Message Passing in the Hierarchical Gaussian Filter

November 6, 2018

1 The generative model of the HGF: Volatility vs. value coupling

In the generative model of the HGF, (hidden) states of the world perform Gaussian random walks in time and can produce outcomes which are perceived by an observer as inputs. States can influence each other via volatility coupling or via value coupling.

In the classical 3-level binary HGF as presented in [3], the two states of interest, x_2 and x_3, are coupled to each other via volatility coupling, which means that for state x_2, the mean of the Gaussian random walk on trial k is given by its previous value $x_2^{(k-1)}$, while the step size (or variance) depends on the current value of the higher level state, $x_3^{(k)}$:

$$x_2^{(k)} \sim \mathcal{N}(x_2^{(k)}|x_2^{(k-1)},\, f(x_3^{(k)})), \tag{1}$$

where the exact dependency is of the form

$$f(x_3^{(k)}) = \exp(\kappa_2 x_3^{(k)} + \omega_2). \tag{2}$$

However, a higher-level state can also have influence on a lower-level state by influencing its mean. In that case, the mean of the Gaussian random walk at one level is a function not only of its own previous value, but also the current value of the higher-level state (with step size either constant or a function of another state):

$$x_2^{(k)} \sim \mathcal{N}(x_2^{(k)}|x_2^{(k-1)} + \alpha_{4,2} x_4^{(k)},\, \exp(\omega_2)), \tag{3}$$

which means constant step size, or

$$x_2^{(k)} \sim \mathcal{N}(x_2^{(k)}|x_2^{(k-1)} + \alpha_{4,2} x_4^{(k)},\, \exp(\kappa_2 x_3^{(k)} + \omega_2)). \tag{4}$$

In other words, any given state in the world can be modelled as having a volatility parent state, a value parent state, or both, or none (in which case it evolves as a Gaussian random walk around its previous value with fixed

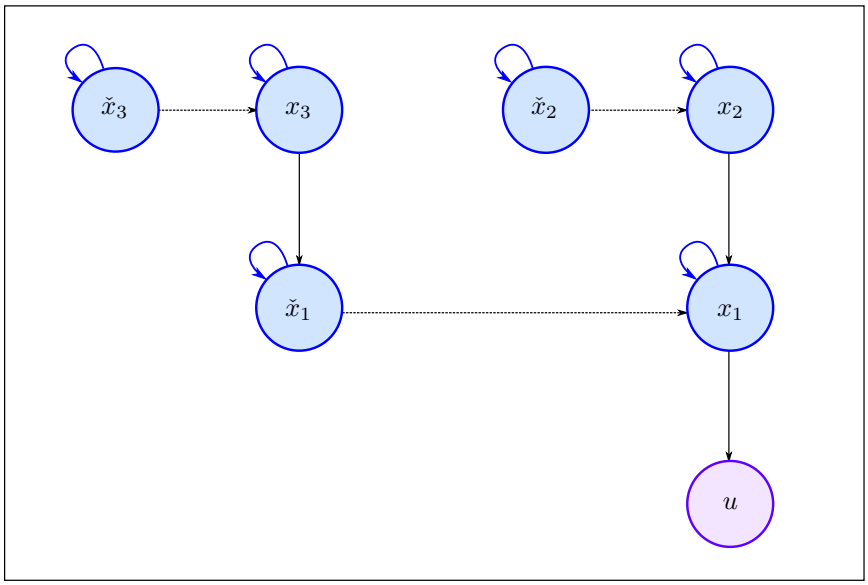

Figure 1: An example of a generative model of sensory inputs with six hidden states. Volatility coupling is depicted with dashed lines, value coupling with straight lines.

step size). Consequently, when inferring on the evolution of these states, the exact belief update equations (which include the computation of new predictions, posterior values, and prediction errors, and represent an approximate inversion of this generative model, see [3]) depend on the nature of the coupling of a given state with its parent and children states. In particular, the nodes that implement the belief updates will communicate with their value parents via value prediction errors, or VAPEs, and via volatility prediction errors, or VOPEs, with their volatility parents.

In figure 1 we have drawn an example setup with six different environmental states and one outcome. Here, we have denoted states that function as value parents for other states as x_i, and states that function as volatility parents as $\check{x}_i$. Volatility coupling is depicted by curvy arrows, value coupling by straight arrows, and observable outcomes are linked to their hidden states via double arrows.

For the example illustrated in figure 1, the following equations describe the generative model:

$$u^{(k)} \sim \mathcal{N}(u^{(k)}|x_1^{(k)}, \qquad \sigma_u) \tag{5}$$
$$x_1^{(k)} \sim \mathcal{N}(x_1^{(k)}|x_1^{(k-1)} + \alpha_{2,1}x_2^{(k)}, \qquad \exp(\kappa_1\check{x}_1^{(k)} + \omega_1)) \tag{6}$$
$$\check{x}_1^{(k)} \sim \mathcal{N}(\check{x}_1^{(k)}|\check{x}_1^{(k-1)} + \alpha_{3,\check{1}}x_3^{(k)}, \qquad \exp(\omega_{\check{1}})) \tag{7}$$
$$x_2^{(k)} \sim \mathcal{N}(x_2^{(k)}|x_2^{(k-1)}, \qquad \exp(\kappa_2\check{x}_2^{(k)} + \omega_2)) \tag{8}$$
$$\check{x}_2^{(k)} \sim \mathcal{N}(\check{x}_2^{(k)}|\check{x}_2^{(k-1)}, \qquad \exp(\omega_{\check{2}})) \tag{9}$$
$$x_3^{(k)} \sim \mathcal{N}(x_3^{(k)}|x_3^{(k-1)}, \qquad \exp(\kappa_3\check{x}_3^{(k)} + \omega_3)) \tag{10}$$
$$\check{x}_3^{(k)} \sim \mathcal{N}(\check{x}_3^{(k)}|\check{x}_3^{(k-1)}, \qquad \exp(\omega_{\check{3}})). \tag{11}$$

Note that in this example, all states that are value parents of other states (or outcomes) have their own volatility parent, while states that are volatility parents to other nodes either have a value parent (as state $\check{x}_1$), or no parents (as states $\check{x}_2$ and $\check{x}_3$). This is deliberately so, and we will see these two motifs - every state of a hierarchy has its own volatility estimation, and volatility states only have value parents - reappear in the following chapters.

2 Belief updates in the HGF: Computations of nodes

In the approximate inversion of the generative model presented above, [3] derived a set of simple, one-step update equations that represent changes in beliefs about the hidden states specified in the generative model. For each state, a belief is held (and updated for every new input) by the agent and described as a Gaussian distribution, fully characterized by its mean $\mu_i^{(k)}$ and its inverse variance, or precision, $\pi_i^{(k)}$ on a given trial k. We conceptualize each belief as a node in a network, where belief updates involve computations within nodes as well as message passing between nodes. The computations of any node within an experimental trial can be ordered in time as shown in the box:

NODE i AT TRIAL k

(`compute` $\text{prediction}_i^{(k)}$)
$\leftarrow$ `receive` $\text{PE}_{i-1}^{(k)}$ from node_{i-1}

UPDATE step
`compute` $\text{posterior}_i^{(k)}$
given: $\text{PE}_{i-1}^{(k)}$ and $\text{prediction}_i^{(k)}$
$\rightarrow$ `send` $\text{posterior}_i^{(k)}$ to node_{i-1}

PE step
`compute` $\text{PE}_i^{(k)}$
given: $\text{prediction}_i^{(k)}$ and $\text{posterior}_i^{(k)}$
$\rightarrow$ `send` $\text{PE}_i^{(k)}$ to node_{i+1}
$\leftarrow$ `receive` $\text{posterior}_{i+1}^{(k)}$ from node_{i+1}

PREDICTION step
`compute` $\text{prediction}_i^{(k+1)}$
given: $\text{posterior}_i^{(k)}$ and $\text{posterior}_{i+1}^{(k)}$

The exact computations in each step depend on the nature of the coupling (via VAPEs vs. VOPEs) with the parent and children nodes and will be outlined in the following two chapters.

Note that we have placed the PREDICTION step in 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 variants of the HGF the prediction also depends on the time that has passed in between trials, which is something that can only be evaluated once the new input arrives - hence the additional computation of the (current) prediction in the beginning of the trial. Conceptually, it makes most sense to think of the prediction as happening continuously between trials. For implementational purposes, 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.

3 Computations for VAPE coupling

The exact computations of the UPDATE step depend on the nature of the coupling with the child node(s), while both the PE step and the PREDICTION step depend on the coupling with the parent node(s).

3.1 Update Step

If Node i is the value parent of Node $i-1$, then the following update equations apply to Node i:

$$\pi_i^{(k)} = \hat{\pi}_i^{(k)} + \alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}$$
$$\mu_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)} + \hat{\pi}_i^{(k)}} \delta_{i-1}^{(k)}$$

We note here that we can let the update of the precision happen first, and therefore use it for the update of the mean:

$$\pi_i^{(k)} = \hat{\pi}_i^{(k)} + \alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)} \tag{12}$$
$$\mu_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)} \tag{13}$$

In sum, at the time of the update, Node i needs to have access to the following quantities:

Its own predictions: $\hat{\mu}_i^{(k)}$, $\hat{\pi}_i^{(k)}$

Coupling strength: $\alpha_{i-1,i}$

From level below: $\delta_{i-1}^{(k)}$, $\hat{\pi}_{i-1}^{(k)}$

All of these are available at the time of the update. Node i therefore only needs to receive the PE and the predicted precision from the level below to perform its update.

3.2 Prediction Error Step

We will assume in the following, that Node i is the value child of Node $i+1$. Then the following quantities have to be sent up to Node $i+1$ (cf. necessary information from level below in a value parent):

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

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

Node i has already performed the PREDICTION step on the previous trial, so it has already computed the predicted precision of the current trial, $\hat{\pi}_i^{(k)}$. Hence, in the PE step, it needs to perform only the following calculation:

$$\delta_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)} \tag{14}$$

3.3 Prediction Step

Still assuming that Node i is the value child of Node $i+1$, the PREDICTION step consists of the following computations:

$$\hat{\mu}_i^{(k+1)} = \mu_i^{(k)} + \alpha_{i,i+1}\mu_{i+1}^{(k)} \tag{15}$$

$$\hat{\pi}_i^{(k+1)} = \frac{1}{\frac{1}{\pi_i^{(k)}} + \nu_i^{(k+1)}} \tag{16}$$

with

$$\nu_i^{(k+1)} = \exp(\omega_i). \tag{17}$$

Note that if Node i additionally has a VOPE parent node, the estimated volatility $\nu_i^{(k+1)}$ that enters the precision update would also depend on the posterior mean of that volatility parent (cf. PREDICTION step for VOPE coupling).

