The generalized Hierarchical Gaussian Filter

Lilian Aline Weber[1,2*], Peter Thestrup Waade[3], Nicolas Legrand[3], Anna Hedvig Møller[3], Klaas Enno Stephan[2,4], Christoph Mathys[2,5,6]

1 Wellcome Centre for Integrative Neuroscience, Department of Psychiatry, University of Oxford, Oxford, United Kingdom
2 Translational Neuromodeling Unit, Institute for Biomedical Engineering, University of Zurich & ETH Zurich, Zurich, Switzerland
3 Cognitive Science Department, Aarhus University, Denmark
4 Max Planck Institute for Metabolism Research, Cologne, Germany
5 Scuola Internazionale Superiore di Studi Avanzati (SISSA), Trieste, Italy
6 Interacting Minds Centre, Aarhus University, Aarhus, Denmark

* lilian.weber@psych.ox.ac.uk

Abstract

Hierarchical Bayesian models of perception and learning feature prominently in contemporary cognitive neuroscience where, for example, they inform computational concepts of mental disorders. This includes predictive coding and hierarchical Gaussian filtering (HGF), which differ in the nature of hierarchical representations. Predictive coding assumes that higher levels in a given hierarchy influence the state (value) of lower levels. In HGF, however, higher levels determine the rate of change at lower levels. Here, we extend the space of generative models underlying HGF to include a form of nonlinear hierarchical coupling between state values akin to predictive coding and artificial neural networks in general. We derive the update equations corresponding to this generalization of HGF and conceptualize them as connecting a network of (belief) nodes where parent nodes either predict the state of child nodes or their rate of change. This enables us to *(1)* create modular architectures with generic computational steps in each node of the network, and *(2)* disclose the hierarchical message passing implied by generalized HGF models and to compare this to comparable schemes under predictive coding. We find that the algorithmic architecture instantiated by the generalized HGF is largely compatible with that of predictive coding but extends it with some unique predictions which arise from precision and volatility related computations. Our developments enable highly flexible implementations of hierarchical Bayesian models for empirical data analysis and are available as open source software.

Keywords: hierarchical Gaussian filter, HGF, predictive coding, perceptual inference, neuromodeling, computational psychiatry

1 Introduction

1.1 Overview

To successfully navigate complex environments, biological agents need to combine noisy sensory inputs with prior knowledge to infer on the true, but hidden state of the world. Hierarchical Bayesian models such as predictive coding (Rao & Ballard, 1999; Friston, 2005) have become popular tools to understand brain functions which enable humans to form beliefs based on sparse and ambiguous sensory inputs. These models assume that the brain constructs a hierarchy of beliefs, with higher level beliefs relating to increasingly abstract features of the world. For example, hierarchical Gaussian filtering (HGF, Mathys et al., 2011, 2014) models a hierarchy in which higher levels encode the rate of change in lower level features, enabling agents to adaptively scale their learning behaviour in response to changes in the stability of their environment. Here, we extend this framework to a different type of hierarchy, where higher-level beliefs influence lower-level beliefs via their expectation which correspond to the mean in the Gaussian case. In other words, higher-level beliefs can here also determine the *value* of lower-level beliefs instead of only their speed of change. This extension significantly widens hierarchical Gaussian filtering's application scope and enables a direct comparison of the implied message-passing scheme with, for example, predictive coding models. Moreover, the new derivations endow HGF with a modular architecture which allows for more versatile and user-friendly implementations of HGF models.

1.2 Hierarchical Bayesian models of perception and learning

Bayesian perspectives on perception have proposed that our brain inverts a hierarchical generative model to infer the causes of its sensory inputs and predict future events (Dayan et al., 1995; Rao & Ballard, 1999; Friston, 2010; Doya et al., 2011; Helmholtz, 1860). Under this ≪Bayesian brain≫ view, perception corresponds to integrating expectations (prior beliefs about hidden states of the world) with incoming sensory information to yield a posterior belief, while *learning* refers to the updating of beliefs about the model's parameters, which takes place more slowly, as experience accumulates). Formally, beliefs are modelled as probability distributions, such that the width of the distribution reflects the uncertainty (inverse precision) associated with that belief. Humans have been shown to take uncertainty into account when combining different sources of information, in a manner that conforms to the statistical optimum as prescribed by Bayes' rule (Ernst & Banks, 2002; Angelaki et al., 2009).

Because natural sensory signals are generated by interacting causes in the external environment that span multiple spatial and temporal scales, the brain in its Helmholtzian description is assumed to reflect this hierarchy of causes in a correspondingly hierarchical genereative model. The Bayesian inversion of this generative model results in a hierarchy of beliefs, where higher levels encode beliefs about increasingly abstract, general, and stable features of the environment. These higher-level beliefs serve as priors for the inference on lower levels. Specifically, at each level of the hierarchy, belief updates serve to reconcile predictions (priors) from higher levels with the actual input (likelihood) from lower levels. Furthermore, under fairly general assumptions (i.e., for all probability distributions from the exponential family, Mathys, 2016; Mathys & Weber, 2020), these belief updates rest on (precision-weighted) prediction errors (PEs), i.e., the (weighted) mismatch between the model's predictions and the actual input (Friston, 2010).

Both predictive coding (Rao & Ballard, 1999; Friston, 2005) and hierarchical Gaussian filtering (Mathys et al., 2011, 2014) have proved useful in the modelling of the inferential hierarchies of the brain and mind. They

Popular hierarchical Bayesian models of perception and learning that are built on these ideas are predictive coding (Rao & Ballard, 1999; Friston, 2005) and hierarchical Gaussian filtering (Mathys et al., 2011, 2014). In these models, subjective estimates of uncertainty take center stage: the impact of prediction errors on belief updates depends on a precision ratio, which relates the precision of the prior to that of the observation, thus scaling the relative impact that new information has on belief updates. Put simply, mismatches (PEs) elicit stronger belief updates if the prediction about the input (likelihood) is precise, relative to the belief in the current estimate (prior). This form of adaptive scaling, a key element of healthy inference and learning, has also been proposed to lie at the heart of perceptual disturbances observed in mental disorders. For example, an imbalance between the influence of expectations and sensory inputs has featured prominently in attempts to explain the emergence of positive symptoms in schizophrenia, such as hallucinations and delusions (Stephan et al., 2006; Fletcher & Frith, 2009; Corlett et al., 2009, 2011; Adams et al., 2013; Friston et al., 2016; Sterzer et al., 2018).

The HGF has been particularly useful in this context, as it can be fit to participants' empirically observed behaviour or physiology, and thereby used to infer individual trajectories of trial-wise precision-weighted PEs and predictions from measured data. By formulating a response model that links trial-wise perceptual quantities (such as predictions and PEs) to measured quantities (such as choices, reaction times, eye movements, evoked response amplitude in EEG, etc.), the HGF can quantify individual differences in inference and learning in terms of model parameters that encode prior beliefs about higher-order structure in the environment (de Berker et al., 2016; Lawson et al., 2017; Powers et al., 2017; Siegel et al., 2020; Sevgi et al., 2020; Henco et al., 2020; Rossi-Goldthorpe et al., 2021; Suthaharan et al., 2021; Kafadar et al., 2022; Sapey-Triomphe et al., 2022; Fromm et al., 2023; Drusko et al., 2023). Such a mechanistic characterization of inter-subject variability is of particular interest for fields like Computational Psychiatry, because such differences may explain the heterogeneous nature of psychiatric diseases, and form a basis for dissecting them into more homogeneous subgroups (Stephan & Mathys, 2014; Mathys, 2016).

1.3 Volatility-coupling, value-coupling, and noise-coupling

The type of belief hierarchy modelled by any particular approach depends on the nature of the generative model. The HGF assumes a particular form of generative model, where hidden states of the world evolve as coupled Gaussian random walks in time. In current HGF models, the mean (state) of the higher level determines the variance (step size) of the lower level's random walk. In other words, higher levels encode the volatility (or inverse stability) of lower levels (we will call this volatility coupling). This is motivated by the observation that learners must take into account different sources of uncertainty in their belief updates, one of which is the current level of stability in the world: if the world is currently changing (volatile), the agent needs to learn faster. Accordingly, in the HGF, subjective estimates of increased environmental volatility directly influence the uncertainty associated with lower level beliefs, leading to faster belief updates on the lower level. Previous work has shown that human learners indeed adjust their learning rate according to experimentally manipulated levels of volatility (Behrens et al., 2007).

By contrast, predictive coding models typically focus on hierarchies in which higher levels predict the *value* of lower levels, or, in other words, the mean of the probability distribution that represents the lower-level belief. We will refer to these hierarchies as implementing value coupling. This type of hierarchy is useful for understanding how beliefs about lower-level features depend on higher-level beliefs – for example, how the perceived brightness of a patch in an image depends on the context (objects, shadows) in which that patch is presented (Rao & Ballard, 1999; Adelson, 2005). It is worth noting that the type of hierarchical coupling in predictive coding is more frequently used in theoretical treatments of hierarchical Bayesian modelling while the HGF offers a flexibly applicable implementation that is being widely used for empirical data analysis.

In a noteworthy exception to this, Kanai et al. (2015) presented a predictive coding model where higher levels encode the (spatial) precision of lower levels in a static environment. This is different from volatility coupling, where higher levels are concerned with the rate of change on lower levels, but captures a second source of uncertainty in beliefs about hidden states: the level of noise or reliability of the sensory input.

Relatedly, in the learning and decision-making literature, two classes of models separately deal with the estimation of process noise (volatility, Behrens et al., 2007; Mathys et al., 2011; Piray & Daw, 2020) and observation noise (stochasticity, Lee et al., 2020; Nassar et al., 2010), with recent attempts to capture both sources of uncertainty in a joint model (Piray & Daw, 2021).

In this work, we show that the HGF can be extended to encompass both volatility and value coupling, and to include a principled way of modelling inference on noise at the level of observations – thus providing a very general modelling framework with a wide range of potential applications.

1.4 Contribution of current work

In this technical note, we extend the generative model of the HGF to consider hierarchical value coupling alongside volatility coupling. Based on the work in Weber (2020), we *(1)* derive simple, efficient one-step belief update equations for linear and non-linear value coupling, and *(2)* conceptualize these equations, together with their volatility coupling counterparts, as a network of interacting nodes which can be implemented in a modular architecture.

In brief, we show that:

1. The HGF provides a very general modelling framework that encompasses multiple types of interactions between states in the world - where higher-level hidden states determine the value, the rate of change (volatility), or even the level of noise in lower-level states or observations.

2. The explicit treatment of volatility estimation in the HGF allows for an implementation that comprises both global control of volatility-related precision (implemented by a global high-level volatility belief that affects multiple low-level states), and local or distributed volatility estimation, enabling modality-specific modulation of learning rates.

3. The message passing scheme for value coupling in the HGF is almost equivalent to recently proposed predictive coding architectures, apart from small, but interesting differences. These differences relate on the one hand to the discrete nature of the updates in the HGF, and, on the other hand, to the volatility-related updates of belief uncertainty.

2 Introducing value coupling

The generative model for value coupling

The HGF assumes that an agent is trying to infer on (and learn about) a continuous uncertain quantity x in their environment, which moves (changes) over time. Without any information about the specific form of movement, a generic way of describing movement of a continuous quantity is a Gaussian random walk:

$$x^{(k)} \sim \mathcal{N}\left(x^{(k-1)},\, \vartheta\right). \tag{1}$$

In the original formulation (Mathys et al., 2011, 2014), coupling between environmental states at different hierarchical levels was implemented in the form of volatility coupling, where the step size ϑ (or rate of change/volatility) of a state x_a varies a function of a higher-level state $\check{x}_a$:

$$\vartheta = f\left(\check{x}_a^{(k)})\right) = \exp\left(\kappa_{\check{a},a}\,\check{x}_a^{(k)} + \omega_a\right), \tag{2}$$

with parameters $\kappa_{\check{a},a}$ (scaling the impact of volatility parent $\check{x}_a$ on x_a) and ω_a (capturing the "tonic" step size or volatility, which does not vary with time). By simultaneously inferring on the state x_a and its rate of change $\check{x}_a$, the agent can learn faster (slower) in times when x_a is changing more (less). We call $\check{x}_a$ a volatility parent of x_a.

In contrast, here, we consider the case where a higher-level state x_b influences the value (mean) of the lower-level state x_a:

$$x_a^{(k)} \sim \mathcal{N}\left(x_a^{(k-1)} + f\left(x_b^{(k)}\right),\, \vartheta\right), \tag{3}$$

where

$$f\left(x_b^{(k)}\right) = \alpha_{b,a}\, g\left(x_b^{(k)}\right) \tag{4}$$

is the coupling function with parameter $\alpha_{b,a}$ (scaling the impact of value parent x_b on x_a). Crucially, as we will show below, the HGF can deal with both linear and nonlinear transformation functions g, as long as the function g is twice differentiable. States x_b and x_a interact via value coupling.

Importantly, the two forms of coupling can be present at the same time: A state x_a can have both a volatility parent $\check{x}_a$ (generating changes in its rate of change) and a value parent x_b (generating changes in its mean value). It can also have a drift parameter ρ_a which is a constant influencing its mean – equivalent to its tonic volatility parameter ω_a which determines its step size in the absence of a phasic volatility influence. Finally, we allow for inputs to arrive at irregular intervals; therefore, we multiply the total variance of the random walk and the total mean drift by the time $t^{(k)}$ that has passed since the arrival of the previous sensory input at index $k-1$ (Mathys et al., 2014). Together with suitably chosen priors on parameters and initial states (see Mathys et al., 2014), the following equation forms the generative model for a state x_a with both volatility and value parent:

$$x_a^{(k)} \sim \mathcal{N}\left(x_a^{(k-1)} + t^{(k)}\left(\rho_a + \alpha_{b,a} g\left(x_b^{(k)}\right)\right),\, t^{(k)}\exp(\omega_a) + \kappa_{\check{a},a}\check{x}_a^{(k)}\right). \tag{5}$$

In the even more general case, a state could have multiple value parents and multiple volatility parents, each impacting the mean value and rate of change of state x_a in proportion to their respective coupling strengths[1]:

[1] We are here only considering the additive effect of multiple parents on a given state, but more sophisticated interactions are conceivable.

