9 Time series from a Nonlinear Dynamical Systems Perspective

Nonlinear dynamics is a *huge* field in mathematics and physics, and we will hardly be able to scratch the surface here. Nevertheless, this field is so tremendously important for our theoretical understanding of brain function and time series phenomena that I felt a book on statistical methods in neuroscience should not go without discussing at least some of its core concepts. Having some grasp of nonlinear dynamical systems may give important insights into *how the observed time series were generated*. In fact, nonlinear dynamics provides a kind of universal language for mathematically describing the *deterministic* part of the dynamical systems generating the observed time series – we will see later (sect. 9.3) how to connect these ideas to stochastic processes and statistical inference. ARMA and state space models as discussed in sects. 7.2 & 7.5 are examples of discrete-time, *linear* dynamical systems driven by noise. However, linear dynamical systems can only exhibit a limited repertoire of dynamical behaviors, and typically do not capture a number of prominent and computationally important phenomena observed in physiological recordings. In the following, we will distinguish between models that are defined in discrete time (sect. 9.1), as all the time series models discussed so far, and continuous-time models (sect. 9.2).

9.1 Discrete-Time Nonlinear Dynamical Systems

Discrete time dynamical systems are formally *maps* which 'map' the state of the system at some discrete point in time t to that at the next time step t+1. More specifically, this section treats maps consisting of difference equations defined through time-recursive relationships of the general form

$$\mathbf{x}_{t+1} = F(\mathbf{x}_t)\,, \tag{9.1}$$

where F is any linear or nonlinear function (giving rise to a *linear* or *nonlinear map*). Note that as defined in (9.1), the system's state at time t+1 through F depends only on the state at the previous time step t: In a physical system, a dependence on more than one previous time step would usually indicate that there are variables missing from our description. Moreover, as we have seen in sect. 7.2.1, AR(p) models with p>1 can always be recast in terms of a higher-dimensional VAR(1) model by including lagged values into the current state vector (a principle called 'delay embedding', to be discussed in sect. 9.4, which, under certain conditions, enables to replace missing system variables in models defined directly in terms of the observables). The same would also be possible for higher-order nonlinear maps.

The brief introduction to nonlinear dynamical systems given in sect. 9.1.1 below, as well as that for continuous-time systems in sect. 9.2, will frequently draw on the excellent presentation in Strogatz (1994). Strogatz (1994) provides a highly readable introduction into nonlinear dynamics that is also accessible to non-mathematicians, and is recommended for further details on the subject. We will start with a discussion of basic concepts and properties of nonlinear dynamical systems, before returning to the topic of statistical inference.

9.1.1 Univariate maps and basic concepts

For introduction, let us return to an AR(1) model, *omitting the noise term* for now ,

(9.2) $x_{t+1} = \alpha x_t + c$,

which constitutes a simple linear, univariate (scalar) map. The value we have for x_0 is called the *initial condition* of the system. Fig. 9.1 gives the time graph and the (first-)return plot (x_t, x_{t+1}) = (x_t, $F(x_t)$) of (9.2) for α=0.5, c=1. Along this line, there is one point which deserves special attention, given by the intersection of $F(x_t)$ with the bisection line. For function values located exactly on the bisectrix we have x_{t+1}= $F(x_t)$= x_t, hence the output of F equals its input at these points. Generally, any points x^* for which we have $x^* = F(x^*)$ are called *fixed points* of the map. If the system is placed exactly in one of these points, then, in the absence of noise, it will stay there indefinitely. In the present linear example, we easily obtain the fixed point analytically by solving $x^* = \alpha x^* + c$ for x^*, giving $x^*=c/(1-\alpha)$ (provided $\alpha \neq 1$). For the parameters used in Fig. 9.1, this is just the single point x^*=2, as confirmed graphically.

