APPENDIX A: EXPANSION AROUND QSS

In this Appendix, we detail how we derive the memory evolution over time. We make use of our
expansion Eq. (13) to find for the first term on the right hand side (RHS) of Eq. (5),

\[ L_{F_b} = \sum_b R_b(x^a, x^b) f_{bs}(x^a, \tau) + \sum_{i} R_{r}(x^a, x^b) \sum_b (x_b - x_b^0) \frac{\partial}{\partial x_b} f_{bs}(x^a, \tau) - \sum_{i} R_r(x^a, x^b) \sum_b \frac{\partial x_b^0}{\partial x_a} f_{bs}(x^a, \tau) \]  

(A1)

where the two last terms arise by differentiating the product \( (x_b - x_b^0) f_{bs}(x^a, \tau) \) w.r.t. \( x^b \) and we have not written the \( x^a \) dependence of \( x_{bs} \) for brevity. The second term on the RHS of Eq. (5) is the expectation of this, obtained by replacing \( x^b \) by \( x_{bs} \),

\[ E[L_{F_b}|x^a] = \sum_b R_b(x^a, x_{bs}) f_{bs}(x^a, \tau) - \sum_{i} R_{r}(x^a, x_{bs}) \sum_b \frac{\partial x_b^0}{\partial x_a} f_{bs}(x^a, \tau). \]  

(A2)

Putting the two together gives for the time evolution Eq. (5) of \( L_{F_b} \):

\[ \frac{\partial L_{F_b}}{\partial \tau} = \sum_b [R_b(x^a, x^b) - R_b(x^a, x_{bs})] f_{bs}(x^a, \tau) + \sum_{i} R_{r}(x^a, x^b) \sum_b (x_b - x_b^0) \frac{\partial}{\partial x_b} f_{bs}(x^a, \tau) \]

\[ - \sum_{i} R_{r}(x^a, x^b) - R_{r}(x^a, x_{bs})] \frac{\partial x_b^0}{\partial x_a} f_{bs}(x^a, \tau). \]  

(A3)

For consistency with Eq. (13), we now linearize the square brackets again in \( x^b - x_{bs} \). In the second term, we similarly replace \( R_{r}(x^a, x^b) \) by \( R_{r}(x^a, x_{bs}) \) as the remaining factor in this term is already linear. Comparing then with the time derivative of the original linearized formula Eq. (13) gives, after appropriate relabelling of indices, the equation for the evolution of \( f_{bs} \) in time Eq. (15).

From the above expansion, one sees that the matrix \( l_{bb'} \) in Eq. (15) takes the form

\[ l_{bb'} = \frac{\partial R_{r}}{\partial x_b} - \sum_b \frac{\partial R_{r}}{\partial x_b} \frac{\partial x_b^0}{\partial x_{b'}}. \]  

(A4)

The form Eq. (16) in the main text is obtained by using the identity

\[ \frac{\partial x_b^0}{\partial x_{b'}} = - \sum_{b'} (J^{-1})_{bb'} \frac{\partial R_{r}}{\partial x_b}. \]  

(A5)

The latter can be obtained by differentiating Eq. (9) with respect to \( x_b \).

To obtain the actual memory function from Eq. (19) is now straightforward as we have already worked out the required expectation in Eq. (A2). The first term on the r.h.s. of Eq. (A2), which we had previously kept to make the ensuing linearization easier to see, actually vanishes because of Eq. (9), yielding

\[ M_r(x^a, \tau) = - \sum_{x^a', x^b} R_{r} \frac{\partial x_b^0}{\partial x_a} f_{bs}(x^a, \tau). \]  

(A6)

Using again the identity Eq. (A5), we obtain our main result Eq. (20).

**APPENDIX B: SOLUTION FOR F**

In this Appendix, we find the solution \( f_{bs}(x^a, \tau) \) to the differential Eq. (15). We start by restating the latter as

\[ \frac{\partial}{\partial \tau} f_{bs} - \sum_{x^a'} v_r(x^a') \frac{\partial}{\partial x_a} f_{bs} = \sum_{b'} l_{bb'}(x^a') f_{bs}, \]  