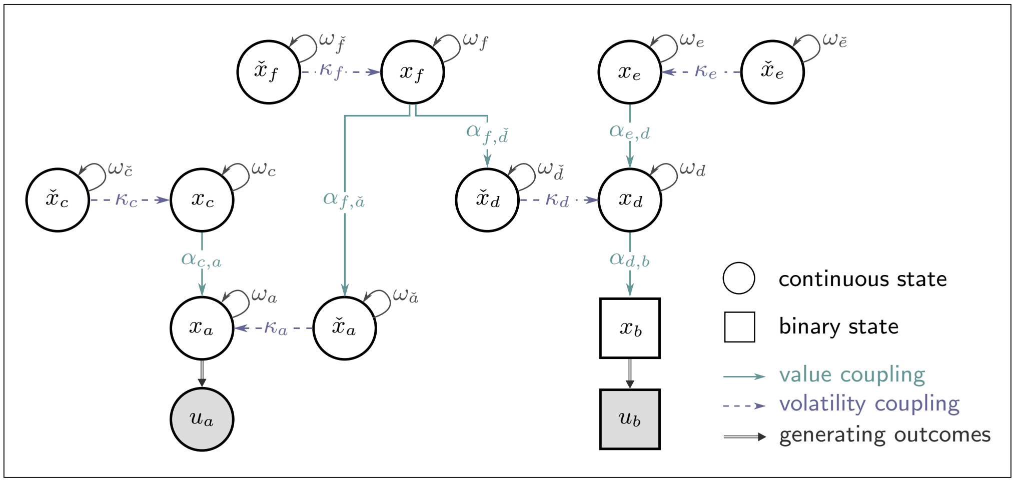


Figure 1. An example of a generative model of sensory inputs with 11 hidden states and two observable outcomes. In this example, the volatility parents $\check{x}_a$ and $\check{x}_d$ share a value parent x_f, which represents a "global" or shared volatility state. Circles – continuous states, squares – binary states, observable outcomes – shaded. Volatility coupling – dashed lines, value coupling – straight lines, links of outcomes to their hidden states – double arrows.

$$x_a^{(k)} \sim \mathcal{N}\left(x_a^{(k-1)} + t^{(k)}\left(\rho_a + \sum_i \alpha_{b_i,a} g\left(x_{b_i}^{(k)}\right)\right), t^{(k)} \exp(\omega_a) + \sum_j \kappa_{\check{a}_j,a} \check{x}_{a_j}^{(k)}\right). \quad (6)$$

Because a given state can also be parent to multiple child states at the same time, these extensions allow us to model fairly complex networks of interacting states of the world.

In Figure 1 we have drawn an example setup with 11 different environmental states and two outcomes. For this example, and together with priors on parameters and initial states (see Mathys et al., 2014), the following equations describe the generative model (for simplicity, we have restricted the example to linear value coupling, no drifts ($\rho = 0$), and inputs at regular intervals, i.e., $t^{(k)} = 1$):

$$u_a^{(k)} \sim \mathcal{N}\left(x_a^{(k)}, \zeta_u\right) \tag{7}$$

$$x_a^{(k)} \sim \mathcal{N}\left(x_a^{(k-1)} + \alpha_{c,a} x_c^{(k)}, \exp\left(\kappa_a \check{x}_a^{(k)} + \omega_a\right)\right) \tag{8}$$

$$\check{x}_a^{(k)} \sim \mathcal{N}\left(\check{x}_a^{(k-1)} + \alpha_{f,\check{a}} x_f^{(k)}, \exp(\omega_{\check{a}})\right) \tag{9}$$

$$x_c^{(k)} \sim \mathcal{N}\left(x_c^{(k-1)}, \exp\left(\kappa_c \check{x}_c^{(k)} + \omega_c\right)\right) \tag{10}$$

$$\check{x}_c^{(k)} \sim \mathcal{N}\left(\check{x}_c^{(k-1)}, \exp\left(\omega_{\check{c}}\right)\right) \tag{11}$$

$$u_b^{(k)} \sim \mathrm{Bern}\left(x_b^{(k)}\right) \tag{12}$$

$$x_b^{(k)} \sim \mathrm{Bern}\left(S\left(x_d^{(k)}\right)\right) \tag{13}$$

$$x_d^{(k)} \sim \mathcal{N}\left(x_d^{(k-1)} + \alpha_{e,d} x_e^{(k)}, \exp\left(\kappa_d \check{x}_d^{(k)} + \omega_d\right)\right) \tag{14}$$

$$\check{x}_d^{(k)} \sim \mathcal{N}\left(\check{x}_d^{(k-1)} + \alpha_{d,f} x_f^{(k)}, \exp(\omega_{\check{d}})\right) \tag{15}$$

$$x_e^{(k)} \sim \mathcal{N}\left(x_e^{(k-1)}, \exp\left(\kappa_e \check{x}_e^{(k)} + \omega_e\right)\right) \tag{16}$$

$$\check{x}_e^{(k)} \sim \mathcal{N}\left(\check{x}_e^{(k-1)}, \exp(\omega_{\check{e}})\right) \tag{17}$$

$$x_f^{(k)} \sim \mathcal{N}\left(x_f^{(k-1)}, \exp\left(\kappa_f \check{x}_f^{(k)} + \omega_f\right)\right) \tag{18}$$

$$\check{x}_f^{(k)} \sim \mathcal{N}\left(\check{x}_f^{(k-1)}, \exp(\omega_{\check{f}})\right). \tag{19}$$

Note that in this example, we introduce two general motifs. First, all states that are value parents of other states (or outcomes) by default have their own volatility parent (and volatility parents therefore share the index with their child node, for example, states x_a and $\check{x}_a$). Even if in practice, many environmental states might have constant volatility, from the perspective of the agent, it makes sense to a-priori allow for phasic changes in volatility. From a modelling perspective, these volatility parents could be removed in scenarios with constant volatility.

Second, states that are volatility parents to other states can either have a value parent (as states $\check{x}_a$ and $\check{x}_d$), or no parents (as states $\check{x}_c$, $\check{x}_e$ and $\check{x}_f$). This is because in practice, volatility parents of volatility parents are rarely required. Instead, we suggest that value parents of volatility states are more useful. In particular, these can be used to model "global" or shared volatility states that affect multiple lower-level volatility beliefs (such as state x_f in this example, which influences volatility beliefs about states x_a and x_d), separately from "local" volatility states that only affect speed of change in a single lower-level state (such as the volatility states $\check{x}_c$, $\check{x}_e$ and $\check{x}_f$).

In summary, we have introduced value coupling to the generative model of the HGF, for the first time considering the case where the mean of x_a is a function not only of its own previous value but also (some transformation of) the current value of some higher-level state x_b, scaled by a coupling parameter $\alpha_{b,a}$. We will now show how an agent can infer on the values of such hidden states.

The belief update equations for value coupling

An agent employing a generative model of the kind described above to do perceptual inference holds a belief about the current value of each of the states (i.e., every x) of this model on every trial (time point). We describe this belief about state x on trial k

as a Gaussian distribution, fully characterized by its mean $\mu^{(k)}$ and its inverse variance, or precision, $\pi^{(k)}$ on a given trial k.

In the approximate inversion of the generative model for volatility coupling, Mathys et al. (2011) derived a set of simple, one-step update equations that represent the approximately Bayes-optimal trialwise changes in these beliefs in response to incoming stimuli. Repeating this derivation for the case of value coupling similarly leads us to simple one-step equations for updating beliefs about states that serve as value parents (for the full derivation of these equations, please see Appendix 5.1). Assuming state x_b is a value parent to state x_a with a coupling strength $\alpha_{b,a}$ (i.e., the coupling function defined by equation 4), then the new posterior belief about state x_b after observing a new input at trial k is given by:

$$\begin{aligned}\pi_b^{(k)} &= \hat{\pi}_b^{(k)} + \hat{\pi}_a^{(k)} \left(\alpha_{b,a}^2 g' \left(\mu_b^{(k-1)} \right)^2 - g'' \left(\mu_b^{(k-1)} \right) \delta_a^{(k)} \right) \\ \mu_b^{(k)} &= \hat{\mu}_b^{(k)} + \frac{\hat{\pi}_a^{(k)} \alpha_{b,a} g' \left(\mu_b^{(k-1)} \right)}{\pi_b^{(k)}} \delta_a^{(k)},\end{aligned} \tag{20}$$

where $\hat{\pi}_b^{(k)}$ and $\hat{\mu}_b^{(k)}$ refer to the prediction about state x_b before seeing the new input, and $\delta_a^{(k)}$ is the prediction error about the child state x_a. Note that in the case of linear value coupling ($g(x) = Ax + B$), the update further simplifies, as $g'(x) = A$ (a factor that we can absorb into $\alpha_{b,a}$) and $g''(x) = 0$:

$$\begin{aligned}\pi_b^{(k)} &= \hat{\pi}_b^{(k)} + \alpha_{b,a}^2 \hat{\pi}_a^{(k)} \delta_a^{(k)} \\ \mu_b^{(k)} &= \hat{\mu}_b^{(k)} + \frac{\alpha_{b,a} \hat{\pi}_a^{(k)}}{\pi_b^{(k)}} \delta_a^{(k)}.\end{aligned} \tag{21}$$

As in the case of volatility coupling, the belief updates are thus driven by precision-weighted prediction errors about the lower belief state in the hierarchy.

To get an intuition for these update equations in the case of nonlinear value coupling, let us consider an example where state x_a acts as a rectified linear unit (ReLU), that is, we choose the function g as the rectifier function, a popular activation function for deep neural networks (Hahnloser et al., 2000; Lecun et al., 2015):

$$g(x_b) := \max(0, x_b). \tag{22}$$

The first and second derivative of g are then

$$g'(x_b) = [x_b > 0] =: \begin{cases} 1, & \text{if } x_b > 0 \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

and

$$g''(x_b) = \delta(x_b = 0), \tag{24}$$

respectively. For our purposes, we can treat the second derivative as

$$g''(x_b) = 0. \tag{25}$$

Plugging this into equation 20, we get

$$\pi_b^{(k)} = \hat{\pi}_b^{(k)} + \alpha_{b,a}^2 \hat{\pi}_a^{(k)} \left[\mu_b^{(k-1)} > 0 \right] \tag{26}$$

and

$$\mu_b^{(k)} = \hat{\mu}_b^{(k)} + \frac{\alpha_{b,a}\hat{\pi}_a^{(k)}\left[\mu_b^{(k-1)} > 0\right]}{\pi_b^{(k)}}\delta_a^{(k)}. \tag{27}$$

In other words, the impact of lower-level prediction errors $\delta_a^{(k)}$ on the posterior belief at the higher level $\mu_b^{(k)}$ and $\pi_b^{(k)}$ depends on the previous state of the higher-level node, such that beliefs only change in response to inputs if they were above zero (or 'active') in the first place.

Critically, this extension of the HGF now allows us to build deep networks with non-linear coupling between the levels.

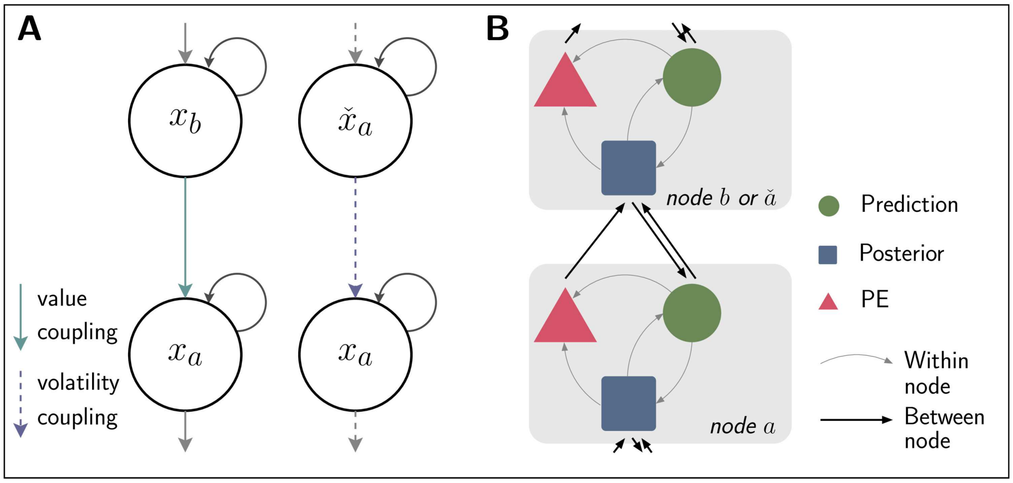


Figure 2. Comparing the flow of information in the generative model of the HGF with the implied belief network. **A** In the generative model, higher-level states influence the evolution of lower-level states (top-down information flow), either by impacting on their mean (value coupling, left) or by changing their evolution rate (volatility coupling, right). **B** Representation of the message-passing within and between belief nodes as implied by the HGF's belief update equations. New observations cause a cascade of message-passing between nodes that includes bottom-up and top-down information flow. Higher-level beliefs send down their posteriors to inform lower-level predictions. Lower-level belief nodes send prediction errors and the precision of their own prediction bottom-up to drive higher-level belief updating. Within a node, we have placed separate units for the three computational steps that each node has to perform on a given trial: the prediction step (green), the update step which results in a new posterior belief (blue), and the prediction error step (red). This message passing scheme generalizes across value and volatility coupling, although the specific messages passed along the connections as well as the computations within the nodes will depend on coupling type (see main text and Figures 3 and 4 for details).

3 Belief updates in the HGF: A network of nodes

Just like we can model complex networks of interacting states in the environment using the generative model of the HGF (Figure 1), we can also think of the inference process of an agent in that environment as a network of interdependent beliefs. The agent entertains a belief about each of the relevant environmental states, and updates these beliefs based on new sensory inputs. Because the agent models her world as a set of hierarchically interacting states, her beliefs about these states will form a hierarchy as well (Figure 2). Before new inputs arrive, higher-level beliefs inform predictions about lower-level beliefs (top-down information flow), whereas after the arrival of a new piece of information, changes in lower-level beliefs trigger updates of higher-level beliefs (bottom-up information flow).