A fixed point can be *stable*, *unstable*, or *neutrally stable*, meaning that the system dynamics either *converges* to it in its vicinity, *diverges* from it along at least one direction, or neither one, respectively. For a linear map like (9.2) a fixed point will be stable if and only if $|\alpha|<1$. In this case, the point x^* is also called a *fixed point attractor* as it attracts nearby system states. Visually, this can be illustrated through the idea of a 'cob web' (Fig. 9.1, right, in red): Start at some point x_0, usually not too far from x^*, e.g. x_0=0.5 in Fig. 9.1 (right), and draw a straight line up to the respective function value $F(x_0)$. From there move horizontally to the bisectrix, as illustrated, which will give point x_1. Move up vertically again to obtain $F(x_1)$, then horizontally to the bisectrix to yield x_2, and so forth. This process can be iterated until we clearly see that with successive iterations we converge into fixed point x^*. In Fig. 9.1 you can repeat this process starting either from the left of x^*, $x_0 < x^*$, or from the right, $x_0 > x^*$, and you will see that in either case you will finally end up in x^* where the system will be stuck. Now let us turn to the case $|\alpha|>1$. Playing the same game again, this time you will see that even if you start close to x^*, the state *trajectory* will diverge from x^* and rush off to infinity. Thus x^* is an *unstable* fixed point (or repeller).

For the special case α=1 and $c\neq 0$, the function $F(x_t)$ will run parallel to the bisectrix and has no fixed point solution. For α=1 and c=0, the function $x_{t+1} = \alpha x_t + c$ will (obviously) coalesce with the bisectrix and we have an infinite line of fixed points, also called a *line attractor*. Fixed points on this line are *neutrally stable* in the sense that there is neither convergence nor divergence along this direction: If you drive the system from some x_0 a bit to the right or the left, it will simply stay at that new position without moving any further on its own under action of map F.

From this discussion we can see that a linear map is *stationary* (in the limit $T \rightarrow \infty$), as defined in sect. 7.1, if and only if it has a *stable fixed point*, as only in this case it will resist perturbations and always return to its stable mean. If $|\alpha|\geq 1$, on the other hand, x_t would either randomly walk around if driven by noise (α=1 and $c_t \sim N(0,\sigma^2)$), with no 'force' opposing perturbations (as in Fig. 7.4, right), or will even actively be driven to +/- infinity ($|\alpha|>1$; cf. Fig. 7.4, center). Thus, the formal conditions for a fixed point x^* of a linear map to be stable, and for an AR(p) process to be stationary, are exactly the same: The absolute slope of the line in the univariate case, or within the direction of maximum slope of the (hyper-)plane in the multivariate case, has to be less than one (i.e., $|\alpha|$ or the largest absolute eigenvalue of coefficient matrix **A**, respectively, needs to be <1).

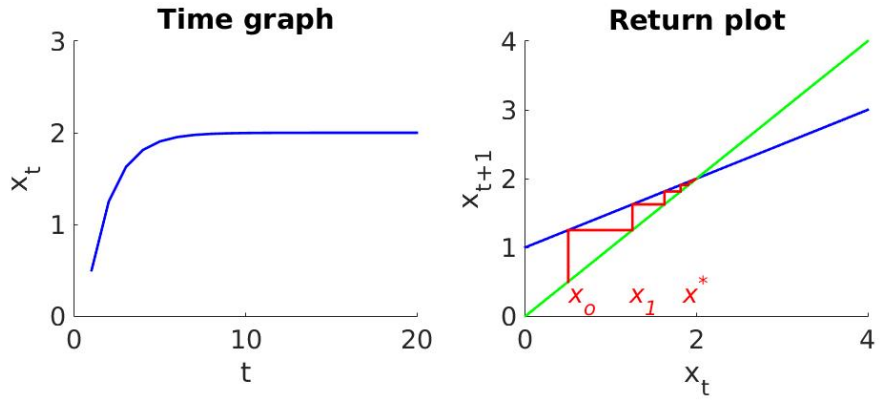

Fig. 9.1. Time graph (left) and return plot with cobweb (right) for linear map (9.2). F(x_t) in blue, bisectrix in green, cobweb in red. **MATL9_1**.