(B1)

where we have used that the factor \( R_{r} \) in Eq. (15) is just the effective drift \( v_r \) defined in Eq. (11). As the equation is linear in \( f_{bs} \), its derivatives it can be solved using the method of characteristics (see, e.g., Ref. [41]). Calling the curve parameter for a characteristic \( u \), the characteristic equations can be read off from Eq. (B1) as

\[ \frac{d\tau}{du} = 1, \]

\[ \frac{dx}{du} = - v_r(x^a), \]

\[ \frac{df_{bs}}{du} = \sum_{b'} l_{bb'}(x^a') f_{bs}. \]  

(B4)

Setting an arbitrary integration constant to zero, the first of these gives \( \tau = u \). To solve Eq. (B3), we call \( \phi_r \), the flow generated by \( v_r(x^a) \), which is defined as the solution of the differential equations

\[ \frac{\partial}{\partial \tau} \phi_r(x^a, \tau) = v_r(\phi_r(x^a, \tau)), \quad \phi_r(x^a, 0) = x^a. \]  

(B5)

The solution of Eq. (B3) is then

\[ x^a(u) = \phi_r(x^a, -u), \]  

(B6)

where \( x^a_0 \) is the value at the beginning of the characteristic curve \( (u = 0) \); the minus sign in the second argument of \( \phi_r \) reflects the backward in time propagation in Eq. (B3). We note for later that, as a consequence of Eq. (B6), the solution values at \( u_1 \) and \( u_2 \) are related by

\[ x^a(u_2) = \phi_r(x^a(u_1), -u_2 + u_1). \]  

(B7)

Finally, the solution of Eq. (B4) is

\[ f_{bs}(u) = \sum_{b'} \left( \sum_{a'} \left( e^u v_r(x^a') l_{bb'}(x^a) \right) \right) f_{bs}(x^a_0), \]  

(B8)

using the initial condition Eq. (14) at \( \tau = u = 0 \). From Eq. (B4), we see that the matrix exponential appearing here must be time ordered, with earlier “times” \( u' \) appearing to the right of later ones.

It now remains to express \( f_{bs}(u) \) in terms of \( x^a(u) \) and \( \tau(u) = u \). We fix a \( \hat{u} = \hat{\tau} \) and call \( \hat{x}^a = x^a(\hat{u}) \). Using Eq. (B7)
with \( u_2 = u' \) and \( u_1 = u \) then shows that the \( x^s \)-solution Eq. (B6) can be expressed in terms of \( \hat{x} \) as
\[
x^s(u') = \phi_b(\hat{x}^s, \hat{t} - u')
\]  
and, in particular, \( x^s_0 = \phi_b(\hat{x}^s, \hat{t}) \), so
\[
f_{bs}(\hat{x}^s, \hat{t}) = \sum_b \left( e^{\int_{u}^{u'} dt' I(\phi_b(\hat{x}^s, \hat{t} - u'))} \right) \hat{f}_{bs} \phi_b(\hat{x}^s, \hat{t}).
\]  
Changing the integration variable to \( t' = \hat{t} - u' \) and dropping the hats then gives the solution Eq. (18) announced in the main text. Note that as \( t' = \hat{t} - u' \), the time ordering of the matrix exponential,
\[
E(\tau) = \exp \left( \int_{0}^{\tau} dt' I(\phi_b(x^s, t')) \right),
\]  
is such that the earlier \( t' \) are now on the left. The appropriate time-ordered matrix exponential is defined formally via its Taylor series,
\[
E(\tau) = 1 + \sum_{n=1}^{\infty} \int_0^{\tau} \prod_{i=1}^{n} \left( d\tau_i I(\phi_b(x^s, \tau_i)) \right) \times \cdots \times I(\phi_b(x^s, \tau_n)),
\]  
with \( I \) the identity matrix and the integration in the other terms running over the range \( 0 < \tau_1 < \cdots < \tau_n < \tau \).