In general, the prediction of the mean will depend only on whether Node i has a value parent or not, whereas the prediction of the precision only depends on whether Node i has a volatility parent or not.

Thus, the PREDICTION step only depends on knowing the node's own posteriors and receiving the value parent's posterior in time before the new input arrives.

4 Computations for VOPE coupling

As in the case of VAPE coupling, the exact computations of the UPDATE step depend on the nature of the coupling with the child node(s), while both the PE step and the PREDICTION step depend on the coupling with the parent node(s).

To describe the computations entailed by VOPE coupling, we will introduce two changes to the notation. First of all, 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 character δ_i only for VAPEs:

$$\delta_i^{(k)} \equiv \delta_i^{(k,VAPE)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)}, \tag{18}$$

and introduce a new character Δ_i for VOPEs, which we define as

$$\begin{aligned}
\Delta_i^{(k)} \equiv \delta_i^{(k,VOPE)} &= \frac{\frac{1}{\pi_i^{(k)}} + (\mu_i^{(k)} - \hat{\mu}_i^{(k)})^2}{\frac{1}{\pi_i^{(k-1)}} + \nu_i^{(k)}} - 1 \\
&= \hat{\pi}_i^{(k)} \left(\frac{1}{\pi_i^{(k)}} + (\mu_i^{(k)} - \hat{\mu}_i^{(k)})^2 \right) - 1 \\
&= \hat{\pi}_i^{(k)} \left(\frac{1}{\pi_i^{(k)}} + (\delta_i^{(k)})^2 \right) - 1 \\
&= \frac{\hat{\pi}_i^{(k)}}{\pi_i^{(k)}} + \hat{\pi}_i^{(k)} (\delta_i^{(k)})^2 - 1.
\end{aligned} \tag{19}$$

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

$$\hat{\pi}_{i-1}^{(k)} = \frac{1}{\frac{1}{\pi_{i-1}^{(k-1)}} + \nu_{i-1}^{(k)}}.$$

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

In sum, we are introducing a second prediction error unit Δ_i 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 level's nodes and a level's volatility parent's nodes.

Second, we will introduce another quantity, which we term the (auxiliary) expected precision

$$\gamma_i^{(k)} = \nu_i^{(k)} \hat{\pi}_i^{(k)}, \tag{20}$$

which will be computed as part of the PREDICTION step and only serves to simplify the equations and the corresponding message passing.

4.1 Update Step

If Node i is the volatility parent of Node $i-1$, then the following update equations apply to Node i:

$$\begin{aligned}
\pi_i^{(k)} &= \hat{\pi}_i^{(k)} + \frac{1}{2}(\kappa_{i-1}\nu_{i-1}^{(k)}\hat{\pi}_{i-1}^{(k)})^2 * (1 + (1 - \frac{1}{\pi_{i-1}^{(k-1)}\nu_{i-1}^{(k)}})\delta_{i-1}^{(k)}) \\
&= \hat{\pi}_i^{(k)} + \frac{1}{2}(\kappa_{i-1}\nu_{i-1}^{(k)}\hat{\pi}_{i-1}^{(k)})^2 * (1 + (2 - \frac{1}{\hat{\pi}_{i-1}^{(k)}\nu_{i-1}^{(k)}})\delta_{i-1}^{(k)}) \\
\mu_i^{(k)} &= \hat{\mu}_i^{(k)} + \frac{1}{2}\kappa_{i-1}\nu_{i-1}^{(k)}\frac{\hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}}\delta_{i-1}^{(k)},
\end{aligned}$$

where we have again used the definition of the predicted precision $\hat{\pi}_{i-1}^{(k)}$ to derive an expression for the posterior precision from the previous trial $\pi_{i-1}^{(k-1)}$:

$$\begin{aligned}
\hat{\pi}_{i-1}^{(k)} &= \frac{1}{\frac{1}{\pi_{i-1}^{(k-1)}} + \nu_{i-1}^{(k)}} \\
\Leftrightarrow \pi_{i-1}^{(k-1)} &= \frac{1}{\frac{1}{\hat{\pi}_{i-1}^{(k)}} - \nu_{i-1}^{(k)}}.
\end{aligned}$$

With the changes from above, namely the definitions of the VOPE Δ_i and the expected precision $\gamma_i^{(k)}$, the update equations for the precision and the mean in volatility coupling simplify to:

$$\pi_i^{(k)} = \hat{\pi}_i^{(k)} + \frac{1}{2}(\kappa_{i,i-1}\gamma_{i-1}^{(k)})^2 + (\kappa_{i,i-1}\gamma_{i-1}^{(k)})^2\Delta_{i-1}^{(k)} - \frac{1}{2}\kappa_{i,i-1}^2\gamma_{i-1}^{(k)}\Delta_{i-1}^{(k)} \tag{21}$$

$$\mu_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{1}{2}\frac{\kappa_{i,i-1}\gamma_{i-1}^{(k)}}{\pi_i^{(k)}}\Delta_{i-1}^{(k)} \tag{22}$$

Therefore, at the time of the update, Node i needs to have access to the following quantities:

Its own predictions: $\hat{\mu}_i^{(k)}$, $\hat{\pi}_i^{(k)}$

Coupling strength: $\kappa_{i,i-1}$

From level below: $\Delta_{i-1}^{(k)}$, $\gamma_{i-1}^{(k)}$

4.2 Prediction Error Step

The exact computation of the prediction error depends, like the computation of the new prediction, on the nature of the coupling with the parent nodes. We will therefore assume in the following, that Node i is the volatility child of Node $i+1$. Then the following quantities have to be sent up to Node $i+1$ (see also necessary information from level below in a volatility parent):

Expected precision: $\gamma_i^{(k)}$

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

Node i has already performed the PREDICTION step on the previous trial, so it has already computed the predicted precision, $\hat{\pi}_i^{(k)}$, and the volatiliy estimate, $\nu_i^{(k)}$, and out of these the expected precision, $\gamma_i^{(k)}$, for the current trial. Hence, in the PE step, it needs to perform only the following calculations:

$$\delta_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)} \tag{23}$$

$$\Delta_i^{(k)} = \frac{\hat{\pi}_i^{(k)}}{\pi_i^{(k)}} + \hat{\pi}_i^{(k)}(\delta_i^{(k)})^2 - 1. \tag{24}$$

4.3 Prediction Step

Still assuming that Node i is the volatility child of Node $i+1$, the PREDICTION step consists of the following simple computations:

$$\hat{\mu}_i^{(k+1)} = \mu_i^{(k)} \tag{25}$$

$$\nu_i^{(k+1)} = \exp(\kappa_i \mu_{i+1}^{(k)} + \omega_i) \tag{26}$$

$$\hat{\pi}_i^{(k+1)} = \frac{1}{\frac{1}{\pi_i^{(k)}} + \nu_i^{(k+1)}} \tag{27}$$

$$\gamma_i^{(k+1)} = \nu_i^{(k+1)} \hat{\pi}_i^{(k+1)} \tag{28}$$

Thus, the prediction for trial $k+1$ depends again only on receiving the posterior mean of Node $i+1$ on trial k, and knowing the Node's own posteriors.

Note that if Node i additionally has a VAPE parent node, the prediction of the new mean, $\hat{\mu}_i^{k+1}$ would also depend on the posterior mean of that value parent (cf. PREDICTION step for VAPE coupling).

5 Implementation HGF coupling

If we assume that a cortical column consists of units (i.e., sub-populations of neurons), one for each computational step, then the computations for VAPE coupling as well as for VOPE coupling could be implemented as sketched in Figure 1.

A few points to note here:

- In this implementation, we assume that the estimated volatility, ν_i^k, is computed within the Prediction unit. This is motivated by the equations presented above and the fact that this unit receives information about the updated quantities (posterior means and precisions) from the level above.

- Similarly, the only unit that needs to know the value of a node's tonic learning rate, ω_i, is the Prediction unit. Therefore, this parameter could be implemented in terms of self-connections or interneurons within that neuronal population.

- A noteworthy arrow in this picture is the one from the Prediction unit of one node to the Update unit of its parent node. This seems to violate the classical assumptions that while top-down connections signal predictions, bottom-up connections would only signal prediction errors. Due to the update equations in the HGF, however, any given parent node needs access to the predicted precision from its child node to perform an update. To avoid this arrow, we could also assume that the Prediction unit sends the predicted precision to the PE unit of the

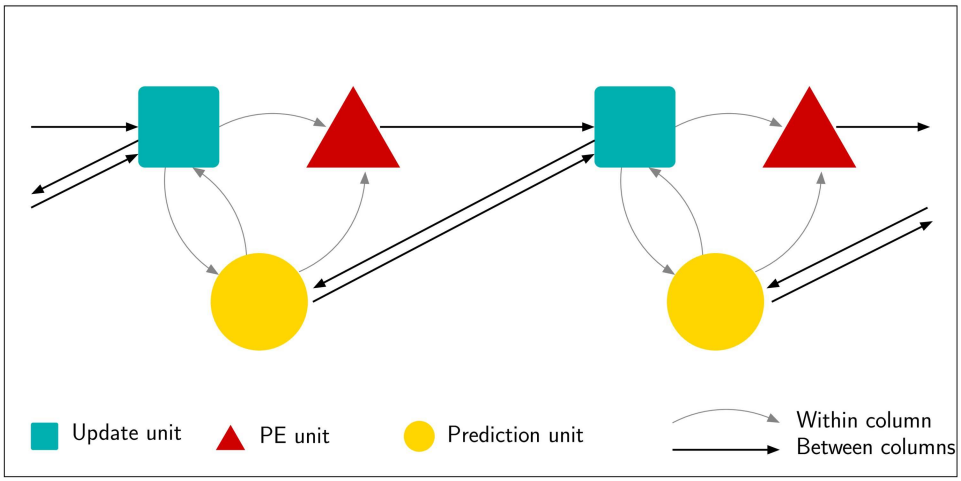

Figure 2: Example implementation of the coupling between cortical columns in the HGF. Figure shows two cortical columns that could be coupled to each other via VAPE or VOPE coupling. In each case, the messages passed along the connections and the computations within the nodes would differ according to the equations presented in the previous chapters.

same node, which then sends this quantity up to the Update unit of the parent node. However, this seems more complicated for two reasons: First, the PE unit doesn't actually need this quantity to perform its computation. Secondly, the PE is computed only somewhat after the prediction of the current trial (which is already computed after the previous trial). Thus, the two signals would be sent up at different times.

