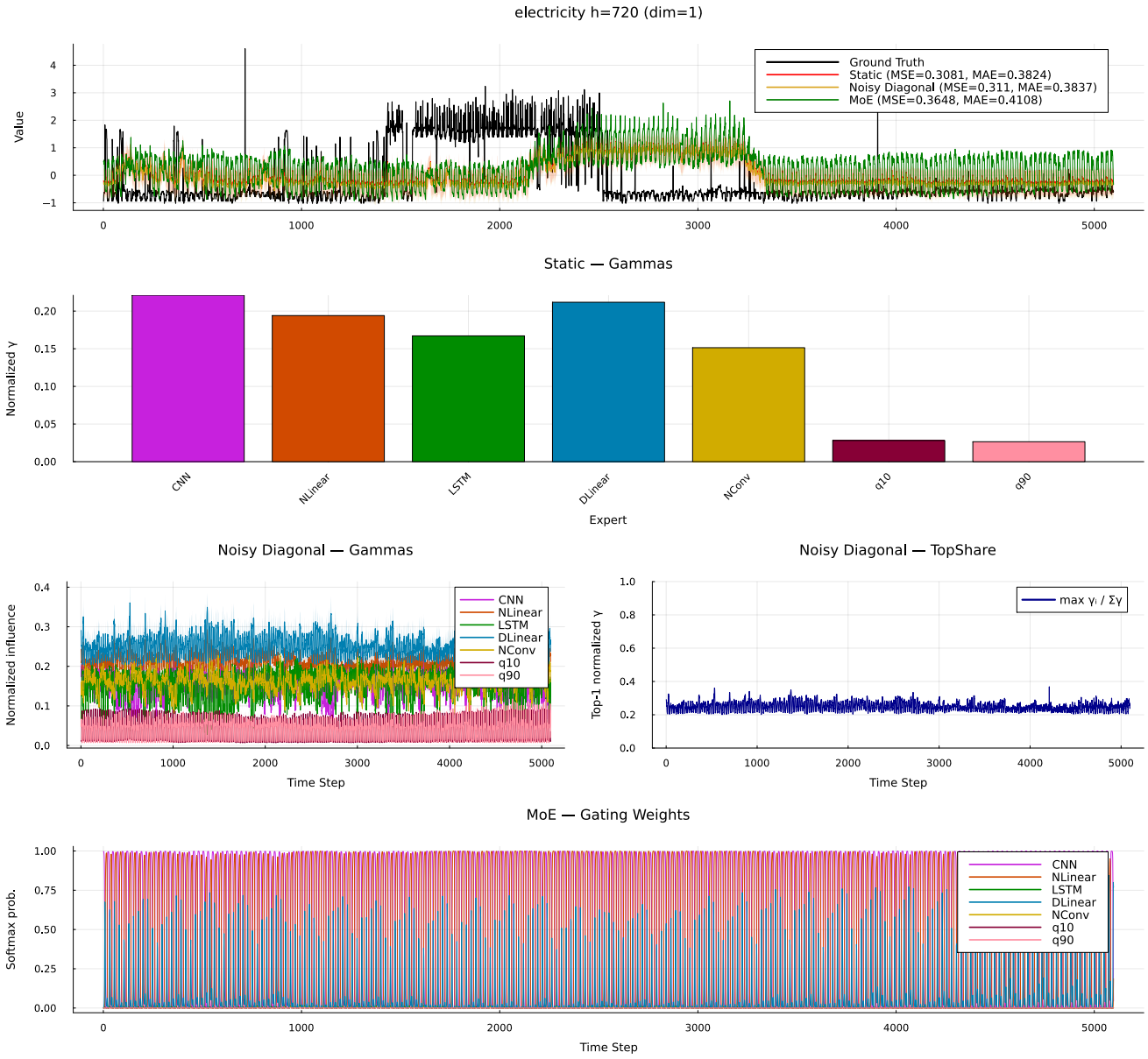


Figure 10: Comparison of forecasting with confidence interval of Static, Noisy Diagonal and MoE ensembles on electricity dataset and horizon 720. With values of each ensemble method for it's experts.