**APPENDIX C: MAPPING OF SELF-CONSISTENT MEMORY TO DIFFERENTIAL EQUATIONS**

We show in this Appendix how to map the subnetwork equations with self-consistent memory,
\[
\frac{\partial}{\partial t} x_{s} = v_{s}(x^s(t)) + \mathcal{N}_s(t),
\]  
to a set of differential equations. The self-consistent memory term \( \mathcal{M}_s(t) \) is given by Eq. (24),
\[
\mathcal{M}_s(t) = \sum_b \int_0^{t} \left( c_b(x^s(t'))(e^{\int_{u}^{u'} dt' \tau(\hat{x}^s(t'))} ) \right) \hat{f}_{bs}^{0} \phi_b(x^s(t)),
\]  
so can be written as
\[
\mathcal{M}_s(t) = \sum_b m_b(t) \hat{f}_{bs}^{0} \phi_b(x^s(t)),
\]  
with
\[
m_b(t) = \int_0^{t} \left( c_b(x^s(t'))(e^{\int_{u}^{u'} dt' \tau(\hat{x}^s(t'))} ) \right) \hat{f}_{bs}^{0}.
\]  
It is then straightforward to check that
\[
\frac{d}{dt} m_b(t) = c_b(x^s(t)) + \sum_b m_b(t) l_{b/b} \phi_b(x^s(t)),
\]  
where the second term arises from the \( \tau \) dependence of the matrix exponential. The \( m_b(t) \) can therefore be obtained numerically by integrating the differential Eq. (C5) together with the subnetwork equations with (self-consistent) memory:
\[
\frac{d}{dt} x_s(t) = v_s(x^s(t)) + \sum_b m_b(t) \hat{f}_{bs}^{0} \phi_b(x^s(t)).
\]  
The appropriate initial conditions for the auxiliary variables follow from Eq. (C4) as \( m_b(0) = 0 \).

**APPENDIX D: CHANNEL DECOMPOSITION**

We begin by writing the expression for the memory function explicitly, combining Eqs. (14), (18), (20), and (21):
\[
M_s(x^s, \tau) = \sum_{b' \neq b''} (J^{-1})_{b''b} \frac{\partial R_{b''}}{\partial x_{b'}} R_{b'} f_{bs}(x^s, \tau),
\]  
\[
= \sum_{b' \neq b''} (J^{-1})_{b''b} \frac{\partial R_{b''}}{\partial x_{b'}} \sum_c c_{b''c} (\tau) \hat{f}_{bc}^{0} (\phi_b(x^s, \tau)),
\]  
\[
= \sum_{b' \neq b''} (J^{-1})_{b''b} \frac{\partial R_{b''}}{\partial x_{b'}} \sum_c c_{b''c} (\tau) \hat{R}_{bc} (\phi_b(x^s, \tau)),
\]  
where the first three factors are evaluated at \( x^s \). We now swap index labels and group the sums into a more intuitive form:
\[
M_s(x^s, \tau) = \sum_{s} \sum_{b} \left( \sum_{b'} (J^{-1})_{b''b} (x^s) \frac{\partial R_{b''}}{\partial x_{b'}} (x^s) R_{b'} (x^s) \hat{E}_{b'} (\tau) \frac{\partial R_{b'}(\phi_b(x^s, \tau))}{\partial x_{b'}} \right).
\]  
As discussed in the main text, the expression up to before the exponential represents a change in the deviation of the bulk species concentration \( x_{b'} \) from its QSS values over some small time interval, in response to changes in the subnetwork concentrations \( x_s \) [see also Eq. (14) below]. In the factor \( \frac{\partial R_{b''}}{\partial x_{b'}} \), only those bulk species \( b'' \) contribute whose time evolution depends explicitly on the subnetwork species \( s' \) driving the bulk time evolution via \( R_{b'} \). The \( b'' \) can then be interpreted as outgoing channels for the signal from \( s' \). After propagation in the bulk network, the signal returns via another bulk species. Here only bulk species \( b \) contribute that appear explicitly in the time evolution of subnetwork species \( s \) as indicated by the factors \( \frac{\partial R_{b''}}{\partial x_{b'}} \). The \( b \) can therefore be interpreted as incoming channels. Overall, we have memory effects from \( s' \) onto \( s \), via an outgoing channel \( (s' \to b') \) and an incoming channel \( (b \to s) \). Consistent with this interpretation, the outgoing channel "susceptibilities" \( \frac{\partial R_{b''}}{\partial x_{b'}} \) are evaluated for the past, i.e., sending, state \( x^s = x^s(t') \) of the subnetwork. The incoming channel susceptibilities \( \frac{\partial R_{b'}}{\partial x_{b'}} \), on the other hand, are evaluated at the current time \( t \) as shown by the propagation via \( \phi_b \) across the time difference \( \tau = t - t' \). Within the self-consistent approximation Eq. (C5), this
APPENDIX E: SELF-CONSISTENT CHANNEL DECOMPOSITION