In the following, we conceptualize each belief modelled by the HGF as a node in a network, where belief updates involve computations within nodes as well as message passing between nodes. The specific within-node computations and messages passed between nodes will depend on the nature of the coupling between nodes. Putting the equations for value and volatility coupling side by side, we will reveal a modular architecture of this network which has important consequences for implementing the inference model (both in a computer and in a brain), and which we summarize in Figure 2B. We will also separately consider the cases of multiple parent or child nodes, and special nodes such as input nodes.

We start by noting that the computations of any node within an experimental trial can be subdivided into three steps, an update step, where a new posterior belief is computed based on a prediction and an incoming input or prediction error (PE), a PE step, where the difference between expectation (prediction) and new posterior is computed for further message passing upwards, and a prediction step, where the new posterior is used to predict the value on the next trial. These can be ordered in time as shown in the box:

```
NODE $a$ AT TRIAL $k$

  (re)compute prediction$_a^{(k)}$
← receive PE$_{child}^{(k)}$ from child node

UPDATE step
  compute posterior$_a^{(k)}$
  given: PE$_{child}^{(k)}$ and prediction$_a^{(k)}$
→ send posterior$_a^{(k)}$ to child node

PE step
  compute PE$_a^{(k)}$
  given: prediction$_a^{(k)}$ and posterior$_a^{(k)}$
→ send PE$_a^{(k)}$ to parent node
← receive posterior$_{parent}^{(k)}$ from parent node

PREDICTION step
  compute prediction$_a^{(k+1)}$
  given: posterior$_a^{(k)}$ and posterior$_{parent}^{(k)}$
```

Two things are worth noting here. First of all, the PE step is a computation that the node performs in service of its parents. From the perspective of the parent node b, the prediction$_a^{(k)}$ represents its expectation of the child node's state, and the posterior$_a^{(k)}$ corresponds to the actual state of this child on trial k. The difference between the two amounts to the prediction error which will serve to update the parent node - in other words, the parent's PE. In the case of multiple parent nodes, node a would thus compute multiple PEs, one for each of its parents. We will return to this point in the section 5.3 of the Appendix.

Second, we have placed the prediction step at the end of a trial. This is because usually, we think about the beginning of a trial as starting with receiving a new input, and of a prediction as being present before that input is received. However, in some cases the prediction also depends on the time that has passed in between trials (e.g., when considering drifts), which is something that can only be evaluated once the new input arrives - hence the additional computation of the (current) prediction at the beginning of the trial. Conceptually, it makes the most sense to think of the prediction as happening continuously between trials. For the implementation, it is however most convenient to only compute the prediction once the new input (and with it its arrival time) enters. This ensures both that the posterior means of parent nodes have had enough time to be sent back to their children for preparation for the new input, and that the arrival time of the new input can be taken into account appropriately.

A node of the above kind is the first computational subunit in our perceptual model, and it can be connected to other nodes via volatility or value coupling depending on the underlying generative model. For node a, another node b can function as a parent node, if the two are connected and node b represents a belief about a higher-level quantity which affects the belief about node a according to the generative model. On the other hand, if node b refers to a lower-level quantity and is connected to node a, it serves as a child node for node a.

Computations of nodes in the HGF

In the following, we examine the exact computations on trial k within a node for each of the three steps introduced above. We will compare the relevant computations between volatility and value coupling and identify the messages that have to be sent and received in each step. Since the update step relies on quantities computed in the (previous) prediction step, we here start with the computation of the predictions for the current trial (which, as explained above, we think of as being computed prior to the arrival of a new input). We will first only consider the case of linear value coupling (alongside volatility coupling), and then separately examine any differences in these computations for the case of nonlinear value coupling. The results of this analysis are summarized in Figures 2 – 4.

The prediction step

In the prediction step, node a prepares for receiving the new input. This entails computing a new prediction, based on the previously updated posterior beliefs. In general, the prediction of a new mean (or the new mean of the predictive distribution) will depend on whether node a has any value parents, whereas the precision of the new prediction will be influenced by the presence and posterior mean of the node's volatility parents.

In the general case, the mean and precision of the new prediction are computed as follows:

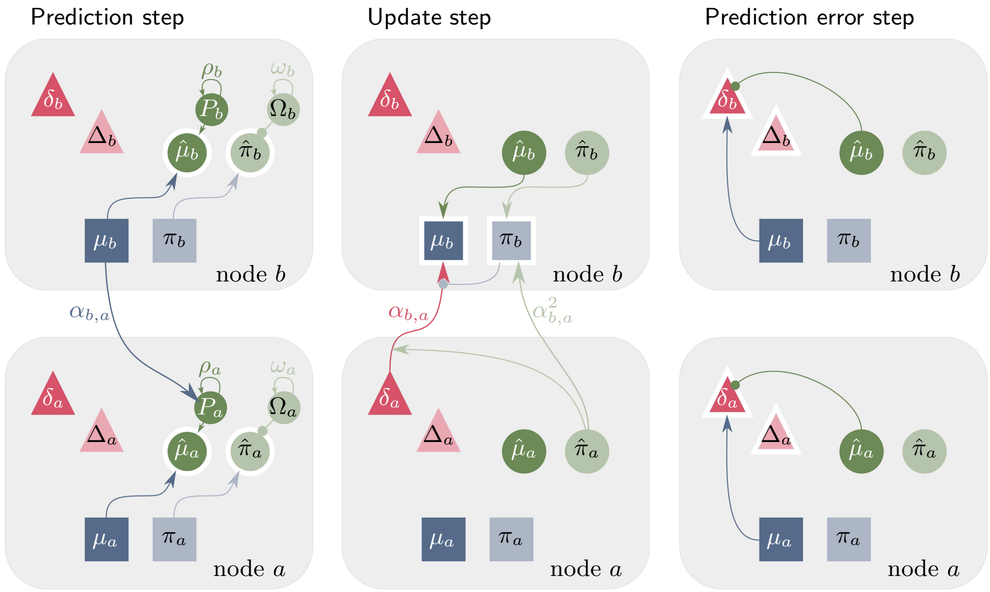
$$\hat{\mu}_a^{(k)} = \mu_a^{(k-1)} + P_a^{(k)} \tag{28}$$

$$\hat{\pi}_a^{(k)} = \frac{1}{\frac{1}{\pi_a^{(k-1)}} + \Omega_a^{(k)}}, \tag{29}$$

where $P_a^{(k)}$ is the total predicted drift of the mean on trial k, and $\Omega_a^{(k)}$ is the total predicted volatility (or change in step size) on trial k. Both of these are computed as a sum of a constant (or tonic) term, given by a model parameter, and a time-varying (or phasic) term which is driven by their parents:

The total predicted mean drift equals to the sum of constant term ρ_a, which is the tonic drift parameter of node a, and the sum of posterior means of all value parents $\mu_{b_i}^{(k-1)}$ (where a parent is indexed by i) on the previous trial $k-1$, weighted by their connection strengths $\alpha_{b_i,a}$:

$$P_a^{(k)} = t^{(k)} \left(\rho_a + \sum_{i=1}^{N_{vapa}} \alpha_{b_i,a} \mu_{b_i}^{(k-1)} \right), \tag{30}$$


Prediction step

Update step

Prediction error step

Figure 3. Message-passing for value coupling. Interactions of two nodes, node a and its value parent node b, are shown during the three steps of a trial (Prediction step, left; Update step, middle; Prediction error step, right). The quantities that are being computed in each step are highlighted in white. Note that for the Update step, we only show the computation of the posterior for the parent node b. Connections with arrowheads indicate positive (excitatory) influences, connections with circular heads indicate negative (inhibitory) influences. Arrows ending on units indicate additive influences, those ending on other arrows indicate multiplicative influences. Each HGF quantity that changes across trials is assigned its own unit. Parameters (α, κ, ω and ρ) determine connection strengths.

where $t^{(k)}$ denotes the time that has passed between trial $k-1$ and trial k, and N_{vapa} is the number of value parents. Similarly, the total predicted volatility $\Omega_a^{(k)}$ for the current trial is a function of a constant term ω_a (the tonic volatility parameter) and the posterior means of all volatility parents $\mu_{\check{a}_j}^{(k-1)}$ on the previous trial $k-1$, weighted by their connection strengths $\kappa_{\check{a}_j,a}$:

$$\Omega_a^{(k)} = t^{(k)} \exp\left(\omega_a + \sum_{j=1}^{N_{vopa}} \kappa_{\check{a}_j,a} \mu_{\check{a}_j}^{(k-1)}\right) \tag{31}$$

If node a does not have any parents, both the predicted drift P_a and the predicted volatility Ω_a are fully determined by constant parameters (ρ_a and ω_a) and the time between subsequent observations, and in the standard HGF without drift ($\rho_a = 0$), the predicted mean for the next trial is equal to the posterior mean of the current trial. Equations 28 to 31 nicely reflect the roles that value parents and volatility parents play in the generative model, where value parents model a phasic influence on a child node's mean, and volatility parents model a phasic influence on a child node's step size or volatility.

In sum, prediction step for node a only depends on knowing its own posterior belief from the previous trial and having received its parents' posteriors in time before the new input arrives. The implied message passing for this computational step is visualized in the left panel of Figure 3 for value coupling, and Figure 4 for volatility coupling.

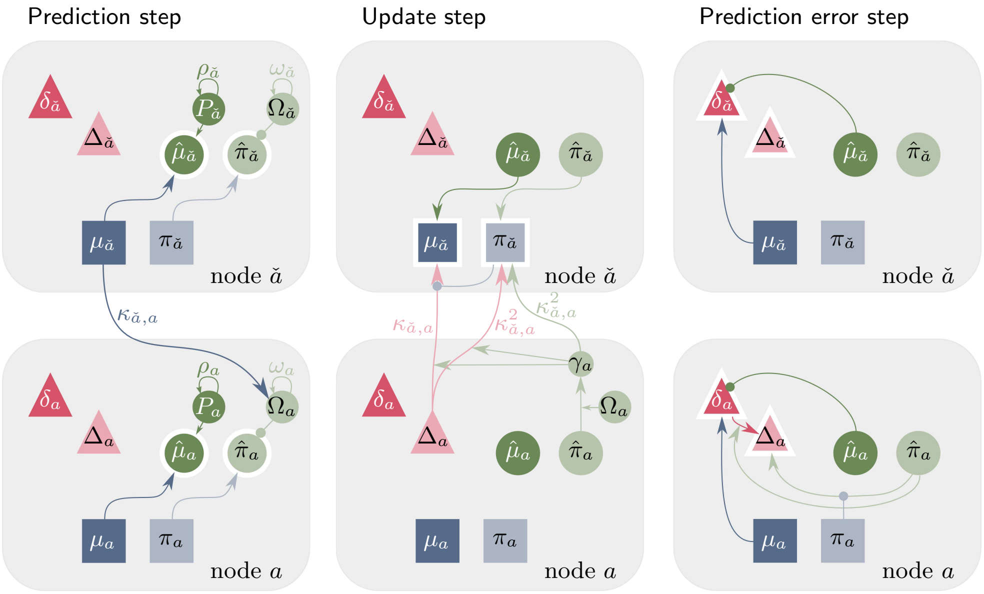


Figure 4. Message-passing for volatility coupling. Interactions of two nodes, node a and its volatility parent node $\breve{a}$, are shown during the three steps of a trial (Prediction step, left; Update step, middle; Prediction error step, right). The quantities that are being computed in each step are highlighted in white. Logic of display as in Figure 3.

The update step

The update step consists of computing a new posterior belief, i.e., a new mean $\mu^{(k)}$ and a new precision $\pi^{(k)}$, given a new input from the level (node) below (usually, a prediction error δ), and the node's own prediction ($\hat{\mu}^{(k)}$ and $\hat{\pi}^{(k)}$). In this case, the exact computations within a node depend on the nature of its children: If node b is the value parent of node a, then the following update equations apply to node b:

$$\pi_b^{(k)} = \hat{\pi}_b^{(k)} + \alpha_{b,a}^2 \hat{\pi}_a^{(k)} \tag{32}$$

$$\mu_b^{(k)} = \hat{\mu}_b^{(k)} + \frac{\alpha_{b,a}\hat{\pi}_a^{(k)}}{\pi_b^{(k)}} \delta_a^{(k)} \tag{33}$$

Thus, at the time of the update, node i needs to have access to the following quantities:

Its own prediction: $\hat{\mu}_b^{(k)}$, $\hat{\pi}_b^{(k)}$

Coupling strength: $\alpha_{b,a}$

From level below: $\delta_a^{(k)}$, $\hat{\pi}_a^{(k)}$

All of these are available at the time of the update. Node b therefore only needs to receive the PE and the precision of the prediction from the child nodes to perform its update. The middle panel of Figure 3 illustrates these computations.

For a node $\breve{a}$ which is the volatility parent of node a, the update equations for computing a new posterior mean $\mu_{\breve{a}}^{(k)}$ and a new posterior precision $\pi_{\breve{a}}^{(k)}$ have been described by Mathys et al. (2011). Here, we will introduce two changes to the notation to simplify (the implementation of) these equations:

First, we will express the volatility PE, or VOPE, as a function of the previously defined value PE, or VAPE. That means from now on, we will use the symbol δ only for VAPEs:

$$\delta_a^{(k)} \equiv \delta_a^{(k,VAPE)} = \mu_a^{(k)} - \hat{\mu}_a^{(k)}, \tag{34}$$

and introduce a new symbol Δ for VOPEs, which we define as

$$\begin{aligned} \Delta_a^{(k)} \equiv \delta_a^{(k,VOPE)} &= \hat{\pi}_a^{(k)} \left(\frac{1}{\pi_a^{(k)}} + \left(\delta_a^{(k)} \right)^2 \right) - 1 \\ &= \frac{\hat{\pi}_a^{(k)}}{\pi_a^{(k)}} + \hat{\pi}_a^{(k)} \left(\delta_a^{(k)} \right)^2 - 1. \end{aligned} \tag{35}$$

For a derivation of this definition based on the equations presented in Mathys et al. (2011), please refer to Appendix 5.2.

Second, we will introduce another quantity, which reflects the volatility-weighted precision of the prediction:

$$\gamma_a^{(k)} = \Omega_a^{(k)} \hat{\pi}_a^{(k)}, \tag{36}$$

which will be computed as part of the prediction step and will be termed effective precision of the prediction due to its role in the update equations. This definition serves to simplify the equations and the corresponding message passing.