- Note that we did not distinguish between the computational steps Prediction and Update for the mean versus for the precision here. It is likely however, that these would be encoded in separate neuronal populations. Also, the implementation of VAPE and VOPE parents might differ. We will address these specifications below in comparison to classical predictive coding schemes.

6 Relation to Predictive Coding

In this section, we consider an implementation of the message passing implicated by the HGF which is as close as possible to current proposals of neural architectures for predictive coding [4] as shown in figure 3.

We will separately consider VAPE and VOPE coupling and realize that the message passing for VAPE coupling is almost equivalent to the proposed

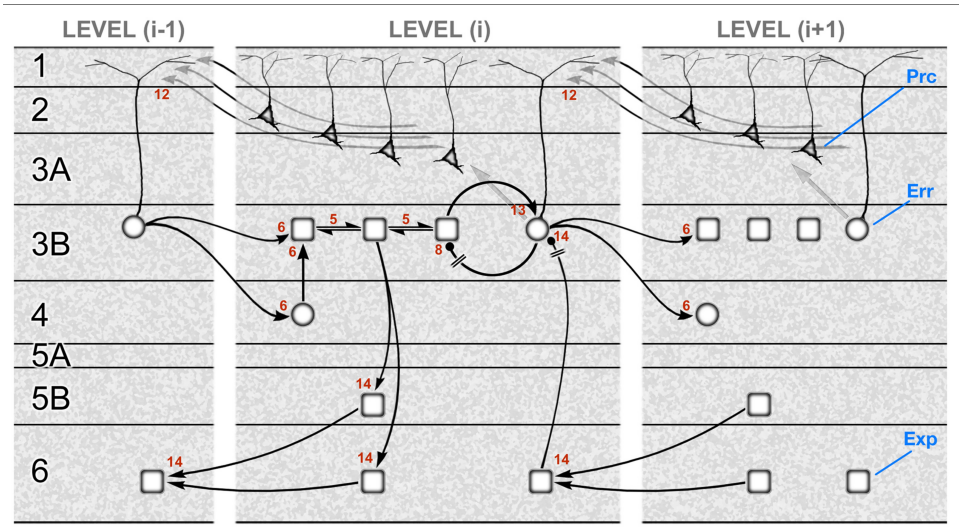

Figure 3: A template circuit diagram to implement predictive coding. All computational units are designated to be subclasses of excitatory pyramidal neurons: expectation units (rectangular, 'Exp'), prediction error units (circular, 'Err'), or precision units (triangular, 'Prc'). Precision signals are shown to arise from layers 2 and 3A, and form a descending chain of transmission through the superficial layers (pathway 12); these signals are capable of modulating pyramidal error units via their apical dendrite. Figure reproduced from [4].

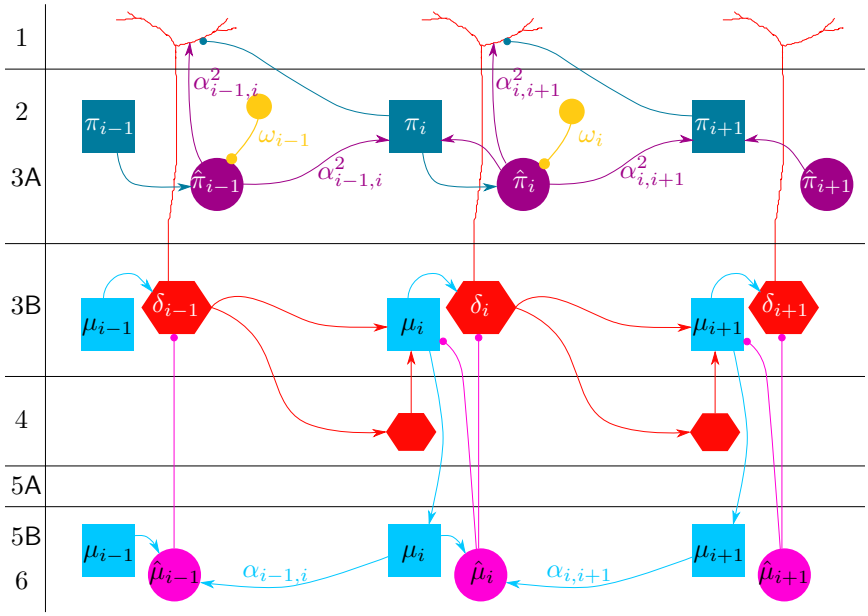

Figure 4: Overview of possible layer-specific message passing in VAPE coupling. Assignment of nodes to layers is loosely based on figure 3 in [4], reproduced in this document in figure 3. Connections with arrowheads indicate positive influences, or excitatory connections, while connections with circular heads indicate inhibitory influences.

PC architecture, while the VOPE coupling comes with more challenges, simply because it hasn't been addressed in the same detail in existing PC proposals on neural architecture (but see [2]).

6.1 VAPE coupling

Figure 4 sketches a possible architecture implementing all computations involved in VAPE coupling for three example cortical regions (levels). The assignment of neural elements to cortical layers follows the proposal in figure 3 of [4], reproduced here in figure 3. For example, we've placed all precision-related nodes into the upper layers, while expectations and predictions of the mean live in intermediate and deep layers, respectively. In the following, we go through the different computational steps and note differences to the proposal for predictive coding.

For the PE step, we have the following computation:

$$\delta_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)}. \tag{29}$$

The message passing implicated is depicted in figure 5 and is in line with PE computations as suggested by [4].

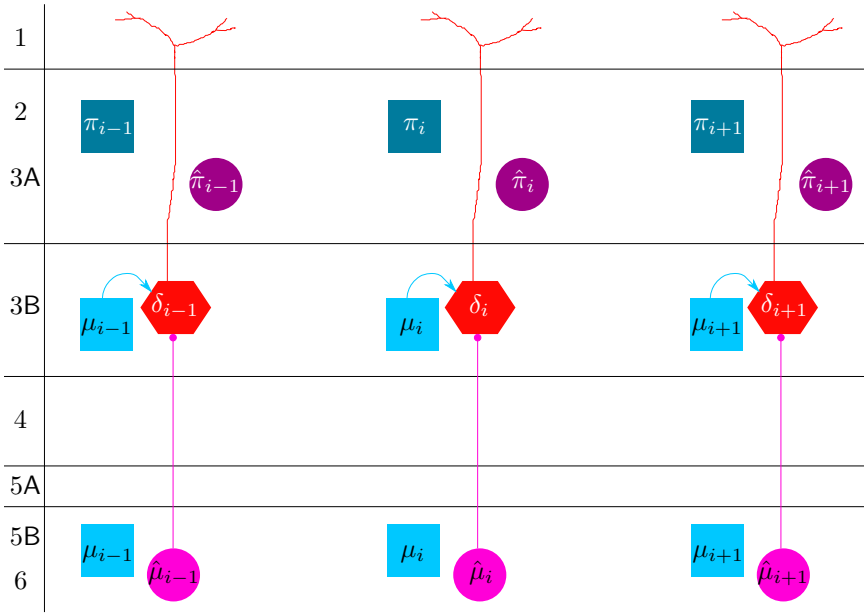

Figure 5: Implementation of VAPE PE computation on three cortical levels: $\delta_i = \mu_i - \hat{\mu}_i$.

In the Update step, the posterior estimates of mean and variance are given by:

$$\mu_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)} \tag{30}$$

and

$$\pi_i^{(k)} = \hat{\pi}_i^{(k)} + \alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}. \tag{31}$$

A possible message passing implementation for these updates is shown in figure 6. There are two important differences to [4] to note here.

First, in the PC proposal, the update of the mean μ_i is driven both by the lower-level PE δ_{i-1} and the PE on the level itself, δ_i, which mediates the influence of the empirical prior, or the prediction, on the mean.

In the HGF equations, we do not have this negative feedback loop from δ_i to μ_i. Instead, the prediction $\hat{\mu}_i$ has a direct influence on μ_i (see equations above), hence the arrow from the $\hat{\mu}_i$ node to the μ_i node in figure 6. This difference is due to the discrete nature of the update equations in the HGF, and can be resolved by considering the potential (PC-like) within-trial temporal evolution of the μ_i node. Noting that μ_i as computed in the above equation will be the endpoint of the update, i.e., the equilibrium value to

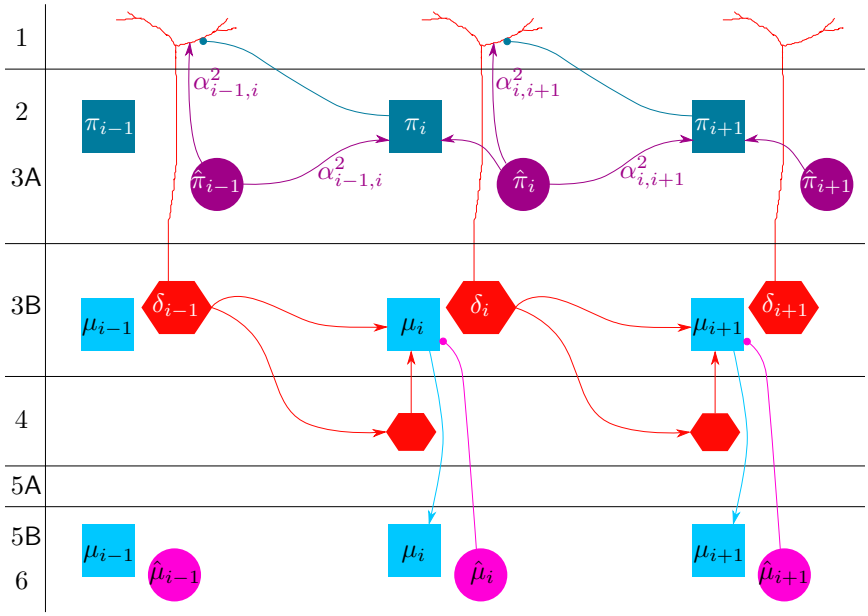

Figure 6: Message passing entailed by the Update step in VAPE coupling.

which the node should stabilize at the end of trial k, we can, in the style of [1], propose the following temporal evolution:

$$\dot{\mu}_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha^2_{i-1,i}\hat{\pi}^{(k)}_{i-1}}{\pi_i^{(k)}}\delta_{i-1}^{(k)} - \mu_i^{(k)} \tag{32}$$

A node with this dynamic will converge to the required posterior value, which can be seen by setting $\dot{\mu}_i$ to zero. Now we note that

$$\dot{\mu}_i^{(k)} = \frac{\alpha^2_{i-1,i}\hat{\pi}^{(k)}_{i-1}}{\pi_i^{(k)}}\delta_{i-1}^{(k)} - \mu_i^{(k)} + \hat{\mu}_i^{(k)} \tag{33}$$

can be summarized as

$$\dot{\mu}_i^{(k)} = \frac{\alpha^2_{i-1,i}\hat{\pi}^{(k)}_{i-1}}{\pi_i^{(k)}}\delta_{i-1}^{(k)} - \delta_i^{(k)}. \tag{34}$$

Following this differential equation for the temporal evolution of node μ_i, we could also place the arrow between δ_i and μ_i, resulting in the same feedback loop as in the PC proposal.[1] The exercise of writing down potential

[1]Note that in the implementation proposed in this chapter, this would not be possible, as the prediction error is always automatically precision-weighted via the modulation of its dendrites in the superficial layers. For the feedback loop however, one would need the prediction error in its unweighted form, a requirement that will be met in the next chapter.