The channel decomposition of Sec. D can also be applied to the self-consistent memory approximation, as we now outline. Writing out the self-consistent memory term Eq. (C2) explicitly and reordering and relabelling terms as in Eq. (D4) gives

\[ \tilde{M}_s(t) = \int_0^t dt' \sum_{x'} \sum_{b'} \sum_{b''} \left( \frac{\partial R_s}{\partial x_b} \right) (x_b(t')) \left( e^{\theta s' \cdot \partial_{x_b} t(x_b(t'))} \right)_{b' b''} \]

\[ \times (J^{-1})_{b' b''} (x_b(t')) \frac{\partial R_{b''}}{\partial x_s} (x_s(t')) R_{s'} (x_s(t')) \]

\[ = \sum_x \sum_{b b'} \frac{\partial R_s}{\partial x_b} (x_b(t')) m_{s b} (t), \]

(E1)

where

\[ m_{s b} (t) = \int_0^t dt' \sum_{b'} (J^{-1})_{b b'} (x_b(t')) \frac{\partial R_{b'}}{\partial x_s} (x_s(t')) R_{s'} (x_s(t')) \]

\[ \times (e^{\theta s' \cdot \partial_{x_b} t(x_b(t'))})_{b' b''}. \]

(E2)

From this last representation, it follows that the \( m_{s b} (t) \) vanish at \( t = 0 \) and obey the differential equations:

\[ \frac{d}{dt} m_{s b} (t) = (J^{-1})_{b b'} (x_b(t')) \frac{\partial R_{b'}}{\partial x_s} (x_s(t')) R_{s'} (x_s(t')) \]

\[ + \sum_{b'} m_{s b'} (t) l_{b' b}(x_b(t)). \]

(E3)

The channel-decomposed memory can therefore also be calculated from differential equations (see Appendix G). Of course, one only needs to find the \( m_{s b} \) for combinations \((s b)\) and \((b' s')\), where the corresponding channel susceptibilities are nonzero.

APPENDIX F: EXACTNESS OF MEMORY

We show that the self-consistent memory \( m_b (t) \) is exact when both \( R_s \) and \( R_b \) contain at most linear terms in \( x^b \). In such a case, the full system can be written as

\[ R_s = v_s + \sum_{b'} \tilde{x}_{b'} f^0_{b s}, \quad R_b = \sum_{b'} J_{b b'} \tilde{x}_{b'}. \]

(F1)

where \( \tilde{x}_{b'} = x_b - x^0 (x^s) \) and the QSS value \( x^0 (x^s) \) is an arbitrary function of \( x^s \). We now want to show that the \( \tilde{x}_{b'} \) correspond exactly to the \( m_b \) from the self-consistent ZMs method. To do this, we work out their evolution in time:

\[ \frac{d}{dt} \tilde{x}_{b'} = R_b - \sum_x R_{b x} \frac{\partial x^0_x}{\partial x_s}. \]

\[ = \sum_{b'} J_{b b'} \tilde{x}_{b'} + \sum_{b'} \left( v_{b'} + \sum_{b''} \tilde{x}_{b''} f^0_{b' b''} \right) \sum_{b''} (J^{-1})_{b' b''} \frac{\partial R_{b''}}{\partial x_s}. \]

(F2)

By using that for an \( x^b \)-linear system as assumed here, one has \( \partial_{x_b} = \partial_{x_b} v_s \), the above can be rewritten as

\[ \frac{d}{dt} \tilde{x}_{b'} = \sum_{b'} (J^{-1})_{b b'} \frac{\partial R_{b'}}{\partial x_s} v_{b'} \]

\[ + \sum_{b'} \tilde{x}_{b'} \left( J_{b b'} + \sum_{b''} (J^{-1})_{b' b''} \frac{\partial R_{b''}}{\partial x_s} \frac{\partial R_{b'}}{\partial x_{b''}} \right). \]

(F3)

Using then the definitions Eqs. (16) and (21), we obtain an expression equivalent to Eq. (C5),

\[ \frac{d}{dt} \tilde{x}_{b'} = c_{b'} + \sum_{b''} \tilde{x}_{b''} l_{b' b''}, \]

(F4)

thus showing that \( \tilde{x}_{b'} = m_b \) when we start from the same initial condition \( \tilde{x}_s = 0 \), i.e., the bulk at QSS.