With these two changes, namely the definitions of the VOPE $\Delta^{(k)}$ and the effective precision of the prediction $\gamma^{(k)}$, the update equations for the precision and the mean of volatility parent $\check{a}$ simplify to:

$$\pi_{\check{a}}^{(k)} = \hat{\pi}_{\check{a}}^{(k)} + \frac{1}{2} \left(\kappa_{\check{a},a} \gamma_a^{(k)} \right)^2 + \left(\kappa_{\check{a},a} \gamma_a^{(k)} \right)^2 \Delta_a^{(k)} - \frac{1}{2} \kappa_{\check{a},a}^2 \gamma_a^{(k)} \Delta_a^{(k)} \tag{37}$$

$$\mu_{\check{a}}^{(k)} = \hat{\mu}_{\check{a}}^{(k)} + \frac{1}{2} \frac{\kappa_{\check{a},a} \gamma_a^{(k)}}{\pi_{\check{a}}^{(k)}} \Delta_a^{(k)} \tag{38}$$

Therefore, at the time of the update, volatility parent node $\check{a}$ needs to have access to the following quantities:

Its own prediction: $\hat{\mu}_{\check{a}}^{(k)}$, $\hat{\pi}_{\check{a}}^{(k)}$

Coupling strength: $\kappa_{\check{a},a}$

From level below: $\Delta_a^{(k)}$, $\gamma_a^{(k)}$

These equations are illustrated in the middle panel of Figure 4. We again note here the structural similarities between nodes that serve as value parents and nodes that serve as volatility parents: updates of the mean are always driven by precision-weighted prediction errors, and updates of the precision require some estimate of the prediction of the precision of the child node ($\hat{\pi}_a$ or γ_a). These similarities allow us to make statements about the message passing architecture within and across nodes that generalize across coupling types (see Figure 2B).

An interesting difference to the implied message passing in predictive coding proposals (Bastos et al., 2012; Shipp, 2016) arises from the update step: The HGF architecture requires that not only (precision-weighted) prediction errors are being sent bottom-up between nodes, but also estimates of prediction precision ($\hat{\pi}_a$ or γ_a) which serve to update belief precision in the higher-level node.

The prediction error step

Finally, in the PE step, a node computes the deviation of its recently updated posterior from its trial- and parent-specific prediction. This can result in two different types of PEs: VAPEs and VOPEs. These will, in turn, be used to communicate with the node's parent nodes, if it has any. Therefore, this step again depends on the nature of a node's parent nodes and can also be considered as the process of gathering all the information required by any existing parents. In addition to the PE, parent nodes will require some estimate of the precision of the prediction (see the previous section on the update step).

If node a is the value child of node b, the following quantities have to be sent up to node b:

Precision of the prediction: $\hat{\pi}_a^{(k)}$

Prediction error: $\delta_a^{(k)}$

Node a has already performed the prediction step (see above), so it has already computed the precision of the prediction for the current trial, $\hat{\pi}_a^{(k)}$. Hence, in the PE step, it needs to perform only the following calculation (illustrated in the right panel of Figure 3):

$$\delta_a^{(k)} = \mu_a^{(k)} - \hat{\mu}_a^{(k)} \tag{39}$$

Note again here, that $\delta_a^{(k)}$ represents a prediction error from the perspective of the parent node - the difference between the expected state of the child and the actual state on trial k. From the perspective of the child node a, the difference between its prior and its posterior instead represents a belief update (Bayesian surprise). These two will only differ from each other in the case of multiple parents.
Further, if node a is the volatility child of node $\check{a}$, the following quantities have to be sent up to node $\check{a}$ (see also necessary information from level below in a volatility parent's update step):

Effective precision of the prediction: $\gamma_a^{(k)}$

Prediction error: $\Delta_a^{(k)}$

Node a has already performed the prediction step on the previous trial, so it has already computed the precision of the prediction, $\hat{\pi}_a^{(k)}$, and the total predicted volatility, $\Omega_a^{(k)}$, and out of these the effective precision of the prediction, $\gamma_a^{(k)}$, for the current trial. Hence, in the PE step, it needs to perform only the following calculations (illustrated in the right panel of Figure 4):

$$\delta_a^{(k)} = \mu_a^{(k)} - \hat{\mu}_a^{(k)} \tag{40}$$

$$\Delta_a^{(k)} = \frac{\hat{\pi}_a^{(k)}}{\pi_a^{(k)}} + \hat{\pi}_a^{(k)} \left(\delta_a^{(k)}\right)^2 - 1. \tag{41}$$

In other words, if node a has any parents, the VAPE will always be computed (as it features in both scenarios), whereas the computation of a VOPE is only necessary if node a has a volatility parent.

Differences for nonlinear mean coupling

So far, we have assumed linear value coupling in presenting the computations of value parent and children nodes. However, in the case of nonlinear value coupling, the update equations only change slightly. Specifically, in the prediction step, we now see function g appear during the computation of the new predicted mean. Assuming that node a is the (nonlinear) value child of node b, the total predicted mean drift for trial k will be

$$P_a^{(k)} = t^{(k)} \left(\rho_a + \sum_{i=1}^{N_{vapa}} \alpha_{b_i,a} g \left(\mu_{b_i}^{(k-1)} \right) \right). \tag{42}$$

In other words, the influence of the higher-level belief $\mu_{b_i}^{(k-1)}$ on the prediction of the lower-level belief $\hat{\mu}_a^{(k)}$ is mediated by the function g, just as we would expect it to be based on the generative model (equation 45).

In the update step for the value parent, we now have:

$$\pi_b^{(k)} = \hat{\pi}_b^{(k)} + \hat{\pi}_a^{(k)} \left(\alpha_{b,a}^2 g' \left(\mu_c^{(k-1)} \right) - \alpha_{b,a} g'' \left(\mu_c^{(k-1)} \right) \delta_a^{(k)} \right) \tag{43}$$

$$\mu_b^{(k)} = \hat{\mu}_b^{(k)} + \frac{\alpha_{b,a} g' \left(\mu_c^{(k-1)} \right) \hat{\pi}_a^{(k)}}{\pi_b^{(k)}} \delta_a^{(k)} \tag{44}$$

Consequently, for the update step, the node b now also needs access to its own previous posterior mean $\mu_c^{(k-1)}$. Apart from these changes, all update equations from the previous section apply.

Multiple parent nodes

In Appendix 5.3, we briefly take a closer look at the case where one node has multiple parent nodes of the same coupling type (i.e., either multiple volatility parents, or multiple value parents). As equations 30 and 31 show, the predicted drift and volatility for the child node depend on the sum of all parent node influences. We will call this the node's overall prediction, which will then be used together with bottom-up input to compute the node's posterior and an overall belief update (Bayesian surprise). However, in the update step, the parent nodes should receive a PE message which tells them specifically how much their own prediction of the child node was off (to be able to update their beliefs about the child node's value or volatility accordingly). For this purpose, we will introduce parent-specific predictions and prediction errors in the child node, which ensure that the update of the parent nodes takes into account their own predictions about the child node and the overall child posterior.

This implementation means that the communication of nodes with their same-coupling-type parents does not only differ between these parents in terms of connection weights, but also in terms of the content that is being sent bottom-up from child to parent. The PE thus is a quantity that belongs to a connection rather than a specific node.

The special case of input nodes

At the lowest level of the belief hierarchy, nodes do not receive prediction errors from other nodes, but are directly fed with sensory inputs, which can either be continuous or binary. This leads to some special cases of nodes: continuous input nodes, binary input nodes, and binary HGF nodes (parents of binary input nodes). These nodes are perhaps less interesting for examining the associated message-passing from a theoretical point of view: for most neuroscientific applications of interest, we would assume that the lowest level modeled (e.g., primary sensory cortices) is already somewhat distant to the actual sensors (e.g., the retina), which means we can cast its inputs as prediction errors computed during upstream processing of the input (e.g., in subcortical structures). However, for implementation of our perceptual model in software for practical applications, where we tend to neglect this intermediate processing and let experimental stimuli enter the lowest levels of our belief hierarchy directly, we require treatment of these special kinds of nodes within our new framework. This treatment is presented in Appendix 5.4. In there, we also deal with the case where the agent forms an explicit and dynamic belief about the level of observation noise (stochasticity) in a particular outcome (noise coupling).

Summary: A network of nodes

In summary, in this section, we have used the update equations of the HGF to propose a conceptualization of the inference machinery as a network of nodes which compute beliefs and exchange messages with other nodes. Every node in this network represents an agent's current belief about a hidden state in her environment, on which she infers given her sensory inputs. Within every node, belief updating in response to a new input proceeds in three steps (an update step, a PE step, and a prediction step).

We have presented the computations for these steps for the two different kinds of coupling that the HGF comprises: value coupling and volatility coupling. While the update equations for volatility coupling have been derived and discussed previously (Mathys et al., 2011, 2014), approximately Bayes-optimal inference equations for (linear and nonlinear) value coupling under the HGF have not been considered prior to our treatment here. Furthermore, our analysis identifies not only the computations entailed by each computational step, but also the message passing between nodes that is required by each step. This is interesting from a theoretical point of view, where we can compare our architecture to other proposals of belief propagation.

From a practical point of view, the division of the belief updating machinery into subunits (nodes) allows for a modular implementation, where networks can easily be extended and modified by adding or removing nodes, or by changing the type of coupling between nodes, without having to derive the relevant equations for the whole network anew. In two open-source projects, we provide such an implementation (in Python: `https://github.com/ilabcode/pyhgf`, Legrand et al. (in prep); in Julia: `https://github.com/ilabcode/HierarchicalGaussianFiltering.jl`, Thestrup Waade et al. (in prep)), which allows users to flexibly design their own HGF structures that can be used for simulation and empirical parameter estimation. These tools will also be available as part of the TAPAS software collection (Frässle et al., 2021).

What conclusions can be drawn with respect to the message passing implied by the HGF? First, while the exact computations performed during the three computational steps depend on the position of the node within the network (e.g., number of children and parent nodes) and the nature of the coupling to other nodes (value vs. volatility coupling), we have identified generic structures in these equations (see Figure 2B), which are of interest from a theoretical point of view, but also facilitate implementation.

For example, belief updates in a node always require messages from lower-level nodes that contain prediction errors (δ_a for value coupling, Δ_a for volatility coupling) and estimates of precision ($\hat{\pi}_a^{(k)}$ in value coupling and $\gamma_a^{(k)}$ in volatility coupling). Similarly, forming a new prediction always entails modifying the mean of the belief by an expectation of drift, and modifying the precision by an expectation of volatility during the next time step. Expectations of drift will be driven by value parents, expectations of volatility by volatility parents, but the structure of the equations is the same for both types of coupling (see equations 28 to 31).

In Figures 3 and 4, we additionally provide a more detailed overview of what happens within and between nodes for specific coupling types. For this purpose, we additionally consider separate subunits for calculations concerning means versus precisions versus prediction errors. Mapping these architectures to structures and networks in biological brains will be an exciting future task.

4 Conclusions

Our work makes several contributions. First, our extension to value coupling includes principles of predictive coding in the HGF framework. This offers a general and versatile modelling framework, offering an approximation to optimal Bayesian inference for different types of interactions between states in the world, allowing for inter-individual differences in the dynamics of belief updating, and providing a principled treatment of the multiple forms of uncertainties agents are confronted with.

Second, we present a modular architecture for the perceptual model of the HGF, where beliefs represent nodes in a network that performs three basic computational steps: an Update step, a PE step, and a prediction step, in response to each sensory input. While the equations for these steps differ depending on the coupling of a node to other nodes, we identify a generic structure that allows for a modular implementation, in which nodes can easily be added to or removed from a network, without having to derive the ensuing update equations for the model anew. We provide such an implementation in two open-source projects (in Python: `https://github.com/ilabcode/pyhgf`, Legrand et al. (in prep); in Julia: `https://github.com/ilabcode/HierarchicalGaussianFiltering.jl`, Thestrup Waade et al. (in prep); inclusion in the TAPAS software collection (Frässle et al., 2021) is pending).

Finally, by considering the case of value coupling and deriving the message passing scheme implied by the HGF, we enable a formal comparison to other proposed architectures for hierarchical Bayesian inference, most prominently Bayesian (or generalized) predictive coding.

4.1 Modelling different sources of uncertainty

Whenever agents are faced with observations that violate their expectations, they need to arbitrate between different explanations – has the world changed, requiring an update of beliefs about hidden states, or was the deviation only due to noise in their observations? As has been shown previously (Mathys et al., 2011, 2014), the HGF models belief updating in an agent who takes into account several forms of uncertainty for determining the optimal learning rate in the face of new observations: sensory uncertainty (how noisy are the sensory inputs I receive), informational uncertainty (how much do I already know about the hidden state that generates the inputs), and environmental uncertainty (what is the rate of change I expect in the hidden state). All of these together will determine whether (and how much) the agent updates her beliefs about a hidden state in response to unexpected observations.

Here, we show that under the HGF, the agent cannot only learn about environmental volatility – where higher estimates of volatility lead to faster learning, but in an undirected fashion –, but also about higher-level hidden states that cause changes in lower-level hidden states in a directed fashion. For example, the weather might be more volatile in some seasons compared to others, making the agent less certain in her predictions (and faster to learn) about the likelihood of rainfall (volatility coupling). On the other hand, she might expect more or less rainfall in certain seasons (value coupling).

The flexibility under the HGF in building models of hierarchically interacting states in the world allows for a few particularly interesting cases. In section 2, Figure 1, we have provided an example where two hidden states (x_a and x_d) evolve with their own respective evolution rates (both determined by a tonic component: ω_a and ω_d, and a phasic component: $x_{\check{a}}$ and $x_{\check{d}}$), but share a higher-level value parent (x_f) that drives changes in the mean of their respective phasic volatility states. This setup allows for separate estimation of ”local” volatility (specific to each hidden state) and more global influences on volatility. For example, the rate of change in the availability of two different foods in the environment might vary over time and seasons in a fashion that is specific to each type of food, but when switching to a different environment (or after a global change to the overall climate), the availability of both foods might become more or less stable. Investigating how the brain represents both forms of volatility, and thus tunes learning rates in a modality-specific versus general manner is an important part of understanding how brains achieve and maintain the delicate balance of precision over different hierarchical levels (Kanai et al., 2015; Clark, 2013) that seems crucial for mental health (Petzschner et al., 2017; Sterzer et al., 2018).