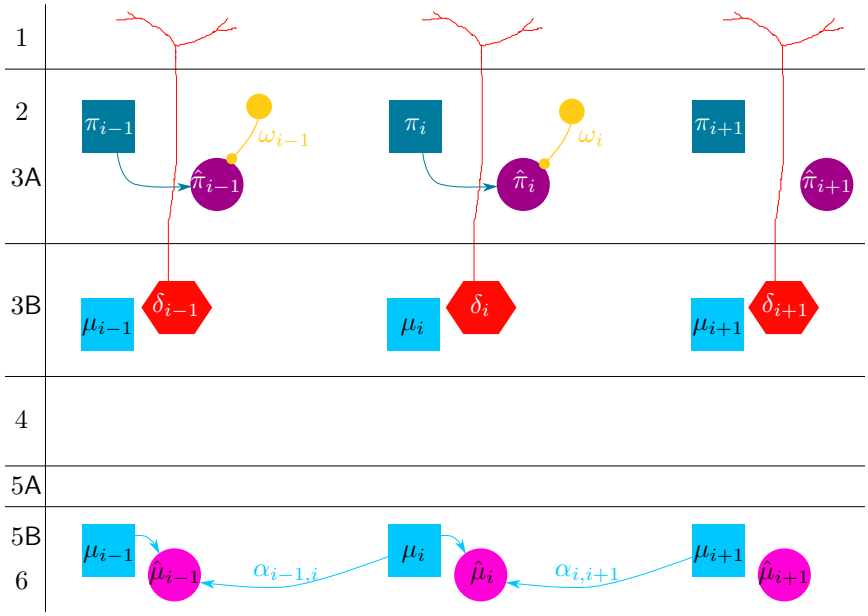

Figure 7: Message passing entailed by the Prediction step in VAPE coupling.

within-trial temporal dynamics for all the HGF nodes will be performed and discussed in the next chapter, where we will see that the resulting equations for continuous time network dynamics result in a more detailed and more biologically plausible implementation scheme.

The second difference to the PC proposal is rather irreducible. As already mentioned in the previous section, the messages sent up to parent nodes do not only consist of prediction errors or their precision weights. Instead, to update the precision node π_i, node i also needs access to the prediction precision $\hat{\pi}_{i-1}$ from the level below. This is reflected in the violet arrow travelling from superficial layers of level i to level $i+1$. However, given that in most PC proposals the update equations were concerned only with hierarchies for estimating the mean and not the precision (but see [2] for an exception), it does not seem too surprising that the explicit treatment of precision updates in the HGF results in new predictions for cortical architecture.

Finally, in the Prediction step, we have:

$$\hat{\mu}_i^{(k+1)} = \mu_i^{(k)} + \alpha_{i,i+1}\mu_{i+1}^{(k)} \tag{35}$$

and

$$\hat{\pi}_i^{(k+1)} = \frac{1}{\frac{1}{\pi_i^{(k)}} + \exp(\omega_i)}. \tag{36}$$

This again seems unproblematic, although the arrow from the posterior μ_i to the prediction $\hat{\mu}_i$ would not appear in the classical PC architecture, where the influence of the posterior on the new prediction is again mediated by the prediction error node. This can be resolved by moving to a continuous time representation in the same way as seen before and will be discussed below.

Finally, we have placed the predicted precision $\hat{\pi}$ in the superficial layers here, for convenience, as it interacts only with the posterior precision π and acts as a weight on the prediction error, and also inaccordance with [4], where precision signals arise from layers 2 and 3A.

6.2 VOPE coupling

For volatility parents, we propose the following: Each level of a cortical hierarchy implements its own volatility parent - or volatility estimation - in subpopulations of neurons within the same cortical column. Coupling across cortical levels, on the other hand, is exclusively value-related. This means that having a volatility parent is nothing more but a more sophisticated way of predicting and updating a level's precision estimates.

This setup deliberately excludes volatility parents of volatility parents. However, it allows for a cortical hierarchy where higher levels predict either the mean, or the volatility (or both), of lower levels. In other words, any given level receives top-down predictions not only about its mean, but also about its volatility, the current estimation of which is represented in the mean of its volatility parent. Higher-level predictions of volatility (top-down) and deviations from these predictions - value prediction errors about the volatility mean (bottom-up) are communicated between cortical levels via value (VAPE) coupling in the same way as higher-level predictions of the mean and deviations (value prediction errors about the state mean).

!Include schematic visualization of the different scenarios here!

Note that this setup also allows for a centralized or global volatility estimation. For example, a cortical or subcortical region which monitors the overall environmental volatility could be coupled to all local volatility units via value coupling. Our proposal is more flexible than previous ones [2] though, in that it also enables differential volatility estimation, e.g. within different sensory channels.

Moreover, in principle, both the mean μ_i as well as the volatility estimate $\check{\mu}_i$ of a given level could have (one or more) value parents in higher cortical areas. This allows for both separate and combined higher-level prediction

of both the precision and the mean of level i.

Due to our proposal, we slightly change the notation for volatility coupling from here on. Level i from now on refers to the hierarchical level within a cortical hierarchy. Nodes on this level comprise the 'original' nodes μ_i, π_i, $\hat{\mu}_i$, $\hat{\pi}_i$, and δ_i, which are concerned with value or state estimation, but also all the nodes belonging to a volatility parent of these nodes, which we now denote as $\check{\mu}_i$, $\check{\pi}_i$, $\hat{\check{\mu}}_i$, $\hat{\check{\pi}}_i$, and $\check{\delta}_i$. Here, $\check{\delta}_i$ refers to the value prediction error about the mean of the volatility, $\check{\mu}_i$.

The VOPE Δ_i, which we have written as a function of the VAPE δ_i, of level i, is what communicates between a level's nodes and a level's volatility parent's nodes. Value estimation and volatility estimation both happen within a cortical level or column. For the ensuing equations (simple reformulations of the computations in VOPE coupling as presented before) and visualizations of this proposal, please refer to the appendix.

!Discuss similarities with and differences to [2] here!

7 Within-trial time dynamics

In this section, we propose differential equations which lay out postential within-trial update dynamics entailed by the HGF. To this end, we consider the posterior values of all our nodes (quantities) as given by the HGF update equations as the equilibrium point towards which all dynamics must converge.

We will note that these equations imply a slightly different coupling between nodes compared to the previous section and we will introduce several new 'auxiliary' nodes of no interest, which help to simplify the implementation and thus increase the biological plausibility of our proposal.

We divide the equations into value estimation and volatility estimation, noting that VAPE and VOPE coupling are not useful divisions anymore (the mean of the value estimation of a level i, μ_i, will be VAPE-coupled to its parent μ_{i+1}, while the precision π_i of the value estimation will be VOPE-coupled to the volatility parent $\check{\mu}_i$ on level i itself; on the other hand, the mean of the volatility estimation, $\check{\mu}_i$, can again have a VAPE coupling to a higher level, e.g. μ_{i+2}, and the precision of volatility, $\check{\pi}_i$, will, in the current framework, not be coupled to any parent).

7.1 Value estimation

For the PE about the mean, we want to reach the following posterior:

$$\delta_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)}. \tag{37}$$

This can be easily achieved by a node with these dynamics:

$$\dot{\delta}_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)} - \delta_i^{(k)}. \tag{38}$$

From this it follows that we require an additional inhibitory self-connection for the PE node.

We now consider an alternative implementation for the PE computation, which will turn out to be very useful.

Instead of considering the unweighted δ_i, we can also directly model a node which corresponds to the precision-weighted prediction error ε_i:

$$\varepsilon_i = \frac{\alpha^2 \hat{\pi}_i}{\pi_{i+1}}(\mu_i - \hat{\mu}_i) = \frac{\alpha^2 \hat{\pi}_i}{\pi_{i+1}}\delta_i. \tag{39}$$

This node could evolve according to

$$\dot{\varepsilon}_i = \mu_i - \hat{\mu}_i - \frac{\pi_{i+1}}{\alpha^2 \hat{\pi}_i}\varepsilon_i, \tag{40}$$

which allows us to re-introduce the unweighted PE δ_i:

$$\begin{aligned} \delta_i &= \frac{\pi_{i+1}}{\alpha^2 \hat{\pi}_i}\varepsilon_i \\ &= \frac{\pi_{i+1}(\mu_i - \hat{\mu}_i)\alpha^2 \hat{\pi}_i}{\alpha^2 \hat{\pi}_i \pi_{i+1}} \\ &= \mu_i - \hat{\mu}_i, \end{aligned} \tag{41}$$

such that

$$\dot{\varepsilon}_i = \mu_i - \hat{\mu}_i - \delta_i \tag{42}$$

and

$$\dot{\delta}_i = \varepsilon_i - \alpha^2 \frac{\hat{\pi}_i}{\pi_{i+1}}\delta_i. \tag{43}$$

This has the advantage that the weighted PE ε_i can travel up the hierarchy to update higher levels, while the unweighted PE δ_i can be used to feedback to the mean on the same level, an element that will be introduced next.