We test the above exactness statement on two different examples that have a linear dependence on a particular species but nonlinear dependencies on other species: a minimal bistable system as described in Ref. [30], and the Brusselator, which is capable of achieving limit cycles [31]. As expected from the above derivation, the self-consistent memory captures the behavior of both systems exactly (Figs. 9 and 10). As further shown in Fig 10, the original nonlinear projection method ZMn is also accurate at capturing the dynamics though not necessarily exact. The corresponding memory function is shown in Fig.11. (We note that the memory functions of the Brusselator grow exponentially in a way that forces memory terms to cancel out to zero at the fixed point; this leads to numerical challenges that we do not pursue here.)

APPENDIX G: LINEAR DYNAMICS

We discuss briefly the case of fully linear dynamics, where the dependence of \( R_s \) and \( R_b \) on all variables \( x^s \) and \( x^b \) (not...
FIG. 10. Minimal bistable system with linear dependence on the second species as described in Ref. [30]; we choose $x_1$ for the subnetwork and place the second species in the bulk. Parameters for the bistable regime in this example are $k_1 = 10$, $k_2 = 1$, $k_3 = 2$, and $k_4 = 1$ [30]. The trajectory of the self-consistent ZMs projection again captures that of the original system (solid line) exactly (see Appendix F and Fig. 9) so is not plotted. G-QSS represents the Gouasmi et al. approach adapted to QSS projection (Appendix I). (a) $x_1(0) = 1.4$. At initial conditions near the fixed point, both the ZMn method (cyan dots) and G-QSS (orange line) behave similarly and accurately capture the full dynamics. (b) $x_1(0) = 11.9$. Further away from the final stable fixed point, the G-QSS predictions become increasingly inaccurate while the ZMn method continues to provide a good approximation.

To illustrate the linearized dynamics approach, we perform a channel decomposition of the amplitude (value at $\tau = 0$) of the linearized memory in the neural tube system as we did in Ref. [27], but now for the method derived in this study (Fig. 12). We find similar profiles to those found in Ref. [27]. The results highlight the relative weakness of the memory from Olig2 into Nkx2.2 via Pax6, supporting the conclusions of Ref. [27]. In addition, channel decomposition for the ZM trajectory shown in Fig. 6 provided memory terms affecting Nkx2.2 and Olig2 (Fig. 13). The method derived in this study is, however, significantly more powerful as it does not rely on an expansion near a steady state and gives access to the full memory and its channel decomposition as described in Appendixes D and E.

APPENDIX H: COMPARISON WITH ALTERNATIVE MEMORY FUNCTION APPROXIMATION

Gouasmi et al. [18] propose an approximation for the memory function for the case where the projection Eq. (10) is defined not by setting the bulk coordinates to their $x^s$-dependent QSS values, but simply to zero:

$$E[g(\cdot)|x^s] = g(x^s, 0). \quad (H1)$$

FIG. 11. Memory functions for the system detailed in Ref. [30] with the parameters chosen for Fig. 10. (a) Using the ZMn. (b) Using the method from Gouasmi et al. as extended to QSS projection in Appendix I. The $x$ axis shows the concentration of the subnetwork species $x_1$ while the $y$ axis indicates time difference $\tau$. By construction, the two memory function approximations predict the same value (scale bar to the right) at $\tau = 0$, as they only differ in how they propagate the memory over time. At $\tau > 0$, the memory functions are relatively similar for $x_1 \in [0, 6]$ but become progressively different as $x_1$ grows beyond this range; for $x_1 \geq 10$, the G-QSS method predicts a negative memory function for all $\tau$ that leads to its poor performance as observed in Fig. 10.
The function $F_s(x^s, x^b, \tau)$ still evolves according to Eq. (5), which written out now reads
\[
\frac{\partial}{\partial \tau} F_s(x^s, x^b, \tau) = LF_s(x^s, x^b, \tau) - E[L F_s(\cdot, \tau)|x^s],
\] (H2)
\[
= \sum_{s'} R_{s'}(x^s, x^b) \frac{\partial F_s}{\partial x_{s'}} + \sum_b R_b(x^s, x^b) \frac{\partial F_s}{\partial x_b}
\]
\[
- \sum_{s'} R_{s'}(x^s, 0) \frac{\partial F_s}{\partial x_{s'}}(x^s, 0, \tau),
\]
\[
- \sum_b R_b(x^s, 0) \frac{\partial F_s}{\partial x_b}(x^s, 0, \tau),
\] (H3)
where the very last factor is the $x_b$ derivative of $F$ evaluated at $x^b = 0$. The approximation of Ref. [18] amounts to ignoring the fact that the derivatives of $F_s$ are evaluated at a different point in the last two lines, which gives
\[
\frac{\partial}{\partial \tau} F_s = \sum_{s'} [R_{s'}(x^s, x^b) - R_{s'}(x^s, 0)] \frac{\partial F_s}{\partial x_{s'}}
\]
\[
+ \sum_b [R_b(x^s, x^b) - R_b(x^s, 0)] \frac{\partial F_s}{\partial x_b}.
\] (H4)