Finally, agents are confronted not only with phasic variations of volatility (driving changes in the agent's environmental uncertainty), but also with dynamically changing sensory (or observation) noise. While existing modelling approaches have focused on either accounting for changes in volatility (Behrens et al., 2007; Mathys et al., 2011; Piray & Daw, 2020) or changes in stochasticity or noise (Lee et al., 2020; Nassar et al., 2010), it has recently been pointed out that real-world agents need to able to detect (and distinguish) changes in both at the same time (Piray & Daw, 2021). In the HGF, dynamic changes in observation noise can be accommodated by hidden states that serve as noise parents to observable outcome states (see Appendix 5.4). Jointly modelling an agent's inference on volatility and stochasticity will be crucial to understanding the computational origin of maladaptive inference and learning in different psychiatric conditions (Piray & Daw, 2021; Pulcu & Browning, 2019; Mikus et al., 2022).

4.2 Implementing the HGF's message passing scheme

Hierarchical filtering and predictive coding are two prominent classes of hierarchical Bayesian models that cast perception as inference and model belief updates in proportion to precision-weighted prediction errors. Models that fall in either of these classes are widely used, both in basic (computational) neuroscience, and for understanding mental disorders in computational psychiatry (Petzschner et al., 2017).

While the message passing architecture implied by different predictive coding models has been examined in detail and partly matched with neuroanatomy and -physiology (for overviews, see Spratling, 2019; Keller & Mrsic-Flogel, 2018; Bastos et al., 2012; Shipp, 2016), and hierarchical filtering models share many similarities with predictive coding models, it is currently not clear whether the respective inference networks would pose differing requirements for implementation (in computers or brains), or make differing predictions about neural readouts of perceptual inference and learning. This is partly due to their non-overlapping applications: predictive coding models consider hierarchies in which higher levels impact on the mean of lower levels, and they are typically used to model inference about static sensory inputs in continuous time.

Here, we reduce this gap by presenting the message-passing scheme implied by the HGF for value coupling (alongside volatility coupling). Our results show that *(1)* the HGF inference network for value coupling is largely compatible with recently proposed predictive coding architectures in that messages passed between nodes of the network contain a bottom-up signalling of precision-weighted prediction errors, and a top-down influence on predictions and *(2)* there are small but interesting differences that relate to updates of belief uncertainty.

One noteworthy difference between the architectures is that the update equations in the HGF require a bottom-up transmission of lower-level precision estimates (Figures 2 and 3)[2]. This prediction is interesting, given that recent neuroanatomical studies point

[2] It is not surprising that the differences between the models concern the signalling of precision: The HGF derivation explicitly includes update equations for the precision associated with beliefs - as do other hierarchical Bayesian architectures based on Markovian processes (Friston et al., 2013). In contrast, most predictive coding schemes still focus on the optimization of the first moment (mode or expectation) of the posterior distribution for perceptual inference (Friston, 2005; Bogacz, 2017),

to additional pathways besides the classical forward (from lower-level supragranular to higher-level granular layers) and backward (from higher-level infragranular to lower-level extragranular layers) connections (Markov et al., 2013, 2014). Strikingly, our architecture would be compatible with an ascending connection within supragranular layers (for bottom-up communication of lower-level precision) that runs in parallel to a descending connection within these layers (for top-down modulation of PEs by higher-level precision), reminiscent of the 'cortical counter streams' identified by these studies. We hope to capitalize on methodological advancements in high-resolution laminar fMRI (Stephan et al., 2019; Haarsma et al., 2022) in future studies to test these predictions.

4.3 Within-trial versus across-trial dynamics

Although potential neurobiological implementations of approximate Bayesian inference in the brain are still hotly debated (Knill & Pouget, 2004; Aitchison & Lengyel, 2017), a growing body of literature suggests that predictive coding-like architectures can account for a large range of neurophysiological findings in perception research (for a recent overview, see Walsh et al., 2020), making these architectures particularly relevant for understanding human perception and inference. It would thus be interesting to examine in detail the commonalities and potential differences between message-passing schemes implied by different forms of coupling in HGF models and classical predictive coding schemes, as we have started to do here.

Importantly, however, direct comparisons of the two models are further complicated by differences in each model's concept of time: while the HGF captures belief updates across trials in discrete time (i.e., across sequentially arriving sensory inputs), predictive coding describes the evolution of beliefs in continuous time and, typically, in response to static sensory input (Rao & Ballard, 1999; Friston, 2005)[3]. In practice, differential equations capture the evolution of beliefs and predictions errors and are used to simulate perceptual inference, starting with new input to the lowest level of the belief hierarchy, and ending when the ensuing PEs have been reconciled, i.e., a stable new posterior belief has emerged (Bogacz, 2017). Together with its strong link to neurobiology, predictive coding thus makes predictions about neural activity that can be compared against measurements. While, to our knowledge, existing predictive coding models have not been fit to data, the simulated neural dynamics display many features that are observed in real data, such as oscillatory tendencies, even in very small networks (Bogacz, 2017). We refer to these simulations as within-trial dynamics of belief updating.

On the other hand, the generative model of the HGF represents a Markovian process in discrete time; in the inference model, one-step update equations, derived based on a mean field approximation to the full Bayesian solution, quantify the change from prior to posterior on all levels of the belief hierarchy. The model is thus examined in sequential input settings to capture trial-by-trial learning - in other words, across-trial dynamics of belief updating. The model provides an approximately Bayes-optimal solution to trial-wise belief updating, useful for ideal observer analyses (Stefanics et al., 2018; Weber et al., 2020, 2022; Hauke et al., 2022), but the parameters of the HGF can also be fit to behavioral responses of individual participants. Model-derived agent-specific trajectories of trial-wise predictions and PEs have proven particularly useful for identifying potential neural and physiological correlates of computational quantities in empirical data (Iglesias et al., 2013; Vossel et al., 2015; de Berker et al., 2016; Diaconescu et al., 2017; Weilnhammer et al., 2018; Katthagen et al., 2018; Palmer et al., 2019; Deserno et al., 2020; Henco et al., 2020; Cole et al., 2020; Lawson et al., 2021;

although the variational approach does allow approximation of the full posterior distribution including its variance.

[3] In fact, the standard form of the generative model in predictive coding does not contain temporal dynamics of the hidden states, but see Friston (2008, 2010) for extensions.

Hein et al., 2021; Hein & Herrojo Ruiz, 2022; Harris et al., 2022; Fromm et al., 2023). In the future, we propose to explicitly consider potential within-trial dynamics of belief updates compatible with the update equations of the HGF (see Weber, 2020, for a first attempt). Such equations could for example be derived by treating the HGF posterior values of all nodes (quantities) as the equilibrium point towards which all dynamics must converge (inspired by Bogacz, 2017). Establishing equations for within-trial belief updating dynamics under the HGF might allow for empirical tests of the proposed architecture in a two-step procedure: first, individual trajectories of predictions and prediction errors are inferred from observed behavior by fitting the HGF to participants' responses, second, these trial-wise point estimates are subsequently used to simulate expected continuous-time neuronal responses according to the differential equations.

In summary, in this paper, we have presented a generalization of the HGF that extends its scope of hierarchical inference mechanisms to include cross-level couplings as proposed by predictive coding. Furthermore, we have demonstrated how this extension can be cast as a modular architecture that allows flexible changes to a model without having to re-derive update equations. We hope that the availability of these developments as open source software will expand the toolkit of computational psychiatry and facilitate future investigations of perceptual inference in health and disease.

Acknowledgements

This work was supported by the René and Susanne Braginsky Foundation (KES), the Aarhus Universitets Forskningsfond (grant AUFF-E-2019-7-10) (CM), and the Carlsberg Foundation (grant CF21-0439) (CM).

References

Adams, R. A., Stephan, K. E., Brown, H. R., Frith, C. D., & Friston, K. J. (2013). The computational anatomy of psychosis. *Frontiers in Psychiatry*, *4*(47), 1–26.

Adelson, E. H. (2005). Checkershadow illusion. 1995. *URL http://web. mit. edu/persci/people/adelson/checkershadow_illusion. html*.

Aitchison, L., & Lengyel, M. (2017). With or without you: predictive coding and bayesian inference in the brain. *Current opinion in neurobiology*, *46*, 219–227.

Angelaki, D. E., Gu, Y., & DeAngelis, G. C. (2009). Multisensory integration: psychophysics, neurophysiology, and computation. *Current Opinion in Neurobiology*, *19*(4), 452–458.

Bastos, A. M., Usrey, W. M., Adams, R. A., Mangun, G. R., Fries, P., & Friston, K. J. (2012). Canonical Microcircuits for Predictive Coding. *Neuron*, *76*(4), 695–711.

Behrens, T. E. J., Woolrich, M. W., Walton, M. E., & Rushworth, M. F. S. (2007). Learning the value of information in an uncertain world. *Nat Neurosci*, *10*, 1214–1221.

Bogacz, R. (2017). A tutorial on the free-energy framework for modelling perception and learning. *Journal of Mathematical Psychology*, *76*, 198–211.

Clark, A. (2013). Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, *36*(3), 181–204.

Cole, D. M., Diaconescu, A. O., Pfeiffer, U. J., Brodersen, K. H., Mathys, C. D., Julkowski, D., Ruhrmann, S., Schilbach, L., Tittgemeyer, M., Vogeley, K., & Stephan, K. E. (2020). Atypical processing of uncertainty in individuals at risk for psychosis. *NeuroImage: Clinical*, *26*, 102239.

Corlett, P. R., Frith, C. D., & Fletcher, P. C. (2009). From drugs to deprivation: A Bayesian framework for understanding models of psychosis. *Psychopharmacology*, *206*(4), 515–530.

Corlett, P. R., Honey, G., Krystal, J., & Fletcher, P. (2011). Glutamatergic model psychoses: Prediction error, learning, and inference. *Neuropsychopharmacology*, *36*, 294–315.

Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz Machine. *Neural Computation*, *7*(5), 889–904.

de Berker, A. O., Rutledge, R. B., Mathys, C., Marshall, L., Cross, G. F., Dolan, R. J., & Bestmann, S. (2016). Computations of uncertainty mediate acute stress responses in humans. *Nature Communications*, *7*(1), 10996. Number: 1 Publisher: Nature Publishing Group.

Deserno, L., Boehme, R., Mathys, C., Katthagen, T., Kaminski, J., Stephan, K. E., Heinz, A., & Schlagenhauf, F. (2020). Volatility Estimates Increase Choice Switching and Relate to Prefrontal Activity in Schizophrenia. *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, *5*(2), 173–183.

Diaconescu, A. O., Mathys, C. D., Weber, L. A., Kasper, L., Mauer, J., & Stephan, K. E. (2017). Hierarchical prediction errors in midbrain and septum during social learning. *Social Cognitive and Affective Neuroscience*, *12*(4), 618–634.

Doya, K., Ishii, S., Pouget, A., & Rao, R. P. N. (2011). *Bayesian brain: probabilistic approaches to neural coding*. Cambridge: MIT Press.

Drusko, A., Baumeister, D., McPhee Christensen, M., Kold, S., Fisher, V. L., Treede, R.-D., Powers, A., Graven-Nielsen, T., & Tesarz, J. (2023). A novel computational approach to pain perception modelling within a Bayesian framework using quantitative sensory testing. *Scientific Reports*, *13*(1), 3196. Number: 1 Publisher: Nature Publishing Group.

Ernst, M. O., & Banks, M. S. (2002). Ernst 2002 Humans integrate visual and haptic information in a. *Nature*, *415*(January), 429–433.

Fletcher, P. C., & Frith, C. D. (2009). Perceiving is believing: a Bayesian approach to explaining the positive symptoms of schizophrenia. *Nature Reviews Neuroscience*, *10*(1), 48–58.

Friston, K. (2008). Hierarchical Models in the Brain. *PLoS Computational Biology*, *4*(11), e1000211.

Friston, K., Schwartenbeck, P., Fitzgerald, T., Moutoussis, M., Behrens, T., & Dolan, R. (2013). The anatomy of choice: active inference and agency. *Frontiers in Human Neuroscience*, *7*.

Friston, K. J. (2005). A theory of cortical responses A theory of cortical responses. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, *360*, 815–836.

Friston, K. J. (2010). The free-energy principle: a unified brain theory? *Nature reviews. Neuroscience*, *11*(2), 127–38.

Friston, K. J., Brown, H. R., Siemerkus, J., & Stephan, K. E. (2016). The dysconnection hypothesis (2016). *Schizophrenia Research*, *176*(2-3), 83–94.

Fromm, S., Katthagen, T., Deserno, L., Heinz, A., Kaminski, J., & Schlagenhauf, F. (2023). Belief Updating in Subclinical and Clinical Delusions. *Schizophrenia Bulletin Open*, *4*(1), sgac074.

Frässle, S., Aponte, E. A., Bollmann, S., Brodersen, K. H., Do, C. T., Harrison, O. K., Harrison, S. J., Heinzle, J., Iglesias, S., Kasper, L., Lomakina, E. I., Mathys, C., Müller-Schrader, M., Pereira, I., Petzschner, F. H., Raman, S., Schöbi, D., Toussaint, B., Weber, L. A., Yao, Y., & Stephan, K. E. (2021). TAPAS: An Open-Source Software Package for Translational Neuromodeling and Computational Psychiatry. *Frontiers in Psychiatry*, *12*, 680811.

Haarsma, J., Kok, P., & Browning, M. (2022). The promise of layer-specific neuroimaging for testing predictive coding theories of psychosis. *Schizophrenia Research*, *245*, 68–76.

Hahnloser, R. H., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., & Seung, H. S. (2000). Digital selection and analogue amplification coexist in a cortex- inspired silicon circuit. *Nature*, *405*(6789), 947–951.

Harris, D. J., Arthur, T., Vine, S. J., Liu, J., Abd Rahman, H. R., Han, F., & Wilson, M. R. (2022). Task-evoked pupillary responses track precision-weighted prediction errors and learning rate during interceptive visuomotor actions. *Scientific Reports*, *12*(1), 22098. Number: 1 Publisher: Nature Publishing Group.