In the Update step, the posterior estimate of the mean is given by:

$$\mu_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)}. \tag{44}$$

Here, we propose the following temporal evolution:

$$\dot{\mu}_i^{(k)} = \hat{\mu}_i^{(k)} + \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)} - \mu_i^{(k)} \tag{45}$$

Clearly, a node with this dynamic converges to the required posterior value, which can be seen by setting $\dot{\mu}_i$ to zero. Now we note that

$$\dot{\mu}_i^{(k)} = \frac{\alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}}{\pi_i^{(k)}} \delta_{i-1}^{(k)} - \mu_i^{(k)} + \hat{\mu}_i^{(k)} \tag{46}$$

can be summarized as

$$\dot{\mu}_i^{(k)} = \varepsilon_{i-1}^{(k)} - \delta_i^{(k)}. \tag{47}$$

We will therefore consider a connection between δ_i and μ_i, instead of connecting the prediction node $\hat{\mu}_i$ with the mean μ_i. As already outlined in the previous section, this also resolves one of the apparent differences between our message passing scheme and classical PC proposals.

For the posterior precision, we have

$$\pi_i^{(k)} = \hat{\pi}_i^{(k)} + \alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)}. \tag{48}$$

Here, we stick with the same approach as for the PE node, and simply add a self-inhibitory connection to implement the dynamics:

$$\dot{\pi}_i^{(k)} = \hat{\pi}_i^{(k)} + \alpha_{i-1,i}^2 \hat{\pi}_{i-1}^{(k)} - \pi_i^{(k)}. \tag{49}$$

Finally, in the Prediction step, we want to reach the following prediction of the mean:

$$\hat{\mu}_i^{(k+1)} = \mu_i^{(k)} + \alpha_{i,i+1} \mu_{i+1}^{(k)}. \tag{50}$$

Rewriting this as a differential equation

$$\dot{\hat{\mu}}_i^{(k+1)} = \alpha_{i,i+1} \mu_{i+1}^{(k)} + \mu_i^{(k)} - \hat{\mu}_i^{(k+1)}, \tag{51}$$

we again see the PE node appear:

$$\dot{\hat{\mu}}_i^{(k+1)} = \alpha_{i,i+1} \mu_{i+1}^{(k)} + \delta_i^{(k)}. \tag{52}$$

Note that $\delta_i^{(k)}$ is actually defined in terms of $\hat{\mu}_i^{(k)}$, and not $\hat{\mu}_i^{(k+1)}$, but given that we are in continuous time now, we suspect that this is probably equivalent here or can be resolved by introducing a delay between the computation of the prediction and the prediction error. For simplicity, we will skip the trial indices from now on.

The precision of this prediction should converge to:

$$\hat{\pi}_i = \frac{1}{\frac{1}{\pi_i} + \exp(\omega_i)}. \tag{53}$$

We can write the dynamics as:

$$\begin{aligned} \dot{\hat{\pi}}_i &= 1 - (\frac{1}{\pi_i} + \exp(\omega_i))\hat{\pi}_i \\ &= 1 - \frac{\hat{\pi}_i}{\pi_i} - \exp(\omega_i)\hat{\pi}_i. \end{aligned} \tag{54}$$

To simplify the implementation, we introduce an additional node p_{i1}:

$$p_{i1} = \frac{\hat{\pi}_i}{\pi_i}. \tag{55}$$

This new node can have the following dynamcis:

$$\dot{p}_{i1} = \hat{\pi}_i - \pi_i p_{i1}, \tag{56}$$

and the dynamics for the precision of the prediction $\hat{\pi}_i$ become

$$\dot{\hat{\pi}}_i = 1 - p_{i1} - \exp(\omega_i)\hat{\pi}_i, \tag{57}$$

where the last term can be implemented as a self-inhibition with a connection weight of $\exp(\omega_i)$. However, to facilitate the implementation of the influence of a volatility parent on the precision estimate, we introduce another additional node, p_{i2}, here and also already consider a node that estimates volatility, ν_i. In the case of value estimation without a volatility node, this estimate will only depend on the parameter value ω_i, i.e., the tonic volatility.

The dynamics of $\hat{\pi}_i$ then become:

$$\dot{\hat{\pi}}_i = 1 - p_{i1} - p_{i2}, \tag{58}$$

with the node p_{i2} defined as

$$p_{i2} = \hat{\pi}_i \exp(\omega_i), \tag{59}$$

with dynamics described by

$$\dot{p}_{i2} = \exp(\omega_i)\hat{\pi}_i - p_{i2} \tag{60}$$

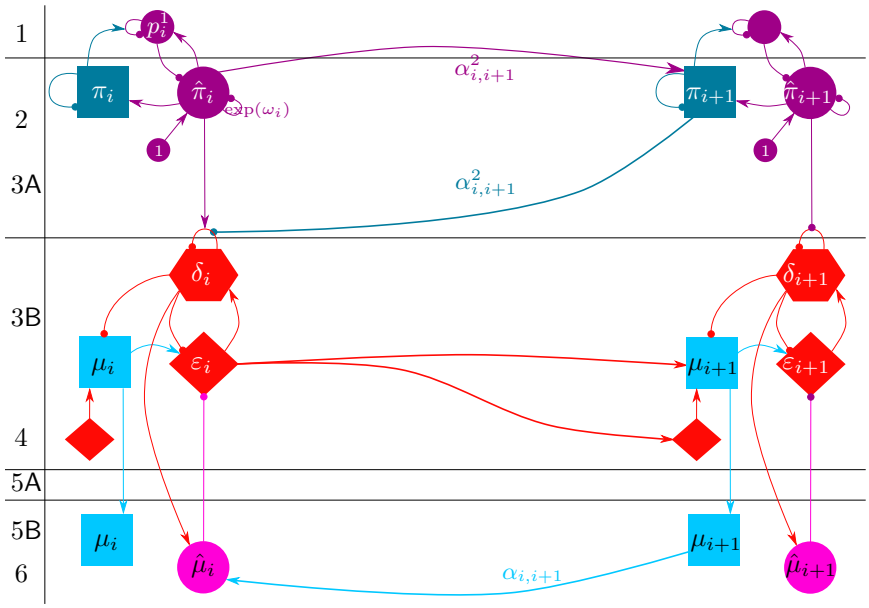

Figure 8: The coupling between a hierarchical level i with its parent $i+1$ in the case of VAPE coupling: implementing the differential equations that describe within-trial dynamics of the nodes.

in the absence of a volatility parent (see below for the extension to volatility estimation). We introduce the volatility estimate ν_i as

$$\nu_i = \exp(\omega_i), \tag{61}$$

evolving as

$$\dot{\nu}_i = \exp(\omega_i) - \nu_i, \tag{62}$$

leading to

$$\dot{p}_{i2} = \nu_i \hat{\pi}_i - p_{i2}. \tag{63}$$

A visualization of the network entailed by the differential equations so far - in the absence of volatility estimation - is depicted in figure 8.

Note that the dynamics of ν_i can be implemented by a self-inhibition and a positive influence from a tonicly active neuron (with activity 1) with a connection weight of $\exp(\omega_i)$. The advantage of this description is that if we add a volatility parent, we simply have to let this tonicly active neuron be driven by the volatility estimate $\check{\mu}_i$, weighted by the coupling parameter κ_i, to arrive at

$$\nu_i = \exp(\kappa_i \check{\mu}_i) \exp(\omega_i). \tag{64}$$

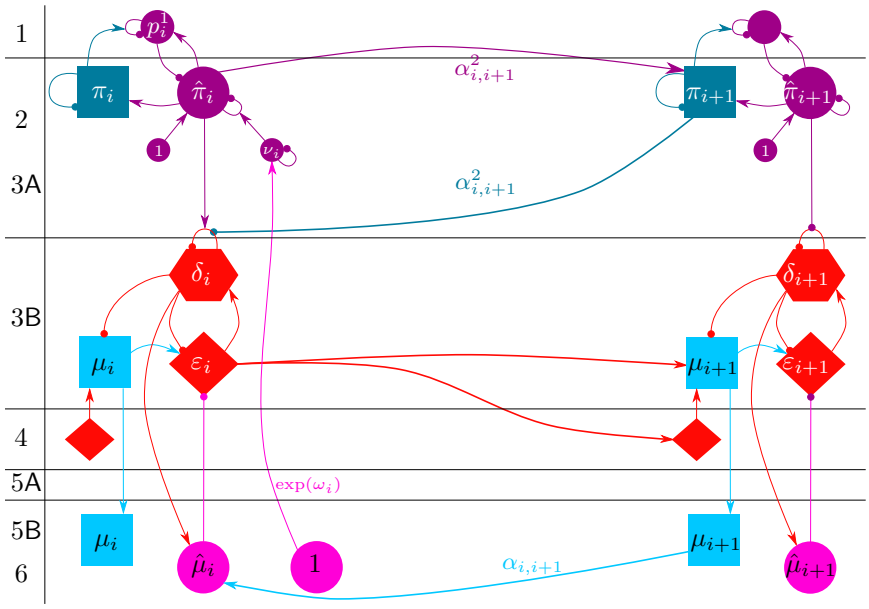

Figure 9: The coupling between a hierarchical level i with its parent $i+1$ in the case of VAPE coupling, using within-trial dynamics: introducing ν as the (tonic) volatility estimation.

Here we note that according to our proposal above, any cortical level i will implement its own volatility estimation, along with the value estimation. Therefore, the precision of the prediction, $\hat{\pi}_i$, will always depend on the value of the volatility parent $\check{\mu}_i$:

$$\hat{\pi}_i = \frac{1}{\frac{1}{\pi_i} + \exp(\kappa_i \check{\mu}_i + \omega_i)}. \tag{65}$$

This variant of the VAPE implementation, which is representing volatility estimation explicitely even in the absence of a VOPE parent, is depicted in figure 9.

We now turn to the computations entailed by phasic volatility estimation, including the modified dynamics for $\hat{\pi}_i$ and ν_i.

7.2 Volatility estimation

Here we start by considering the changes in the dynamics of the precision prediction in node i, i.e. the dynamics of $\hat{\pi}_i$ and ν_i, which are the consequence of adding an estimate of phasic volatility $\check{\mu}_i$.