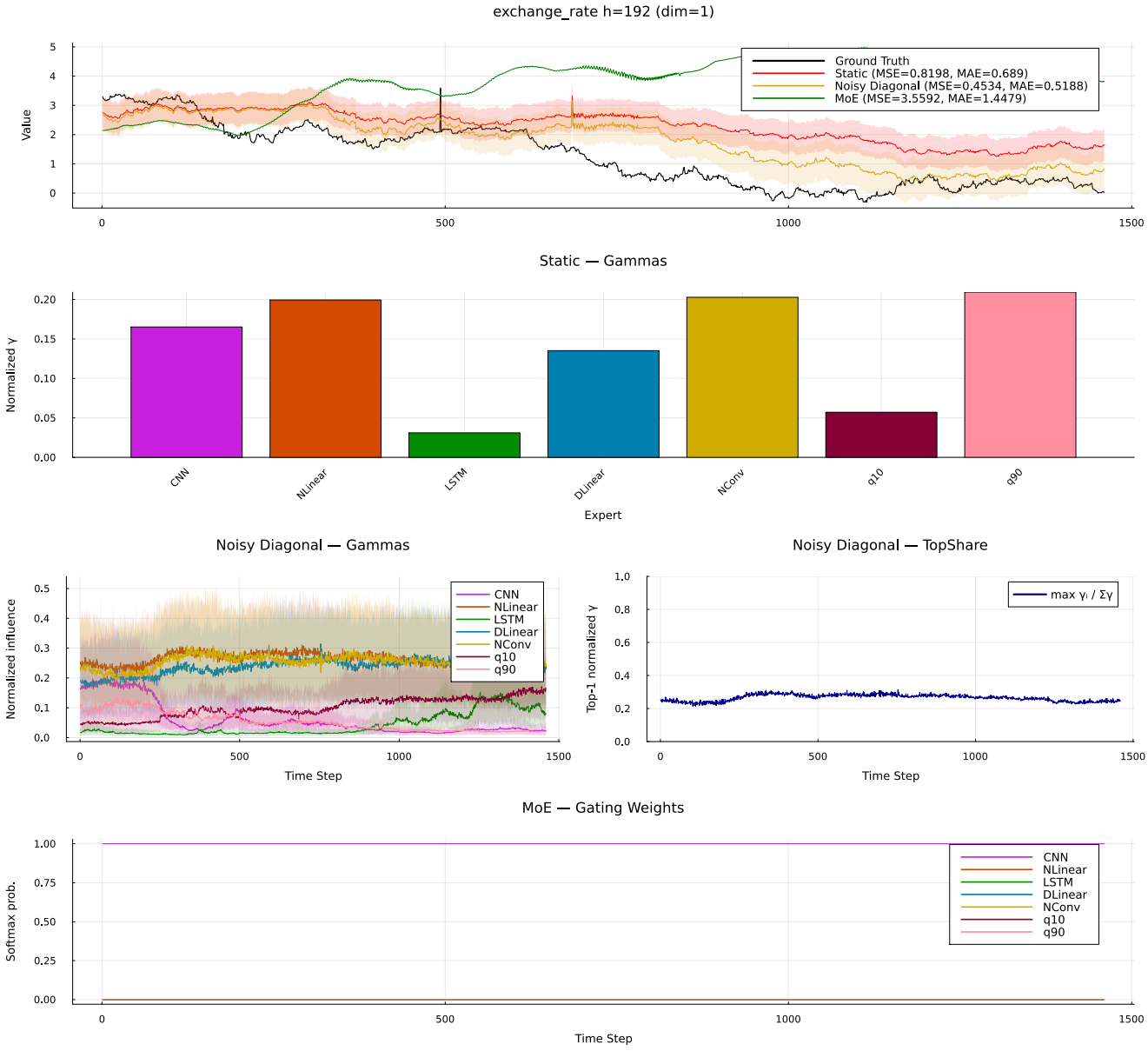


Figure 11: Comparison of forecasting with confidence interval of Static, Noisy Diagonal and MoE ensembles on exchange rate dataset and horizon 192. This example shows how classic neural network can collapse into one expert, which is not the best for this task.