Hauke, D. J., Charlton, C. E., Schmidt, A., Griffiths, J., Woods, S. W., Ford, J. M., Srihari, V. H., Roth, V., Diaconescu, A. O., & Mathalon, D. H. (2022). Aberrant hierarchical prediction errors are associated with transition to psychosis: A computational single-trial analysis of the mismatch negativity. Pages: 2022.12.20.22283712.
URL https://www.medrxiv.org/content/10.1101/2022.12.20.22283712v1

Hein, T. P., de Fockert, J., & Ruiz, M. H. (2021). State anxiety biases estimates of uncertainty and impairs reward learning in volatile environments. *NeuroImage*, *224*, 117424.

Hein, T. P., & Herrojo Ruiz, M. (2022). State anxiety alters the neural oscillatory correlates of predictions and prediction errors during reward-based learning. *NeuroImage*, *249*, 118895.

Helmholtz, H. v. (1860). *Handbuch der physiologischen Optik (Vol. 3)*. English translation (1962): Southall JPC.

Henco, L., Brandi, M.-L., Lahnakoski, J. M., Diaconescu, A. O., Mathys, C., & Schilbach, L. (2020). Bayesian modelling captures inter-individual differences in social belief computations in the putamen and insula. *Cortex*, *131*, 221–236.

Iglesias, S., Mathys, C. D., Brodersen, K. H., Kasper, L., Piccirelli, M., DenOuden, H. E., & Stephan, K. E. (2013). Hierarchical Prediction Errors in Midbrain and Basal Forebrain during Sensory Learning. *Neuron*, *80*(2), 519–530.

Kafadar, E., Fisher, V. L., Quagan, B., Hammer, A., Jaeger, H., Mourgues, C., Thomas, R., Chen, L., Imtiaz, A., Sibarium, E., Negreira, A. M., Sarisik, E., Polisetty, V., Benrimoh, D., Sheldon, A. D., Lim, C., Mathys, C., & Powers, A. R. (2022). Conditioned Hallucinations and Prior Overweighting Are State-Sensitive Markers of Hallucination Susceptibility. *Biological Psychiatry*, *92*(10), 772–780.

Kanai, R., Komura, Y., Shipp, S., & Friston, K. J. (2015). Cerebral hierarchies: predictive processing, precision and the pulvinar. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *370*(1668), 20140169–20140169.

Katthagen, T., Mathys, C., Deserno, L., Walter, H., Kathmann, N., Heinz, A., & Schlagenhauf, F. (2018). Modeling subjective relevance in schizophrenia and its relation to aberrant salience. *PLOS Computational Biology*, *14*(8), e1006319. Publisher: Public Library of Science.

Keller, G. B., & Mrsic-Flogel, T. D. (2018). Predictive processing: a canonical cortical computation. *Neuron*, *100*(2), 424–435.

Knill, D. C., & Pouget, A. (2004). The bayesian brain: the role of uncertainty in neural coding and computation. *TRENDS in Neurosciences*, *27*(12), 712–719.

Lawson, R. P., Bisby, J., Nord, C. L., Burgess, N., & Rees, G. (2021). The Computational, Pharmacological, and Physiological Determinants of Sensory Learning under Uncertainty. *Current Biology*, *31*(1), 163–172.e4.

Lawson, R. P., Mathys, C., & Rees, G. (2017). Adults with autism overestimate the volatility of the sensory environment. *Nature Neuroscience*, *20*(9), 1293–1299. Number: 9 Publisher: Nature Publishing Group.

Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Lee, S., Gold, J. I., & Kable, J. W. (2020). The human as delta-rule learner. *Decision*, *7*(1), 55.

Legrand, N., Weber, L., Thestrup Waade, P., Møller, A. H., Allen, M., & Mathys, C. (in prep). pyhgf: The generalized, nodalized and multilevel Hierarchical Gaussian Filter for predictive coding.

Markov, N. T., Ercsey-Ravasz, M., Van Essen, D. C., Knoblauch, K., Toroczkai, Z., & Kennedy, H. (2013). Cortical high-density counterstream architectures. *Science*, *342*(6158).

Markov, N. T., Vezoli, J., Chameau, P., Falchier, A., Quilodran, R., Huissoud, C., Lamy, C., Misery, P., Giroud, P., Ullman, S., Barone, P., Dehay, C., Knoblauch, K., & Kennedy, H. (2014). Anatomy of hierarchy: Feedforward and feedback pathways in macaque visual cortex. *Journal of Comparative Neurology*, *522*(1), 225–259.

Mathys, C., & Weber, L. (2020). Hierarchical Gaussian Filtering of Sufficient Statistic Time Series for Active Inference. In T. Verbelen, P. Lanillos, C. L. Buckley, & C. De Boom (Eds.) *Active Inference*, vol. 1326, (pp. 52–58). Cham: Springer International Publishing. Series Title: Communications in Computer and Information Science.
URL https://link.springer.com/10.1007/978-3-030-64919-7_7

Mathys, C. D. (2016). How could we get nosology from computation? In *Computational Psychiatry: New Perspectives on Mental Illness*, (pp. 121–135). MIT Press.
URL https://esforum.de/publications/sfr20/ComputationalPsychiatry.html

Mathys, C. D., Daunizeau, J., Friston, K. J., & Stephan, K. E. (2011). A Bayesian foundation for individual learning under uncertainty. *Frontiers in Human Neuroscience*, *5*(May), 1–20.

Mathys, C. D., Lomakina, E. I., Daunizeau, J., Iglesias, S., Brodersen, K. H., Friston, K. J., & Stephan, K. E. (2014). Uncertainty in perception and the Hierarchical Gaussian Filter. *Frontiers in Human Neuroscience*, *8*(November), 1–24.

Mikus, N., Lamm, C., & Mathys, C. (2022). Computational phenotyping of aberrant belief updating in schizotypy. In *European College of Neuropsychopharmacology Congress*.

Nassar, M. R., Wilson, R. C., Heasly, B., & Gold, J. I. (2010). An approximately bayesian delta-rule model explains the dynamics of belief updating in a changing environment. *Journal of Neuroscience*, *30*(37), 12366–12378.

Palmer, C. E., Auksztulewicz, R., Ondobaka, S., & Kilner, J. M. (2019). Sensorimotor beta power reflects the precision-weighting afforded to sensory prediction errors. *NeuroImage*, *200*, 59–71.

Petzschner, F. H., Weber, L. A., Gard, T., & Stephan, K. E. (2017). Computational Psychosomatics and Computational Psychiatry: Toward a Joint Framework for Differential Diagnosis. *Biological Psychiatry*, *82*(6), 421–430.

Piray, P., & Daw, N. D. (2020). A simple model for learning in volatile environments. *PLoS computational biology*, *16*(7), e1007963.

Piray, P., & Daw, N. D. (2021). A model for learning based on the joint estimation of stochasticity and volatility. *Nature communications*, *12*(1), 6587.

Powers, A. R., Mathys, C., & Corlett, P. R. (2017). Pavlovian conditioning-induced hallucinations result from overweighting of perceptual priors. *Science (New York, N.Y.)*, *357*(6351), 596–600.

Pulcu, E., & Browning, M. (2019). The misestimation of uncertainty in affective disorders. *Trends in Cognitive Sciences*, *23*(10), 865–875.

Rao, R. P. N., & Ballard, D. H. (1999). Predictive Coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, *2*(1), 79–87.

Rossi-Goldthorpe, R. A., Leong, Y. C., Leptourgos, P., & Corlett, P. R. (2021). Paranoia, self-deception and overconfidence. *PLOS Computational Biology*, *17*(10), e1009453. Publisher: Public Library of Science.

Sapey-Triomphe, L.-A., Weilnhammer, V. A., & Wagemans, J. (2022). Associative learning under uncertainty in adults with autism: Intact learning of the cue-outcome contingency, but slower updating of priors. *Autism*, *26*(5), 1216–1228.

Sevgi, M., Diaconescu, A. O., Henco, L., Tittgemeyer, M., & Schilbach, L. (2020). Social Bayes: Using Bayesian Modeling to Study Autistic Trait–Related Differences in Social Cognition. *Biological Psychiatry*, *87*(2), 185–193.

Shipp, S. (2016). Neural elements for predictive coding. *Frontiers in Psychology*, *7*, 1–21.

Siegel, J. Z., Curwell-Parry, O., Pearce, S., Saunders, K. E. A., & Crockett, M. J. (2020). A Computational Phenotype of Disrupted Moral Inference in Borderline Personality Disorder. *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging*, *5*(12), 1134–1141.

Spratling, M. (2019). Fitting predictive coding to the neurophysiological data. *Brain research*, *1720*, 146313.

Stefanics, G., Heinzle, J., Horváth, A. A., & Stephan, K. E. (2018). Visual Mismatch and Predictive Coding: A Computational Single-Trial ERP Study. *Journal of Neuroscience*, *38*(16), 4020–4030. Publisher: Society for Neuroscience Section: Research Articles.

Stephan, K. E., Baldeweg, T., & Friston, K. J. (2006). Synaptic Plasticity and Dysconnection in Schizophrenia. *Biological Psychiatry*, *59*(10), 929–939.

Stephan, K. E., & Mathys, C. (2014). Computational approaches to psychiatry. *Current Opinion in Neurobiology*, *25*, 85–92. Theoretical and computational neuroscience.

Stephan, K. E., Petzschner, F. H., Kasper, L., Bayer, J., Wellstein, K. V., Stefanics, G., Pruessmann, K. P., & Heinzle, J. (2019). Laminar fMRI and computational theories of brain function. *NeuroImage*, *197*(August 2017), 699–706.

Sterzer, P., Adams, R. A., Fletcher, P., Frith, C., Lawrie, S. M., Muckli, L., Petrovic, P., Uhlhaas, P., Voss, M., & Corlett, P. R. (2018). The predictive coding account of psychosis. *Biological psychiatry*, *84*(9), 634–643.

Suthaharan, P., Reed, E. J., Leptourgos, P., Kenney, J. G., Uddenberg, S., Mathys, C. D., Litman, L., Robinson, J., Moss, A. J., Taylor, J. R., Groman, S. M., & Corlett, P. R. (2021). Paranoia and belief updating during the COVID-19 crisis. *Nature Human Behaviour*, *5*(9), 1190–1202. Number: 9 Publisher: Nature Publishing Group.

Thestrup Waade, P., Møller, A. H., Comoglio, J., Mikus, N., Legrand, N., Stephan, K. E., Weber, L., & Mathys, C. (in prep). The Generalized Hierarchical Gaussian Filter in Julia.

Vossel, S., Mathys, C., Stephan, K. E., & Friston, K. J. (2015). Cortical Coupling Reflects Bayesian Belief Updating in the Deployment of Spatial Attention. *Journal of Neuroscience*, *35*(33), 11532–11542. Publisher: Society for Neuroscience Section: Articles.

Walsh, K. S., McGovern, D. P., Clark, A., & O'Connell, R. G. (2020). Evaluating the neurophysiological evidence for predictive processing as a model of perception. *Annals of the New York Academy of Sciences*, *1464*(1), 242–268.

Weber, L. A. (2020). *Perception as Hierarchical Bayesian Inference-Toward non-invasive readouts of exteroceptive and interoceptive processing*. Ph.D. thesis, ETH Zurich. URL: https://doi.org/10.3929/ethz-b-000476505.

Weber, L. A., Diaconescu, A. O., Mathys, C., Schmidt, A., Kometer, M., Vollenweider, F., & Stephan, K. E. (2020). Ketamine affects prediction errors about statistical regularities: a computational single-trial analysis of the mismatch negativity. *Journal of Neuroscience*, *40*(29), 5658–5668.

Weber, L. A., Tomiello, S., Schöbi, D., Wellstein, K. V., Mueller, D., Iglesias, S., & Stephan, K. E. (2022). Auditory mismatch responses are differentially sensitive to changes in muscarinic acetylcholine versus dopamine receptor function. *eLife*, *11*, e74835.

Weilnhammer, V. A., Stuke, H., Sterzer, P., & Schmack, K. (2018). The Neural Correlates of Hierarchical Predictions for Perceptual Decisions. *Journal of Neuroscience*, *38*(21), 5008–5021. Publisher: Society for Neuroscience Section: Research Articles.

5 Appendix

5.1 Approximate inversion for value coupling

In Mathys et al. (2011), we presented a variational approximation to the exact Bayesian inversion of our generative model which employed a mean-field approximation, and derived analytic one-step update equations using a new quadratic approximation to the variational energies. Following this procedure for the case of value coupling, we specify here the variational energy for a value parent to derive the update equations presented in the main text.

The generative model for a state x_a with a (non)linear value parent x_b (and a volatility parent $\check{x}_a$) is given by[4]

$$x_a^{(k)} \sim \mathcal{N}\left(x_a^{(k-1)} + \alpha_{b,a} g\left(x_b^{(k)}\right), \exp\left(\kappa_{\check{a},a}\check{x}_a^{(k)} + \omega_a\right)\right), \tag{45}$$

where the value coupling between x_a and x_b is mediated by function g, which can be nonlinear.