We simply have to add a (multiplicative) term in the dynamics of the predicted volatility ν_i, which will, through the dependence of p_{i2} on ν_i and

$\hat{\pi}_i$ on p_{i2}, lead to the desired influence:

$$\dot{\nu}_i = \exp(\kappa_i \check{\mu}_i) \exp(\omega_i) - \nu_i. \tag{66}$$

Additionally, to simplify the precision weighting of the volatility prediction error, we use the auxiliary expected precision node γ_i as introduced in the previous section:

$$\gamma_i = \nu_i \hat{\pi}_i \tag{67}$$

with, for example,

$$\dot{\gamma}_i = \nu_i \hat{\pi}_i - \gamma_i. \tag{68}$$

For the PE about the level of volatility Δ_i, we have previously seen that we want to reach the following posterior:

$$\Delta_i^{(k)} = \frac{\hat{\pi}_i^{(k)}}{\pi_i^{(k)}} + \hat{\pi}_i^{(k)} (\delta_i^{(k)})^2 - 1. \tag{69}$$

This corresponds to the unweighted volatility prediction error. Again, we start by considering a node which in turn reflects the precision-weighted (VO)PE, which we will call E_i:

$$E_i^{(k)} = \frac{1}{2} \frac{\kappa_i \gamma_i^{(k)}}{\check{\pi}_i^{(k)}} \Delta_i^{(k)}. \tag{70}$$

This could evolve according to

$$\dot{E}_i^{(k)} = \Delta_i^{(k)} - \frac{2\check{\pi}_i^{(k)}}{\kappa_i \gamma_i^{(k)}} E_i^{(k)}, \tag{71}$$

which can also be written as

$$\dot{E}_i^{(k)} = \frac{\hat{\pi}_i^{(k)}}{\pi_i^{(k)}} + \hat{\pi}_i^{(k)} (\delta_i^{(k)})^2 - 1 - \Delta_i^{(k)}, \tag{72}$$

with the unweighted PE node Δ_i

$$\Delta_i^{(k)} = \frac{2\check{\pi}_i^{(k)}}{\kappa_i \gamma_i^{(k)}} E_i^{(k)} \tag{73}$$

evolving as

$$\dot{\Delta}_i^{(k)} = E_i^{(k)} - \frac{\kappa_i \gamma_i^{(k)}}{2\check{\pi}_i^{(k)}} \Delta_i^{(k)}. \tag{74}$$

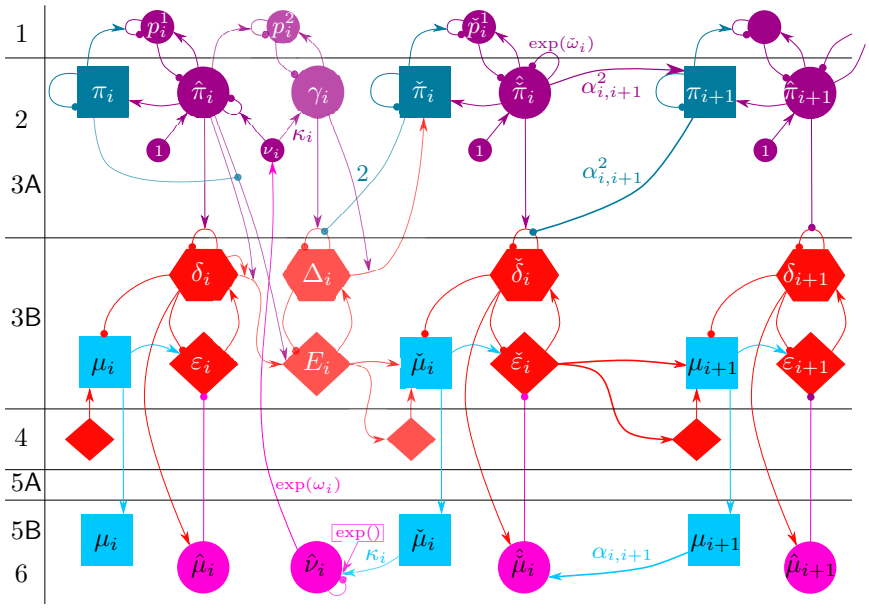

Figure 10: The coupling between a hierarchical level i with its parent $i+1$ in the case of VOPE coupling: implementing the differential equations that describe within-trial dynamics of the nodes.

This leads us to the same setup as for the value prediction error nodes, where we considered both an ε_i node and an unweighted PE node δ_i, as depicted in figure 10. In the case of volatility estimation, this again turns out to be useful: As before, the precision-weighted PE node E_i can be passed on to Update the volatility parent node $\check{\mu}_i$:

$$\check{\mu}_i^{(k)} = \hat{\check{\mu}}_i^{(k)} + \frac{1}{2}\frac{\kappa_i \gamma_i^{(k)}}{\check{\pi}_i^{(k)}}\Delta_i^{(k)} = \hat{\check{\mu}}_i^{(k)} + E_i^{(k)} \tag{75}$$

with

$$\dot{\check{\mu}}_i^{(k)} = E_i^{(k)} - (\check{\mu}_i^{(k)} - \hat{\check{\mu}}_i^{(k)}) = E_i^{(k)} - \check{\delta}_i. \tag{76}$$

There are three things worth mentioning here. First, in contrast to the previous section, we now use γ_i and Δ_i in the update equations (instead of γ_{i-1} and Δ_{i-1}). This is simply due to the fact that we have put the volatility parent $\check{\mu}_i$ into the same cortical level i as the node μ_i itself. Instead of moving from level i to level $i+1$ for the volatility estimation, we now denote value-related nodes as μ and volatility-related nodes as $\check{\mu}$.

Second, the (unweighted) PE node $\check{\delta}_i$, which is used to feedback to the update of $\check{\mu}_i$, is again a value PE, but now about the mean of the volatility estimate. This VAPE can again be used (in its precision-weighted form $\check{\varepsilon}_i$)

to communicate with higher cortical levels μ_{i+1}, completely parallel to the precision-weighted value prediction about the mean, ε_i.

Third, we note that in volatility estimation, we have to distinguish between the (weighted and unweighted) prediciton errors that are used to *update* the volatility estimate, E and Δ, and the (weighted and unweighted) prediciton errors *about* the volatility estimate, $\check{\varepsilon}_i$ and $\check{\delta}_i$, which will be used to update any higher cortical level $i+1$ predicting this volatility estimate.

Finally, although the unweighted volatility PE Δ_i is not needed for a feedback to μ_i or $\check{\mu}$, it is still very useful as it is used in the update of the precision of the volatility estimation $\check{\pi}_i$:

$$\check{\pi}_i^{(k)} = \hat{\check{\pi}}_i^{(k)} + \frac{1}{2}(\kappa_i\gamma_i^{(k)})^2 + (\kappa_i\gamma_i^{(k)})^2\Delta_i^{(k)} - \frac{1}{2}\kappa_i^2\gamma_i^{(k)}\Delta_i^{(k)}, \tag{77}$$

which can be summarized as

$$\check{\pi}_i^{(k)} = \hat{\check{\pi}}_i^{(k)} + \frac{1}{2}(\kappa_i\gamma_i^{(k)})^2 + ((\kappa_i\gamma_i^{(k)})^2 - \frac{1}{2}\kappa_i^2\gamma_i^{(k)})\Delta_i^{(k)}. \tag{78}$$

This could have the temporal evolution

$$\dot{\check{\pi}}_i^{(k)} = \hat{\check{\pi}}_i^{(k)} + \frac{1}{2}(\kappa_i\gamma_i^{(k)})^2 + ((\kappa_i\gamma_i^{(k)})^2 - \frac{1}{2}\kappa_i^2\gamma_i^{(k)})\Delta_i^{(k)} - \check{\pi}_i^{(k)}. \tag{79}$$

Finally, turning towards the Prediction step in the volatility estimation, this is again very simple, as we only consider value parents for volatility nodes. For the prediction of the mean, we have

$$\hat{\check{\mu}}_i^{(k+1)} = \check{\mu}_i^{(k)} + \alpha_{i,i+1}\mu_{i+1}^{(k)}, \tag{80}$$

which could evolve as

$$\dot{\hat{\check{\mu}}}_i^{(k+1)} = \check{\mu}_i^{(k)} - \hat{\check{\mu}}_i^{(k+1)} + \alpha_{i,i+1}\mu_{i+1}^{(k)} = \check{\delta}_i + \alpha_{i,i+1}\mu_{i+1}^{(k)}, \tag{81}$$

and for the precision of that prediction, we get

$$\hat{\check{\pi}}_i^{(k+1)} = \frac{1}{\frac{1}{\check{\pi}_i^{(k)}} + \exp(\check{\omega}_i)}, \tag{82}$$

which could evolve as shown above in the value estimation (using the auxiliary nodes $\check{p}_{i1}$):

$$\dot{\hat{\check{\pi}}}_i = 1 - \check{p}_{i1} - \hat{\check{\pi}}_i \exp(\check{\omega}_i) \tag{83}$$

with

$$\dot{\check{p}}_{i1} = \hat{\check{\pi}}_i - \check{\pi}_i\check{p}_{i1}. \tag{84}$$

Note that we do not bother to introduce $\check{p}_{i2}$ or $\check{\nu}_i$ here, as we exclude the possibility of a volatility parent $\check{\mu}_i$ having a volatility parent.

As can be seen from figure 10, the value coupling of the volatility mean with a higher-level region is now completely parallel to the value coupling seen in figure 9, where the higher-level region was concerned with predicting the lower-level mean. Thus the predictions about extrinsic connections - those connecting different levels of a cortical hierarchy - in volatility estimation are equivalent to those in mean or state estimation.

7.3 Conclusion

We have proposed a set of differential equations describing potential within-trial dynamics of network nodes that implement the discrete update equations of the HGF as the endpoint of their evolution in each trial. These equations, together with the introduction of a few additional nodes such as the precision-weighted prediction errors, imply a simple implementation scheme that bears a strong resemblance to current proposals of layer-specific implementations of PC schemes.

Moreover, these differential equations can be used to simulate within-trial responses of all nodes, which allows both for examining the stability of the proposed network, and for predicting layer-specific neural dynamics which could for example be compared with electrophysiological data.

8 Open Questions and Issues

- To Do:
  - Decide on the most useful visualizations for the paper:
    - move figure 8 to the appendix or discard it?
    - include a figure that schematically displays volatility and value estimation within a cortical column and how this could be coupled to higher-level regions, skipping all computational details within the columns, for section 5.2?
    - separate versions of figure 1 for VAPE and VOPE coupling with a specification of the quantities being signaled by every arrow?
    - update VOPE figures for new definition of gamma (i.e., move the kappa weight to the other connections)
  - Discuss overlap and differences with Karls precision paper [2].
  - more elaborate description of VOPE figures in the appendix
  - Implement the new network setup in the python code.

  - Additionally implement the differential equations, examine the stability of the network and present exemplary simulations.