This has the form of a Liouville equation as noticed in Ref. [18] and so its solution can be written as
\[
F_s(x^s, x^b, \tau) = F_s(\psi^s(x^b, \tau), \psi^b(x^s, x^b, \tau)),
\] (H5)
where the components of the vector functions $\psi^s$ and $\psi^b$ evolve with $\tau$ according to
\[
\frac{\partial}{\partial \tau} \psi^s = R_s(\psi^b) - R_s(\psi^s, 0),
\] (H6)
\[
\frac{\partial}{\partial \tau} \psi^b = R_b(\psi^s, \psi^b) - R_b(\psi^b, 0),
\] (H7)
from the initial conditions
\[
\psi_s(x^b, 0) = x_b, \quad \psi_b(x^s, x^b, 0) = x_s.
\] (H8)

The function $F_s$ at $\tau = 0$, which as before we write without a time argument, is given by the analog of Eq. (12):
\[
F_s(x^s, x^b) = R_s(x^s, x^b) - R_s(x^s, 0).
\] (H9)

The corresponding memory as defined in Eq. (19) is
\[
M^{(s)}(x^s, \tau) = \sum_{s'} R_{s'} \frac{\partial}{\partial x_{s'}} F_s(\psi^b, \psi^s) + \sum_b R_b \frac{\partial}{\partial x_b} F_s(\psi^s, \psi^b),
\] (H10)
where all $R_e$, $R_b$ and the derivatives are evaluated at $(x^e, 0)$. Gouasmi et al. propose to find these derivatives numerically, but in fact a closed form expression can be obtained as follows. Applying the chain rule gives

$$M^G(x^e, \tau) = \sum_{s, t} R_e \frac{\partial F_e}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^e_t} + \sum_{s, t} R_b \frac{\partial F_b}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^b_t}$$

$$+ \sum_{s, t} R_b \frac{\partial F_b}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^b_t}.$$  \hspace{1cm} (H11)

Now note that in the final evaluation we always use $x^b = 0$, which from the differential Eqs. (H6) and (H7) implies $\psi^b = x^e$, $\psi^b = 0$ for all $\tau$. Hence, in particular $\psi_s$ is independent of $x^s$ and so $\partial \psi_s / \partial x^s = 0$. We also have $F_e(x^e, 0) = 0$ from Eq. (H9), which implies $\partial F_e / \partial \psi_s = 0$. Only the last term from Eq. (H11) thus survives,

$$M^G(x^e, \tau) = \sum_{s, t} \frac{\partial F_e}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^e_t}.$$  \hspace{1cm} (H12)

and it remains to find $\partial \psi_s / \partial x^b_t$. By differentiating Eq. (H7) for $\partial \psi_s / \partial \tau$ w.r.t. $x^b_t$, one finds

$$\frac{\partial \psi_s}{\partial x^b_t} = \frac{\partial R_b}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^b_t}.$$  \hspace{1cm} (H13)

On the r.h.s., a similar term from the variation of $\psi^b$ vanishes because it would be proportional to

$$\frac{\partial R_b}{\partial \psi_s} \frac{\partial \psi_s}{\partial \psi^b} = \frac{\partial R_b}{\partial \psi_s} \frac{\partial \psi_s}{\partial x^b_t}.$$  \hspace{1cm} (H14)