Using the mean-field approximation as in Mathys et al. (2011), the variational energy and its first two derivatives for the value parent x_b are given by

$$\begin{aligned} I\left(x_b^{(k)}\right) =& -\frac{1}{2}\hat{\pi}_a^{(k)}\left(\frac{1}{\pi_a^{(k)}} + \left(\mu_a^{(k)} - \left(\mu_a^{(k-1)} + g\left(x_b^{(k)}\right)\right)\right)^2\right) \\ & -\frac{1}{2}\hat{\pi}_b^{(k)}\left(x_b^{(k)} - \mu_b^{(k-1)}\right)^2 + const. \end{aligned} \tag{46}$$

$$\begin{aligned} I'\left(x_b^{(k)}\right) =& \hat{\pi}_a^{(k)} g'\left(x_b^{(k)}\right)\left(\mu_a^{(k)} - \left(\mu_a^{(k-1)} + g\left(x_b^{(k)}\right)\right)\right) \\ & - \hat{\pi}_b^{(k)}\left(x_b^{(k)} - \mu_b^{(k-1)}\right) \end{aligned} \tag{47}$$

$$\begin{aligned} I''\left(x_b^{(k)}\right) =& \hat{\pi}_a^{(k)}\left(g''\left(x_b^{(k)}\right)\left(\mu_a^{(k)} - \left(\mu_a^{(k-1)} + g\left(x_b^{(k)}\right)\right)\right)\right) \\ & - g'\left(x_b^{(k)}\right)^2 - \hat{\pi}_b^{(k)} \end{aligned} \tag{48}$$

We calculate the mean and precision of the Gaussian posterior for the value parent $x_b^{(k)}$ using the rules as stated in Mathys et al. (2011) (equations 38 and 40 therein), which follow a quadratic approximation to the variational energy with expansion point at the posterior belief $\mu_b^{(k-1)}$ from the previous trial. For this, we need the derivatives of the variational energies at this point:

$$I'\left(\mu_b^{(k-1)}\right) = \hat{\pi}_a^{(k)} g'\left(\mu_b^{(k-1)}\right)\left(\mu_a^{(k)} - \left(\mu_a^{(k-1)} + g\left(\mu_b^{(k-1)}\right)\right)\right) \tag{49}$$

Here, we identify the prediction of the mean $\hat{\mu}_a^{(k)}$ about the value child state x_a as

$$\hat{\mu}_a^{(k)} = \mu_a^{(k-1)} + g\left(\mu_b^{(k-1)}\right) \tag{50}$$

and thus the prediction error about x_a as

$$\delta_a^{(k)} = \mu_a^{(k)} - \left(\mu_a^{(k-1)} + g\left(\mu_b^{(k-1)}\right)\right). \tag{51}$$

Therefore, the first derivative of the variational energy becomes

$$I'\left(\mu_b^{(k-1)}\right) = \hat{\pi}_a^{(k)} g'\left(\mu_b^{(k-1)}\right)\delta_a^{(k)}. \tag{52}$$

[4] Note that for brevity, we are omitting all priors here - strictly speaking, these equations only form a generative model if combined with appropriate priors on the model parameters and the initial states.

Similarly, the second derivative then reads:

$$I''\left(\mu_b^{(k-1)}\right) = \hat{\pi}_a^{(k)}\left(g''\left(\mu_b^{(k-1)}\right)\delta_a^{(k)} - g'\left(\mu_b^{(k-1)}\right)^2\right) - \hat{\pi}_b^{(k)} \tag{53}$$

With these, we can specify the update equations for the precision π_b and the mean μ_b of the value parent (see Mathys et al. (2011) and Appendix B of Mathys et al. (2014)):

$$\begin{aligned} \pi_b^{(k)} &= -I''\left(\mu_b^{(k-1)}\right) \\ &= \hat{\pi}_b^{(k)} + \hat{\pi}_a^{(k)}\left(g'\left(\mu_b^{(k-1)}\right)^2 - g''\left(\mu_b^{(k-1)}\right)\delta_a^{(k)}\right) \end{aligned} \tag{54}$$

$$\begin{aligned} \mu_b^{(k)} &= \hat{\mu}_b^{(k)} + \frac{I'\left(\mu_b^{(k-1)}\right)}{\pi_b^{(k)}} \\ &= \hat{\mu}_b^{(k)} + \frac{\hat{\pi}_a^{(k)} g'\left(\mu_b^{(k-1)}\right)}{\pi_b^{(k)}}\delta_a^{(k)} \end{aligned} \tag{55}$$

If $g(x) = x$ (linear value coupling), then $g'(x) = 1$ and $g''(x) = 0$, and we end up with the update equations specified in section 3.

5.2 Definition of a VOPE

In the main text, we introduced a new definition of the volatility prediction error or VOPE Δ, which we express as a function of the previously defined value prediction error, or VAPE, δ. Here, we show how our new definition derives from the definition presented in earlier work (Mathys et al., 2011):

$$\begin{aligned} \Delta_a^{(k)} \equiv \delta_a^{(k,VOPE)} &= \frac{\frac{1}{\pi_a^{(k)}} + \left(\mu_a^{(k)} - \hat{\mu}_a^{(k)}\right)^2}{\frac{1}{\pi_a^{(k-1)}} + \Omega_a^{(k)}} - 1 \\ &= \hat{\pi}_a^{(k)}\left(\frac{1}{\pi_a^{(k)}} + \left(\mu_a^{(k)} - \hat{\mu}_a^{(k)}\right)^2\right) - 1 \\ &= \hat{\pi}_a^{(k)}\left(\frac{1}{\pi_a^{(k)}} + \left(\delta_a^{(k)}\right)^2\right) - 1 \\ &= \frac{\hat{\pi}_a^{(k)}}{\pi_a^{(k)}} + \hat{\pi}_a^{(k)}\left(\delta_a^{(k)}\right)^2 - 1. \end{aligned} \tag{56}$$

Note that from the first to the second line, we have used the following definition:

$$\hat{\pi}_a^{(k)} = \frac{1}{\frac{1}{\pi_a^{(k-1)}} + \Omega_a^{(k)}}.$$

This ensures that a given node does not need to have access to the posterior precision from the level below: $\pi_a^{(k-1)}$, which facilitates implementation.

In sum, we are introducing a second prediction error unit Δ which is concerned with deviations from predicted uncertainty and is informed by value prediction errors and other estimates of uncertainty. It is this prediction error - a function of the unweighted (squared) value prediction error with a new precision weight - which communicates between a node and its volatility parent.

5.3 The case of multiple parent nodes

Let's assume that node a has two volatility parents, nodes b and c. In addition to the overall prediction given in equation 28, it would also compute a parent-specific volatility expectation:

$$\Omega_{a,b}^{(k)} = t^{(k)} \exp\left(\omega_a + \kappa_{b,a}\mu_b^{(k-1)}\right) \tag{57}$$

$$\Omega_{a,c}^{(k)} = t^{(k)} \exp\left(\omega_a + \kappa_{c,a}\mu_c^{(k-1)}\right), \tag{58}$$

and based on that a parent-specific prediction:

$$\hat{\pi}_{a,b}^{(k)} = \frac{1}{\frac{1}{\pi_a^{(k-1)}} + \Omega_{a,b}^{(k)}} \tag{59}$$

$$\gamma_{a,b}^{(k)} = \Omega_{a,b}^{(k)}\hat{\pi}_{a,b}^{(k)} \tag{60}$$

$$\hat{\pi}_{a,c}^{(k)} = \frac{1}{\frac{1}{\pi_a^{(k-1)}} + \Omega_{a,c}^{(k)}} \tag{61}$$

$$\gamma_{a,c}^{(k)} = \Omega_{a,c}^{(k)}\hat{\pi}_{a,c}^{(k)}. \tag{62}$$

While the node's posterior would still be based on the overall prediction (eq. 28), in the PE step, the node will now compute separate prediction errors for each of its parents:

$$\Delta_{a,b}^{(k)} = \frac{\hat{\pi}_{a,b}^{(k)}}{\pi_a^{(k)}} + \hat{\pi}_{a,b}^{(k)}\left(\delta_a^{(k)}\right)^2 - 1 \tag{63}$$

$$\Delta_{a,c}^{(k)} = \frac{\hat{\pi}_{a,c}^{(k)}}{\pi_a^{(k)}} + \hat{\pi}_{a,c}^{(k)}\left(\delta_a^{(k)}\right)^2 - 1, \tag{64}$$

and each of the parent nodes will be be updated based on the child node's parent-specific prediction errors and precision weights:

$$\pi_b^{(k)} = \hat{\pi}_b^{(k)} + \frac{1}{2}\left(\kappa_{b,a}\gamma_{a,b}^{(k)}\right)^2 + \left(\kappa_{b,a}\gamma_{a,b}^{(k)}\right)^2\Delta_{a,b}^{(k)} - \frac{1}{2}\kappa_{b,a}^2\gamma_{a,b}^{(k)}\Delta_{a,b}^{(k)} \tag{65}$$

$$\mu_b^{(k)} = \hat{\mu}_b^{(k)} + \frac{1}{2}\frac{\kappa_{b,a}\gamma_{a,b}^{(k)}}{\pi_b^{(k)}}\Delta_{a,b}^{(k)}, \tag{66}$$

and accordingly for node c.

The same logic applies to the case where node a has multiple value parents - in this case, the parent-specific quantities will be $P_{a,vapa}$, $\hat{\mu}_{a,vapa}$, and $\delta_{a,vapa}$. Note that the child node always has to compute both the parent-specific predictions, prediction errors and precision weights, as well as the overall predictions, prediction errors and precision weights. For example, if node a has multiple volatility parents, it still need to compute the overall precision of its prediction $\hat{\pi}_a$ that is based on all volatility parents, as this will be needed to compute its own posterior in the update step, and to communicate with any value parents it might have. Similarly, for communication with volatility parents, an overall VAPE δ_a is needed to compute the VOPE(s).

5.4 Computations of input nodes

Input nodes differ from regular HGF nodes in that the input does not perform a Gaussian random walk in time, but instead is generated by the lowest state in the hierarchy on each trial (see section 2 of the main text), and potentially corrupted by additional noise. Note that we refer to the sensory stimuli that enter the HGF network as ”inputs” (and thus call the receiving nodes input nodes), but from the perspective of the generative model, these are the observable ”outputs” of the network of interacting hidden states.

The input nodes are important elements in the HGF belief network. The main steps that need to happen within an input node on a given trial are:

- receive a new input and store it
- either receive as a second input the exact time of the input, or infer the time as ’plus 1’ (next trial)
- compute prediction errors and whatever else needs to be signalled bottom-up to the first actual HGF node
- send bottom-up: usually input or PE, some estimate of precision, and time
- receive top-down: usually $\hat{\mu}$ from the parent
- compute surprise, given input and prediction

The quantities being signalled bottom-up, and the computation of surprise, depend on the nature of the input node (continuous or binary) and on the nature of the coupling with the parent.

Because the input nodes are not full HGF nodes, but rather serve as a relay station for the input and for computing surprise, and because they capture any observation noise that might be inherent in the input, the message passing (and the within-node computations) differs from the generic scheme presented in the main text.

Continuous input nodes

A continuous input node serves to receive inputs that live on a continuous scale. In terms of the generative model, we think of them as being sampled from a normal distribution with a mean – which is determined by the HGF node it is coupled to–, and a variance – which can be either constant, or additionally driven by another HGF node. This variance is what we call the observation noise.

In analogy to the coupling types introduced in the main text, we call the HGF node that tracks the input’s mean its value parent. However, the HGF node which represents the phasic component of the input’s variance (or observation noise), is not a volatility parent, since the input does not perform any movement in time by itself, thus has no volatility. Instead, we call this HGF node the input node’s noise parent. Any continuous input node will have a value parent, but having a noise parent is optional (in the absence of it, the observation noise on every trial will be determined by a constant parameter of the input node).

Value parents of continuous input nodes

If one wanted to construct the same computational steps for input nodes as for regular HGF nodes, we would again start with a the prediction step for the current trial. Here, the input node would have to compute the predicted mean $\hat{\mu}_{inp}^{(k)}$ and the precision of the prediction $\hat{\pi}_{inp}^{(k)}$ for the current trial. However, the predicted mean of an input node is simply the prediction from the value parent:

$$\hat{\mu}_{inp}^{(k)} = \hat{\mu}_{vapa}^{(k)} \tag{67}$$

Given that the mean parent might operate under a drift parameter, the current prediction can only be computed once the new input has arrived. Then it needs to be signalled top-down immediately.

In the absence of a noise parent, the precision of the prediction for the input node is fully determined the input node's noise parameter ζ:

$$\hat{\pi}_{inp}^{(k)} = \frac{1}{\exp\left(\zeta_{inp}\right)} \tag{68}$$

However, in the presence of a noise parent, this will additionally depend on the posterior $\mu_{nopa}^{(k-1)}$ of that parent on the previous trial, and the coupling parameter $\kappa_{nopa,inp}$ of the input node with its noise parent *nopa*:

$$\hat{\pi}_{inp}^{(k)} = \frac{1}{\exp\left(\kappa_{nopa,inp}\mu_{nopa}^{(k-1)} + \zeta_{inp}\right)}. \tag{69}$$

In the update step, the posterior mean of the input node simply amounts to the input itself, while the posterior precision will be determined by the value parent's posterior precision:

$$\mu_{inp}^{(k)} = u^{(k)} \tag{70}$$

$$\pi_{inp}^{(k)} = \pi_{vapa}^{(k)} \tag{71}$$

Finally, in the PE step, the value PE (or VAPE) will be computed as the difference the prediction and the posterior:

$$\delta_{inp}^{(k)} = \mu_{inp}^{(k)} - \hat{\mu}_{inp}^{(k)} = u^{(k)} - \hat{\mu}_{inp}^{(k)} \tag{72}$$

This means that prior to the update of the input node, it needs to receive the current prediction $\hat{\mu}_{vapa}^{(k)}$ of its parent node.

The update of the value parent node will look like the regular value coupling updates from previous chapters:

$$\pi_{vapa}^{(k)} = \hat{\pi}_{vapa}^{(k)} + \hat{\pi}_{inp}^{(k)} \tag{73}$$

$$\mu_{vapa}^{(k)} = \hat{\mu}_{vapa}^{(k)} + \frac{\hat{\pi}_{inp}^{(k)}}{\pi_{vapa}^{(k)}}\delta_{inp}^{(k)} \tag{74}$$

This means that the input node needs to signal bottom-up to its mean parent:

Predicted precision: $\hat{\pi}_{inp}^{(k)}$

Prediction error: $\delta_{inp}^{(k)}$.

Note that the connection between a continuous input node and its value parent always has a connection weight of $\alpha = 1$.

Constructing the computational steps for the input node in this way, though it might seem slightly artificial (as the actual belief about the input is represented in the parent node), allows for a more modular implementation, where the value parent node does not have to "be aware" of whether its child is another regular HGF node or a continuous input node (as it receives the same kinds of messages from both).