- Equations: The coupling parameters α and κ must be available to both the connection between a node's PE unit and the parent's UPDATE unit, as well as the connection between a node's PREDICTION unit and the parent's UPDATE unit. How can it be ensured that the coupling strength is the same for both connections? For this question we would actually have to consider the learning or update equations for the parameters of the model (see for example the approach in [1]). However, even without writing them down, we could assume that under 'healthy' conditions, the weights of these different connections automatically converge to the same (or at least very similar) values. This opens up new possibilities of modeling abnormal inference: Using the simulations presented above, we could find out how an agent's inference would be altered if these values were not the same, and what range of difference would actually lead to observable changes in the inference.

- Implementation: To whom does the knowledge about parameters, in particular coupling strengths, belong? Ideally, this would belong to the connections themselves and be accessible to both child and parent node. However, in terms of synaptic signalling, how can a node have direct access to a coupling strength other than via the signalled value (which is the product of the coupling strength and the signal)?

- Not covered here: Computations of input nodes and binary input nodes.

9 Appendix

9.1 Computations in VOPE coupling

In the following, we present the computations of volatility nodes according to the proposal that each cortical level of a hierarchy implements its own volatility estimation.

The Update steps in volatility coupling are:

$$\check{\mu}_i^{(k)} = \hat{\check{\mu}}_i^{(k)} + \frac{1}{2}\frac{\kappa_i \gamma_i^{(k)}}{\check{\pi}_i^{(k)}}\Delta_i^{(k)} \tag{85}$$

$$\check{\pi}_i^{(k)} = \hat{\check{\pi}}_i^{(k)} + \frac{1}{2}(\kappa_i \gamma_i^{(k)})^2 + (\kappa_i \gamma_i^{(k)})^2 \Delta_i^{(k)} - \frac{1}{2}\kappa_i^2 \gamma_i^{(k)} \Delta_i^{(k)} \tag{86}$$

In the PE step, the node computes:

$$\Delta_i^{(k)} = \frac{\hat{\pi}_i^{(k)}}{\pi_i^{(k)}} + \hat{\pi}_i^{(k)}(\delta_i^{(k)})^2 - 1. \tag{87}$$

Finally, In the Prediction step, we now need to compute four nodes: the predicted (volatility) mean

$$\hat{\check{\mu}}_i^{(k+1)} = \check{\mu}_i^{(k)}, \tag{88}$$

the precision of that prediction

$$\hat{\check{\pi}}_i^{(k+1)} = \frac{1}{\frac{1}{\check{\pi}_i^{(k)}} + \nu_i^{(k+1)}}, \tag{89}$$

the predicted environmental uncertainty (as a function of the next higher level in the hierarchy, μ_{i+1})

$$\nu_i^{(k+1)} = \exp(\kappa_{i,i+1}\mu_{i+1}^{(k)} + \omega_i), \tag{90}$$

and the new (auxiliary) expected precision

$$\gamma_i^{(k+1)} = \nu_i^{(k+1)}\hat{\pi}_i^{(k+1)}. \tag{91}$$

The last node is only defined for convenience in terms of simplifying the equations and the corresponding message passing.

9.2 Possible PC-like implementation of VOPE coupling

Figure 11 displays one proposal for within-column volatility estimation, where we've zoomed in to a level i of the cortical hierarchy (and a value parent $i+1$) and added its volatility parent to the superficial layers.

An alternative implementation of this idea is displayed in figure 12. Here, only the precision-related computations of the volatility parent are placed in the superficial layers, while the corresponding prediction errors, Δ_i and $\check{\delta}_i$ as well as the actual volatility estimate $\check{\mu}_i$ live in middle layers, and the prediction of the volatility estimate, $\hat{\check{\mu}}_i$, lives in layer 6. This setup stresses the structural similarities to the message passing entailed by mean coupling. Moreover, using this setup, we can now easily depict how a higher cortical level $i+1$ would serve as a value parent to the volatility estimate in level i - instead of predicting the mean - by simply exchanging the arrows between the levels as shown in figure 13.

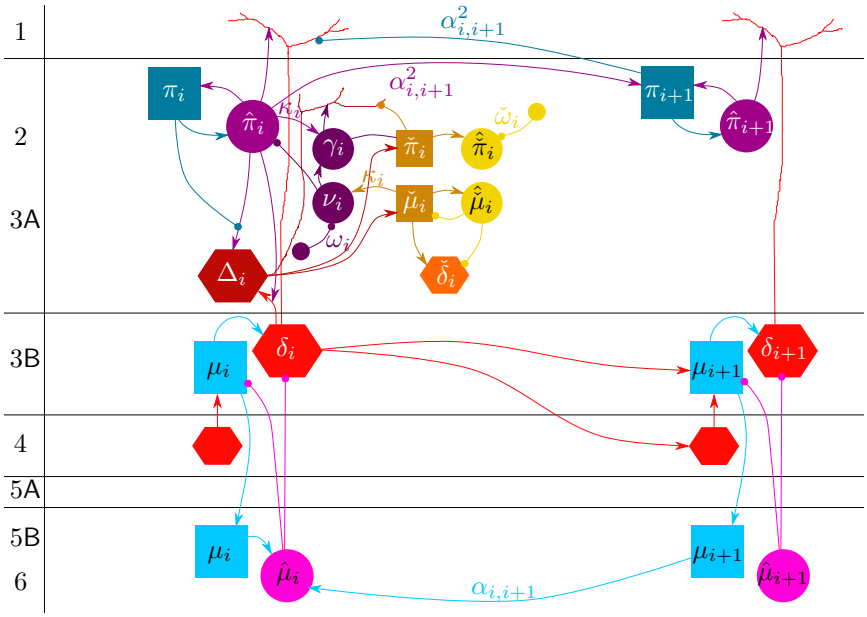

Figure 11: Overview of possible layer-specific message passing in VOPE coupling. Assignment of nodes to layers is loosely based on Figure 3 in [4]

.

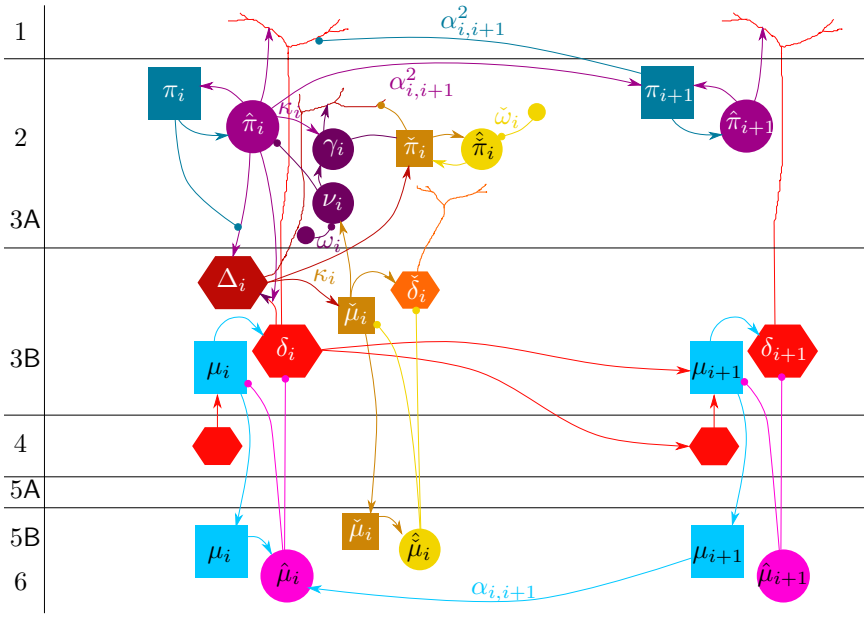

Figure 12: Zoomed-in version of the coupling between a hierarchical level i with its parent $i+1$ in the case of VAPE coupling, with an additional VOPE parent *within* node i.

.

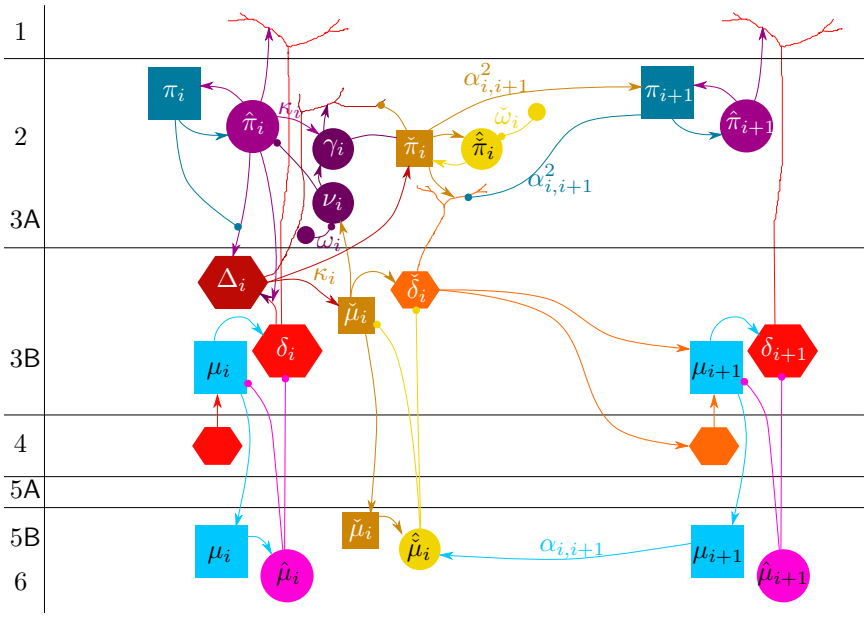

Figure 13: Zoomed-in version of the coupling between a hierarchical level i with its parent $i+1$ in the case of VOPE coupling. Note that the coupling is of VOPE type within level i, to update $\check{\mu}_i$, which is turn predicted by level $i+1$, where the higher level now serves as a VAPE parent to the volatility estimate $\check{\mu}_i$.

9.3 Computations of input nodes

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, the message passing (and the within-node computations) differs from the previously discussed scheme.

9.3.1 Continuous input nodes

A continuous input node can compute two prediction errors: VAPEs and VOPEs. We will first deal with the computations related to VAPE coupling of the continuous input node with a parent HGF node. Note that an input node will always have a VAPE or value parent, while having a volatility parent is optional.

Value parents of continuous input nodes

In the PE step, the VAPE will be computed as the difference the prediction and the posterior, where the current prediction $\hat{\mu}_{vapa}$ of the value parent $vapa$ is used as the prediction, and the value of the input itself, u, is used as the posterior:

$$\delta_{inp}^{(k)} = \mu_{inp}^{(k)} - \hat{\mu}_{inp}^{(k)} = u^{(k)} - \hat{\mu}_{vapa}^{(k)} \tag{92}$$

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.