This difference is zero in the final evaluation at $x^b = 0$ (which implies $\psi^b = 0$). For the same reason, the derivatives $\partial R_b / \partial \psi_s = \partial R_b / \partial x^b_t$ are evaluated at $(x^e, 0)$ and constant in time $\tau$. Collecting these derivatives into a matrix $k$ with elements $k_{b'b'}$ and using that $\partial \psi_{s,b} / \partial x^s_{b'} = \delta_{bb'} (= 1$ for $b = b'$ and $= 0$ otherwise) at $\tau = 0$ gives then as the explicit solution of Eq. (H13),

$$\frac{\partial \psi_s}{\partial x^b_t} = (e^{k\tau})_{b'b}.$$  \hspace{1cm} (H15)

and inserting into Eq. (H12) yields

$$M^G(x^e, \tau) = \sum_{s, t} \frac{\partial R_b}{\partial \psi_s} (e^{k\tau})_{b'b} R_{b'b},$$  \hspace{1cm} (H16)

where we have used that $\partial F_e / \partial \psi_s = \partial F_b / \partial x^b_t = \partial R_b / \partial x^b_t$; this derivative and the factor $R_b$ are evaluated at $(x^e, 0)$ in the approximation from Ref. [18] for the memory function.

We do not show here how the above memory approximation performs in our test systems because the nature of the approach can lead to fixed points appearing after projection or new fixed points appearing. We observed both of these effects in numerical evaluations for the bistable switch from Ref. [30].

**APPENDIX I: EXTENDING GOUASMI ET AL. APPROXIMATION WITH QSS PROJECTION**

The Gouasmi et al. approximation [18] for the memory function rests on projecting to $x^b = 0$, but this is not generally an appropriate baseline for our case as it would correspond to setting all bulk concentrations to zero. However, we can adapt the approximation to the spirit of our work by changing coordinate system so that zero bulk coordinates correspond to the origin (under the approximation to the spirit of our work by changing coordinates $x^s$ to $x^s' = \sum_s c_s x^s$)

$$\tilde{x}_s = x_s, \hspace{1cm} \tilde{x}_b = x_b - x_s'(x^s) \hspace{1cm} (11)$$

because $\tilde{x}_b = 0$ is then equivalent to $x_b = x_s'(x^s)$. The time evolution of the new variables follows as

$$\frac{d}{dt} \tilde{x}_s = R_s(\tilde{x}_s, \tilde{x}_b^0) = R_s(\tilde{x}_s' + x_s^b, x_s^b + \tilde{x}_b^0),$$  \hspace{1cm} (12)

$$\frac{d}{dt} \tilde{x}_b = R_b(\tilde{x}_s', \tilde{x}_b^0),$$  \hspace{1cm} (13)

$$= R_b(\tilde{x}_s', x_b^s + \tilde{x}_b^0),$$  \hspace{1cm} (14)

where the factors enclosed in square brackets are the explicit expression for $-\partial x_s'/\partial x_s$ and have to be evaluated at $x_s'$. The Gouasmi memory function approximation, adapted for our QSS projection, is now given by Eq. (H16) applied to the new variables $\tilde{x}_s, \tilde{x}_b$ and corresponding drift functions $R_s, R_b$. The last factor is $R_b(\tilde{x}_s', 0)$, which can be read off from Eq. (I4). The first term in Eq. (I4) vanishes as $R_b = 0$ at QSS, while the remainder is seen to be precisely $c_s(x^s)$ from Eq. (21). The matrix $k$ in the new variables has elements

$$k_{b'b} = \frac{\partial R_b}{\partial \tilde{x}_b} = \frac{\partial R_b}{\partial x^b_t} = \sum_s \left[ \sum_{b'} \left( j^{-1} \right)_{b'b} \frac{\partial R_b}{\partial x_s} \right] \frac{\partial R_{b'b}}{\partial \tilde{x}_b} = l_{b'b}.$$  \hspace{1cm} (15)

Note that the terms in square brackets are already just dependent on $x^s$, so do not contribute to the derivative. The remaining factor in the memory function is, again in the new variables,

$$\frac{\partial \tilde{R}_b}{\partial \tilde{x}_b} = \frac{\partial R_b}{\partial x^b_t}.$$

so, overall,

$$\tilde{M}^G(x^e, \tau) = \sum_{b'b'} \frac{\partial \tilde{R}_b}{\partial \tilde{x}_b} \left( e^{k\tau} \right)_{b'b} R_{b'b}.$$  \hspace{1cm} (H17)

with

$$\tilde{f}_{b'b}(x^e, \tau) = \left( e^{d(x^e)} \right)_{b'b} f_{b'b}^0(x^e),$$  \hspace{1cm} (18)

where we have used the definition of $f_{b'b}^0$ from Eq. (14). Comparing now Eqs. (18) and (19) shows that the memory approximation Eq. (17), though derived here from rather different arguments, is quite similar to our expression Eq. (20): The only difference is that the propagation from $x^s$ to $\phi_s(x^e, \tau)$ is absent in $\tilde{f}_{b'b}$, which is the analog of our $f_{b'b}$. We show that without the $\phi_s$ propagation, the method can still perform accurately in some situations but breaks down in other settings (Fig. 10).
APPENDIX J: NEURAL TUBE MODEL