Finally, to compute the surprise associated with the current input, the node needs to compute the negative log of the probability of input $u^{(k)}$ under a Gaussian prediction with $\hat{\mu}_{inp}^{(k)}$ as mean and $\hat{\pi}_{inp}^{(k)}$ as the precision:

$$-\log\left(p\left(u^{(k)}\right)\right)=\frac{1}{2}\left(\log\left(2\pi\right)-\log\left(\hat{\pi}_{inp}^{(k)}\right)+\hat{\pi}_{inp}^{(k)}\left(u^{(k)}-\hat{\mu}_{vapa}^{(k)}\right)^{2}\right). \tag{75}$$

Noise parents of continuous input nodes
Having a noise parent for a continuous input node means that a noise PE (or NOPE) will be computed and signalled bottom-up during the PE step. We denote this PE with the symbol ε_{inp}. Importantly, the NOPE (as opposed to the VOPE) is not a direct function of the VAPE. Instead, both the posterior precision as well as the posterior mean are taken from the value parent *vapa*:

$$\varepsilon_{inp}^{(k)}=\frac{\hat{\pi}_{inp}^{(k)}}{\pi_{vapa}^{(k)}}+\hat{\pi}_{i}^{(k)}\left(u^{(k)}-\mu_{vapa}^{(k)}\right)^{2}-1. \tag{76}$$

This in turn requires that the update of the value parent happens before the computation of the NOPE, and the posterior of the value parent is already available to the input node.

The update step for the noise parent is similar to the update in volatility parents (equations 38) with a modified prediction error and an effective precision term γ_{inp} fixed to 1. Hence the update of the mean reads:

$$\begin{aligned}\mu_{nopa}^{(k)}&=\hat{\mu}_{nopa}^{(k)}+\frac{1}{2}\frac{\kappa_{nopa,inp}\gamma_{inp}^{(k)}}{\pi_{nopa}^{(k)}}\varepsilon_{inp}^{(k)} &(77)\\ &=\hat{\mu}_{nopa}^{(k)}+\frac{1}{2}\frac{\kappa_{nopa,inp}}{\pi_{nopa}^{(k)}}\varepsilon_{inp}^{(k)}. &(78)\end{aligned}$$

This similarity again means that the parent node does not necessarily have to be "aware" of whether it serves as a volatility or a noise parent - as long as the input node also signals a value of 1 as the effective precision term γ on every trial. Importantly, this also works for the update of the precision of the noise parent. Setting γ_{inp} to 1 (and replacing the VOPE Δ with the NOPE ε and *vopa* with *nopa*) in the previously established precision update for volatility parents (equation 38) leads to:

$$\begin{aligned}\pi_{nopa}^{(k)}&=\hat{\pi}_{nopa}^{(k)}+\frac{1}{2}\left(\kappa_{nopa,inp}\gamma_{inp}^{(k)}\right)^{2}+\left(\kappa_{nopa,inp}\gamma_{inp}^{(k)}\right)^{2}\varepsilon_{inp}^{(k)}-\frac{1}{2}\kappa_{nopa,inp}^{2}\gamma_{inp}^{(k)}\varepsilon_{inp}^{(k)} &(79)\\ &=\hat{\pi}_{nopa}^{(k)}+\frac{1}{2}\left(\kappa_{nopa,inp}\right)^{2}+\left(\kappa_{nopa,inp}\right)^{2}\varepsilon_{inp}^{(k)}-\frac{1}{2}\kappa_{nopa,inp}^{2}\varepsilon_{inp}^{(k)} &(80)\\ &=\hat{\pi}_{nopa}^{(k)}+\frac{1}{2}\left(\kappa_{nopa,inp}\right)^{2}+\frac{1}{2}\left(\kappa_{nopa,inp}\right)^{2}\varepsilon_{inp}^{(k)} &(81)\\ &=\hat{\pi}_{nopa}^{(k)}+\frac{1}{2}\left(\kappa_{nopa,inp}\right)^{2}\left(1+\varepsilon_{inp}^{(k)}\right), &(82)\end{aligned}$$

which is exactly the update the noise parent needs.

Peculiarities of continuous input nodes and consequences for their parents
First, due to the dependence of the NOPE on the posterior beliefs of the mean parent, the continuous input node needs to communicate with its value parent first and wait for the posteriors to be computed there and sent top-down in order to elicit a new update in its noise parent.

Second, the value parent needs to send top-down not only the posterior mean, but also the posterior precision, for the same reason.

Third, the connection weight for value connections will always be $\alpha = 1$.

Fourth, for issuing a new prediction $\hat{\mu}_{inp}$, the node needs to receive the predicted mean of its value parent at the beginning of a new trial. This means it must be possible to elicit a new prediction in regular hgf nodes without actually sending a prediction error, instead by only sending a new time point. The hgf node needs to react to this by sending top-down the new predicted mean, such that the input node can compute the PE and signal it back bottom-up for a new update.

Thus, the steps for a continuous input node are:

- receive input u
- determine time of input
- send bottom-up to value parent: time of input (to elicit a prediction)
- receive top-down: predicted mean $\hat{\mu}_{vapa}$
- compute prediction $\hat{\mu}_{inp}$ and retrieve $\hat{\pi}_{inp}$
- compute surprise using u, $\hat{\mu}_{inp}$ and $\hat{\pi}_{inp}$
- compute VAPE using u and $\hat{\mu}_{inp}$
- send bottom-up to value parent: VAPE, $\hat{\pi}_{inp}$, and time
- receive top-down: posteriors μ_{vapa} and π_{vapa}
- if relevant, compute NOPE using u, $\hat{\pi}_{inp}$, μ_{vapa} and π_{vapa}
- send bottom-up to noise parent: NOPE, $\gamma_{inp} = 1$, and time
- receive top-down: posterior μ_{nopa}
- compute new precision of its prediction $\hat{\pi}_{inp}^{(k+1)}$ using its tonic observation noise ζ and, if present, the posterior mean of its noise parent $\mu_{nopa}^{(k)}$.

The value parent of a continuous input node needs to

1. be able to elicit new predictions based on time input
2. send new predictions top-down immediately in the case of a continuous input node child
3. send down not only its posterior mean, but also the precision after each update.

Binary input nodes

Binary input nodes serve to receive inputs that can only take on one of two values. These input nodes can only have value parents. These value parents in turn are binary HGF nodes, which are special cases of HGF nodes, which themselves only have value parents. Thus we here aim for an implementation where the value parents of binary HGF nodes are regular HGF nodes (and do not need to be aware of whether their child node is a regular HGF node, a binary HGF node, or a continuous input node.

For binary input nodes, the observation noise is given by their noise parameter ζ. Therefore, the precision of the input prediction $\hat{\pi}_{inp}$ is constant (i.e., we can treat it as a parameter). We distinguish two cases: Either the noise is zero: $\zeta_{inp} = 0$ (or precision is infinite, $\hat{\pi}_{inp} = \inf$), or there is noise (and the precision has a finite value).

In general, the steps for a binary input node are:

- receive input u
- determine time of input
- compute prediction errors, if necessary
- send bottom-up: u or two prediction errors, input precision, and time
- receive top-down: prediction of parent $\hat{\mu}_{pa}$
- compute surprise based on message from parent.

The bottom-up messages and the surprise computation depend on the distinction between infinite and finite precision. We will now lay these out for the two cases.

Infinite precision
This case is very simple. If $\hat{\pi}_{inp}$ is infinite, then the bottom-up messages are simply this precision $\hat{\pi}_{inp}$ itself, and u. The surprise computation is also very simple:

$$surprise^{(k)} = \begin{cases} -\log\left(1 - \hat{\mu}_{pa}^{(k)}\right), & \text{for } u^{(k)} = 1 \\ -\log\left(\hat{\mu}_{pa}^{(k)}\right), & \text{for } u^{(k)} = 0. \end{cases} \tag{83}$$

Finite precision
Here, the input u is actually not binary, but a real number whose distribution is a mixture of Gaussians. If $x_1 = 1$, the probability of u is normally distributed with constant variance ζ_{inp} around a constant value η_0, corresponding to the most likely sensation if $x_1 = 1$. If, however, $x_1 = 0$, the most likely sensation is η_0 with the probability of u normally distributed with the same variance.

The messages to be sent bottom-up will in this case again be the precision $\hat{\pi}_{inp}$, which is determined by the observation noise:

$$\hat{\pi}_{inp} = \frac{1}{\exp\left(\zeta_{inp}\right)}, \tag{84}$$

along with two prediction errors, namely the deviations of the input u from both possible values η_a and η_b:

$$\delta_{inp,1}^{(k)} = u^{(k)} - \eta_1 \tag{85}$$

$$\delta_{inp,0}^{(k)} = u^{(k)} - \eta_0. \tag{86}$$

Additionally, as always, the nodes need to send up the information about the time of the input.

For surprise computation, the node depends again on receiving the prediction of the parent node $\hat{\mu}_{pa}^{(k)}$, and the two η values:

$$surprise^{(k)} = -\log\left(\hat{\mu}_{pa}^{(k)} * \mathcal{N}\left(u^{(k)}; \eta_1, \hat{\pi}_{inp}\right) + \left(1 - \hat{\mu}_{pa}^{(k)}\right) * \mathcal{N}\left(u^{(k)}; \eta_0, \hat{\pi}_{inp}\right)\right) \tag{87}$$

The special cases that follow for the update of the parent node are restricted to binary HGF nodes, which therefore represent their own special case of HGF nodes.

Binary HGF nodes

Binary nodes are parents of binary input nodes. Their cycle thus starts with receiving a bottom-up message from their child node, which, first of all, needs to trigger the prediction step. Similarly to continuous input nodes, the predictions of a binary HGF node depend on its parent's predictions:

$$\hat{\mu}_{bin}^{(k)} = \frac{1}{1+\exp\left(-\hat{\mu}_{pa}^{(k)}\right)} \tag{88}$$

$$\hat{\pi}_{bin}^{(k)} = \frac{1}{\hat{\mu}_{bin}^{(k)} * \left(1-\hat{\mu}_{bin}^{(k)}\right)}. \tag{89}$$

Note that the precision of the prediction is a direct function of the mean due to the binary nature of the state.

Again, we need to introduce an additional top-down signalling step at the beginning of each trial, where the parent node sends down its current prediction of the mean, given the time of a new input - as we have already noted for the communication with continuous input nodes.

The bottom-up message that binary HGF nodes receive will, depending on the precision of the child binary input node, be comprised of 3 ($\hat{\pi}_{inp}$, $u^{(k)}$, and the time of the input), or 4 quantities ($\hat{\pi}_{inp}$, $\delta_{inp,1}^{(k)}$, $\delta_{inp,0}^{(k)}$, and the time of the input). Consequently, we have to distinguish two cases for the update step. In case 1 (3 messages), the updates read:

$$\mu_{bin}^{(k)} = u^{(k)} \tag{90}$$

$$\pi_{bin}^{(k)} = \hat{\pi}_{inp} \tag{91}$$

if

$$\hat{\pi}_{inp} = \text{inf}, \tag{92}$$

and in case 3 (4 messages), they read:

$$\mu_{bin}^{(k)} = \frac{\hat{\mu}_{bin}^{(k)} * \exp\left(-\frac{1}{2}\hat{\pi}_{inp}\left(\delta_{inp,1}\right)^2\right)}{\hat{\mu}_{bin}^{(k)} * \exp\left(-\frac{1}{2}\hat{\pi}_{inp}\left(\delta_{inp,1}\right)^2\right) + \left(1-\hat{\mu}_{bin}^{(k)}\right) * \exp\left(-\frac{1}{2}\hat{\pi}_{inp}\left(\delta_{inp,0}\right)^2\right)} \tag{93}$$

$$\pi_{bin}^{(k)} = \frac{1}{\hat{\mu}_{bin}^{(k)} * \left(1-\hat{\mu}_{bin}^{(k)}\right)} \tag{94}$$

if

$$\hat{\pi}_{inp} \neq \text{inf}\,. \tag{95}$$

Finally, in the PE step, the binary node computes a VAPE for its parent node (which is always a value parent):

$$\delta_{bin}^{(k)} = \mu_{bin}^{(k)} - \hat{\mu}_{bin}^{(k)}. \tag{96}$$

The parent node will also perform its update step, but with two differences compared to the normal value update in HGF nodes. First, the precision of the prediction of the binary HGF node will enter the precision update in an unusual way:

$$\pi_{pa}^{(k)} = \hat{\pi}_{pa}^{(k)} + \frac{1}{\hat{\pi}_{bin}^{(k)}}. \tag{97}$$

Second, the VAPE will not be precision-weighted by the low-level precision:

$$\mu_{pa}^{(k)} = \hat{\mu}_{pa}^{(k)} + \frac{1}{\pi_{pa}^{(k)}}\delta_{bin}^{(k)}. \tag{98}$$

For the implementation, this means that we can either give the HGF node knowledge about who its child is and let the exact update depend on that, or we can let this be solved by the value connection, in which case this connection would need to signal the precision weight that is used for the mean update separately from the term that is used for the precision update.

In any case, the information which needs to be sent bottom-up from a binary HGF node to its value parent is:

Prediction error: $\delta_{bin}^{(k)}$

Predicted precision: $\hat{\pi}_{bin}^{(k)}$

In case the parent does not have knowledge about its child, the information would have to be sent in the following form:

Prediction error: $\delta_{bin}^{(k)}$

Precision weight for mean update: 1

Precision term for precision update: $\frac{1}{\hat{\pi}_{bin}^{(k)}}$

Summary: Implementational consequences

Due to the special cases of continuous input nodes and binary HGF nodes, which both can be potential children of regular HGF nodes, we need to introduce a few changes to the update and connection logic of regular HGF nodes:

- HGF nodes need to elicit new predictions if prompted for by receiving information about the time of the new input, and send this prediction top-down. This is needed both for the computation of surprise in the continuous input nodes, but also for the computation of prediction error in continuous input nodes, and for the computation of predictions in binary HGF nodes.

- For the case of value parents of continuous input nodes, HGF nodes need to signal top-down not only their posterior mean, but also their posterior precision.

- Value-coupling connections need to separately bottom-up signal the precision weight of the upcoming prediction error, and the precision term needed to update the parent's precision.

- Implementing noise and volatility connections in the same way allows for an implementation where regular HGF nodes are completely unaware about which kind of node their child is. The computation necessary for the precision update, which is more elaborate in volatility and noise coupling, will be part of the connection logic.

Everything else that is unusual about the computations within binary input nodes, continuous input nodes, and binary HGF nodes will be implemented within these nodes and not affect the regular HGF node.