If one wanted to construct an UPDATE step for the input node, the posterior mean simply amounts to the input itself, while the posterior precision will be determined by the value parent's precision:

$$\mu_{inp}^{(k)} = u^{(k)} \tag{93}$$

$$\pi_{inp}^{(k)} = \pi_{vapa}^{(k)} \tag{94}$$

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

$$\pi_{vapa}^{(k)} = \hat{\pi}_{vapa}^{(k)} + \hat{\pi}_{inp}^{(k)} \tag{95}$$

$$\mu_{vapa}^{(k)} = \hat{\mu}_{vapa}^{(k)} + \frac{\hat{\pi}_{inp}^{(k)}}{\pi_{vapa}^{(k)}} \delta_{inp}^{(k)} \tag{96}$$

That means that the input node needs to signal bottom-up to its value 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 would always have a connection weight of $\alpha = 1$.

In the PREDICTION step, the input node would have to compute the predicted mean $\hat{\mu}_{inp}^{(k+1)}$ and the predicted precision $\hat{\pi}_{inp}^{(k+1)}$ for the next trial. However, as described above, the predicted mean will be received from the value parent:

$$\hat{\mu}_{inp}^{(k+1)} = \hat{\mu}_{vapa}^{(k+1)} \tag{97}$$

Given that the value parent might operate under a drift parameter, the current prediction can only be computed given the time of the next input, i.e., at the beginning of the next trial, when the new input (and its time information) comes in. Then it needs to be signalled top-down immediately.

The predicted precision of the input node is a function of the input node's ω parameter, i.e., a constant, in the absence of a volatility parent:

$$\hat{\pi}_{inp}^{(k+1)} = \frac{1}{\exp(\omega_{inp})} \tag{98}$$

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

$$\hat{\pi}_{inp}^{(k+1)} = \frac{1}{\exp(\kappa_{vopa,inp}\mu_{vopa}^{(k)} + \omega_{inp})}. \tag{99}$$

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 $\hat{\mu}_{vapa}^{(k)}$ as mean and $\hat{\pi}_{inp}^{(k)}$ as the precision:

$$-\log(p(u^{(k)})) = \frac{1}{2}(\log(2\pi) - \log(\hat{\pi}_{inp}^{(k)}) + \hat{\pi}_{inp}^{(k)}(u^{(k)} - \hat{\mu}_{vapa}^{(k)})^2). \tag{100}$$

Volatility parents of continuous input nodes

Having a volatility parent for a continuous input node means that a VOPE will be computed and signalled bottom-up during the PE step. Importantly, the definition of the VOPE differs from previous definitions in that both the posterior precision as well as the posterior mean are taken from the value parent *vapa*:

$$\Delta_{inp}^{(k)} = \frac{\hat{\pi}_{inp}^{(k)}}{\pi_{vapa}^{(k)}} + \hat{\pi}_{i}^{(k)}(u^{(k)} - \mu_{vapa}^{(k)})^2 - 1. \tag{101}$$

This means that the VOPE is not a direct function of the VAPE anymore, which was instead defined in terms of the difference between the input $u^{(k)}$ (as the posterior) and the predicted mean of the value parent $\hat{\mu}_{vapa}^{(k)}$ as the prediction. This in turn requires that the update of the value parents happens before the computation of the VOPE, and the posterior of the value parent is already available to the input node.

The UPDATE step for the volatility parent is also unusual in that the expected precision term γ_{inp} is fixed to 1. Hence the update of the mean reads:

$$\mu_{vopa}^{(k)} = \hat{\mu}_{vopa}^{(k)} + \frac{1}{2}\frac{\kappa_{vopa,inp}\gamma_{inp}^{(k)}}{\pi_{vopa}^{(k)}}\Delta_{inp}^{(k)} \tag{102}$$

$$= \hat{\mu}_{vopa}^{(k)} + \frac{1}{2}\frac{\kappa_{vopa,inp}}{\pi_{vopa}^{(k)}}\Delta_{inp}^{(k)}. \tag{103}$$

This simply means that along with the modified VOPE, the input node needs to signal a value of 1 as the expected precision on every trial. Setting γ_{inp} to 1 in the update of the precision of the volatility parent leads to:

$$\pi_{vopa}^{(k)} = \hat{\pi}_{vopa}^{(k)} + \frac{1}{2}(\kappa_{vopa,inp}\gamma_{inp}^{(k)})^2 + (\kappa_{vopa,inp}\gamma_{inp}^{(k)})^2\Delta_{inp}^{(k)} - \frac{1}{2}\kappa_{vopa,inp}^2\gamma_{inp}^{(k)}\Delta_{inp}^{(k)} \tag{104}$$

$$= \hat{\pi}_{vopa}^{(k)} + \frac{1}{2}(\kappa_{vopa,inp})^2 + (\kappa_{vopa,inp})^2\Delta_{i-1}^{(k)} - \frac{1}{2}\kappa_{vopa,inp}^2\Delta_{i-1}^{(k)} \tag{105}$$

$$= \hat{\pi}_{vopa}^{(k)} + \frac{1}{2}(\kappa_{vopa,inp})^2 + \frac{1}{2}(\kappa_{vopa,inp})^2\Delta_{i-1}^{(k)} \tag{106}$$

$$= \hat{\pi}_{vopa}^{(k)} + \frac{1}{2}(\kappa_{vopa,inp})^2(1 + \Delta_{i-1}^{(k)}). \tag{107}$$

Finally, the PREDICTION step of a continuous input node will be modified by the presence of a volatility parent in that the predicted precision now also depends on the posterior mean of the volatility parent:

$$\hat{\pi}_{inp}^{(k+1)} = \frac{1}{\exp(\kappa_{inp}\mu_{vopa}^{(k)} + \omega_{inp})}. \tag{108}$$

Peculiarities of continuous input nodes and consequences for their parents
First, due to the dependence of the VOPE on the posterior beliefs of the value 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 volatility 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 continuous 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}
- compute VOPE using u, $\hat{\pi}_{inp}$, μ_{vapa} and π_{vapa}

- send bottom-up to volatility parent: VOPE, $\gamma_{inp} = 1$, and time
- receive top-down: posterior μ_{vopa}
- compute new prediction for precision $\hat{\pi}_{inp}^{(k+1)}$ using $\mu_{vopa}^{(k)}$,

and 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.

9.3.2 Binary input nodes

Binary input nodes 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.

For binary input nodes, 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 precision is infinite, i.e., $\hat{\pi}_{inp} = \text{inf}$, or in other words, there is no sensory noise, or 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-messages are simply $\hat{\pi}_{inp}$ itself, and u. The surprise computation is also very simple:

$$surprise^{(k)} = \begin{cases} -\log(1 - \hat{\mu}_{pa}^{(k)}), & \text{for } u^{(k)} = 1 \\ -\log(\hat{\mu}_{pa}^{(k)}), & \text{for } u^{(k)} = 0. \end{cases} \tag{109}$$

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 precision $\hat{\pi}_{inp}$ 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 precision.

The messages to be sent bottom-up will in this case again be the precision $\hat{\pi}_{inp}$, along with two prediction errors, namely the deviations of the input u from both possible values η_a and η_b:

$$\delta_{inp,1}^{(k)} = u^{(k)} - \eta_1 \tag{110}$$

$$\delta_{inp,0}^{(k)} = u^{(k)} - \eta_0. \tag{111}$$

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

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

$$surprise^{(k)} = -\log(\hat{\mu}_{pa}^{(k)} * \mathcal{N}(u^{(k)}; \eta_1, \hat{\pi}_{inp}) + (1 - \hat{\mu}_{pa}^{(k)}) * \mathcal{N}(u^{(k)}; \eta_0, \hat{\pi}_{inp})) \tag{112}$$

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.

9.3.3 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, depending on the precision of the child binary input node, can 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). This message elicits the UPDATE step, where the two cases are reflected as follows:

$$\mu_i^{(k)} = u^{(k)} \tag{113}$$

$$\pi_i^{(k)} = \hat{\pi}_{inp} \tag{114}$$

if

$$\hat{\pi}_{inp} = \inf, \tag{115}$$

and

$$\mu_i^{(k)} = \frac{\hat{\mu}_i^{(k)} * \exp(-\frac{1}{2}\hat{\pi}_{inp}(\delta_{inp,1})^2)}{\hat{\mu}_i^{(k)} * \exp(-\frac{1}{2}\hat{\pi}_{inp}(\delta_{inp,1})^2) + (1-\hat{\mu}_i^{(k)}) * \exp(-\frac{1}{2}\hat{\pi}_{inp}(\delta_{inp,0})^2)} \tag{116}$$

$$\pi_i^{(k)} = \frac{1}{\hat{\mu}_i^{(k)} * (1-\hat{\mu}_i^{(k)})} \tag{117}$$

if

$$\hat{\pi}_{inp} \neq \inf. \tag{118}$$

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

$$\delta_i^{(k)} = \mu_i^{(k)} - \hat{\mu}_i^{(k)}. \tag{119}$$

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

$$\pi_{pa}^{(k)} = \hat{\pi}_{pa}^{(k)} + \frac{1}{\hat{\pi}_i^{(k)}}. \tag{120}$$

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

$$\mu_{pa}^{(k)} = \hat{\mu}_{pa}^{(k)} + \frac{1}{\pi_{pa}^{(k)}}\delta_i^{(k)}. \tag{121}$$

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 by a binary HGF node is:

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

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

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

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

Precision weight for mean update: 1

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

Finally, in the PREDICTION step, the binary HGF node would compute its predictions based on its parents' predictions:

$$\hat{\mu}_i^{(k)} = \frac{1}{1 + \exp(-\hat{\mu}_{pa}^{(k)})} \tag{122}$$

$$\hat{\pi}_i^{(k)} = \frac{1}{\hat{\mu}_i^{(k)} * (1 - \hat{\mu}_i^{(k)})}. \tag{123}$$

Thus, we need to introduce a new top-down signalling step at the beginning of each trial, where the parent nodes send down its current prediction of the mean, given the time of a new input. We anyway need to do this for the communication with continuous input nodes.

9.3.4 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, even though this is never needed except in the mentioned case.
- Value 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 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 coupling, will be part of the connection logic. An argument for this could be that the exact wiring of the connections will lead to the excitatory input needed in the level above to perform the precision update.

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.

References

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

[2] Kanai, R., Komura, Y., Shipp, S., and 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.

[3] Mathys, C. D. (2011). A Bayesian foundation for individual learning under uncertainty. *Frontiers in Human Neuroscience*, 5(May):1–20.

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