We detail the model taken from Ref. [24] used to model ventral neural tube patterning. The equations are as follows:

\[
\frac{d}{dt} x_P = \frac{\alpha_P w_{\text{p}}}{w_{\text{p}} + (1 + k_{\text{PO}}x_O)^2(1 + k_{\text{PN}}x_N)^2} - \beta_P x_P, \quad (J1)
\]

\[
\frac{d}{dt} x_O = \frac{\alpha_O w_{\text{o}} (1 + k_{\text{O,in}}x_{\text{in}})(1 + k_{\text{OI}}x_I)^2(1 + k_{\text{ON}}x_N)^2}{(1 + k_{\text{ON}}x_N)^2} - \beta_O x_O, \quad (J2)
\]

\[
\frac{d}{dt} x_N = \frac{\alpha_N w_{\text{N,p}} (1 + k_{\text{NI}}x_{\text{in}})(1 + k_{\text{NP}}x_P)^2(1 + k_{\text{NO}}x_O)^2(1 + k_{\text{NO}}x_N)^2}{(1 + k_{\text{NO}}x_O)^2} - \beta_N x_N, \quad (J3)
\]

\[
\frac{d}{dt} x_I = \frac{\alpha_I w_{\text{i,p}} (1 + k_{\text{IO}}x_O)^2(1 + k_{\text{IN}}x_N)^2}{w_{\text{I,p}} + (1 + k_{\text{IO}}x_O)^2(1 + k_{\text{IN}}x_N)^2} - \beta_I x_I. \quad (J4)
\]

The parameters and their meaning in a biological sense are detailed as follows:

| Name | Meaning | Value |
|------|---------|-------|
| \(\alpha_P\) | Pax6 production rate | 2 |
| \(\alpha_O\) | Olig2 production rate | 2 |
| \(\alpha_N\) | Nkx2.2 production rate | 2 |
| \(\alpha_I\) | Irx3 production rate | 2 |
| \(\beta_P\) | Pax6 degradation rate | 2 |
| \(\beta_O\) | Olig2 degradation rate | 2 |
| \(\beta_N\) | Nkx2.2 degradation rate | 2 |
| \(\beta_I\) | Irx3 degradation rate | 2 |
| \(k_{\text{PO}}\) | Olig2 binding to Pax6 DNA | 1.9 |
| \(k_{\text{PN}}\) | Nkx2.2 binding to Pax6 DNA | 26.7 |
| \(k_{\text{ON}}\) | Nkx2.2 binding to Olig2 DNA | 60.6 |
| \(k_{\text{OI}}\) | Irx3 binding to Olig2 DNA | 28.4 |
| \(k_{\text{NP}}\) | Pax6 binding to Nkx2.2 DNA | 4.8 |
| \(k_{\text{NO}}\) | Olig2 binding to Nkx2.2 DNA | 27.1 |
| \(k_{\text{NI}}\) | Irx3 binding to Nkx2.2 DNA | 47.1 |
| \(k_{\text{IO}}\) | Olig2 binding to Irx3 DNA | 58.8 |
| \(k_{\text{IN}}\) | Nkx2.2 binding to Irx3 DNA | 76.2 |
| \(w_{\text{p}}\) | Polymerase binding to Pax6 DNA | 3.84 |
| \(w_{\text{o}}\) | Polymerase binding to Olig2 DNA | 2.01263 |
| \(w_{\text{N,p}}\) | Polymerase binding to Nkx2.2 DNA | 0.572324 |
| \(w_{\text{I,p}}\) | Polymerase binding to Irx3 DNA | 18.72 |
| \(k_{\text{O,in}}\) | Gli (Shh signal) binding to Olig2 DNA | 180 |
| \(k_{\text{N,in}}\) | Gli (Shh signal) binding to Nkx2.2 DNA | 373 |

The signal input concentration \(x_{\text{in}}\) is the gradient \(e^{-p/0.15}\), which depends on the dorsal-ventral neural tube position \(p\) ranging from 0 to 1 as in Ref. [24].