Line attractors have played an important role in neuroscience for explaining a variety of experimental phenomena, from the ability of goldfish to maintain any arbitrary eye position (Seung et al. 2000; Aksay et al. 2001), the ability to retain continuously valued variables in working memory (Machens et al. 2005), to animals' ability to predict interval times across various scales (Durstewitz 2003). In all these cases the property of line attractors that they provide a continuum of fixed points is exploited: A stimulus, like the frequency of tactile flutter (Machens & Brody 2005), may place the system at any point on the line which then would be stably maintained even after removal of the stimulus. Of course drift produced by noise in the system would degrade the memory over time. In the single-neuron timer model of Durstewitz (2003), slight displacement of the line attractor is used to generate flows with effectively arbitrarily slow time constants, thus providing a neural representation of arbitrary temporal intervals. (Strictly, the timer model, like the other models cited above, is a *continuous time* dynamical system [to be covered in sect. 9.2] where the property is exploited that in the vicinity of a just barely destabilized attractor state the flow will be very slow, documenting the attractor's former existence. However, similar phenomena can also be produced in discrete time dynamical systems, e.g. if we choose α=1 and c very small in eq. 9.2, generating an x_t series with tiny increments at each time step.) Almost linear ramping in neural firing rates, with a slope adapting to the temporal interval between a cue and a subsequent response-triggering stimulus, has been observed in many different brain areas and tasks (Quintana & Fuster 1999; Komura et al. 2001; Brody et al. 2003). The neural timer model has been advanced as a neurodynamical explanation for these empirical observations.

Drift-diffusion models of decision making (Ratcliff 1978; Ratcliff & McKoon 2008; usually formulated in continuous time, although discrete time versions have been set up as well) highlight another important application of linear dynamical systems in the neutrally stable (no drift) or unstable regime. When expressed in discrete time, drift-diffusion-type models may be thought of as linear (AR-like) maps, like $x_{t+1} = \alpha x_t + \beta S + \varepsilon_t$, which integrate binary stimulus information $S \in \{-1, +1\}$ over time, where for $|\alpha|<1$ we have a 'leaky integrator', and for α=1 a perfect integrator. The animal may decide S=+1 was presented when $x_t > \theta_{up}$, i.e. x_t crosses an upper threshold θ_{up}, and S=-1 when x_t falls below a lower threshold θ_{low}. Indeed, during experiments with binary noisy sensory stimuli, neurons were observed in parietal cortex that ramp up their activity almost linearly (as with α close to 1) with stimulus viewing time until the decision (Mazurek et al. 2003; Huk & Shadlen 2005). The slope of the ramping activity was a function of the uncertainty (signal strength) in the stimulus, suggesting that less uncertainty (or larger signal strength) can be captured in the model by a larger β weight. A similar model was recently entertained by Brunton et al. (2013) to differentiate different (stimulus and intrinsic) noise sources in the decision

making process.

With the basic concepts introduced above, we turn our attention now to an example for a *nonlinear* map, the logistic map (originally introduced to describe population growth, and popularized in its nonlinear dynamical aspects by May 1976), defined as

$$x_{t+1} = F(x_t) = \alpha x_t(1-x_t)\,,\; x_0 \in [0,1]\,,\; \alpha \in [0,4]\,. \tag{9.3}$$

Obviously, $F(x_t)$ is a quadratic function in x_t, hence nonlinear. Given the stated conditions on α and x_0, x_t will stay bounded within the interval [0, 1] forever. Fig. 9.2 shows time graphs and return plots for three different values of parameter α. For α=0.7 (Fig. 9.2, top) there is just a single fixed point x^*=0 (which is always a fixed point of eq. 9.3 for any α), while for α=2 or α=3 we would get two fixed points located at x^*=0, and x^*=1/2 or x^*=2/3, respectively. More formally, we can solve for the fixed points of the logistic map by requiring

$$\begin{aligned} x^* = F(x^*) = \alpha x^*(1-x^*) \;&\Rightarrow\; \alpha x^{*2} + (1-\alpha)x^* = 0 \\ &\Rightarrow\; x^*_{1/2} = -\frac{(1-\alpha)}{2\alpha} \pm \sqrt{\frac{(1-\alpha)^2}{4\alpha^2}} \in \left\{0, \frac{\alpha-1}{\alpha}\right\}. \end{aligned} \tag{9.4}$$

So we see that x^*=0 is always a fixed point of the map, regardless of the value for α, while a second fixed point exists in the range [0,1] only for α>1.

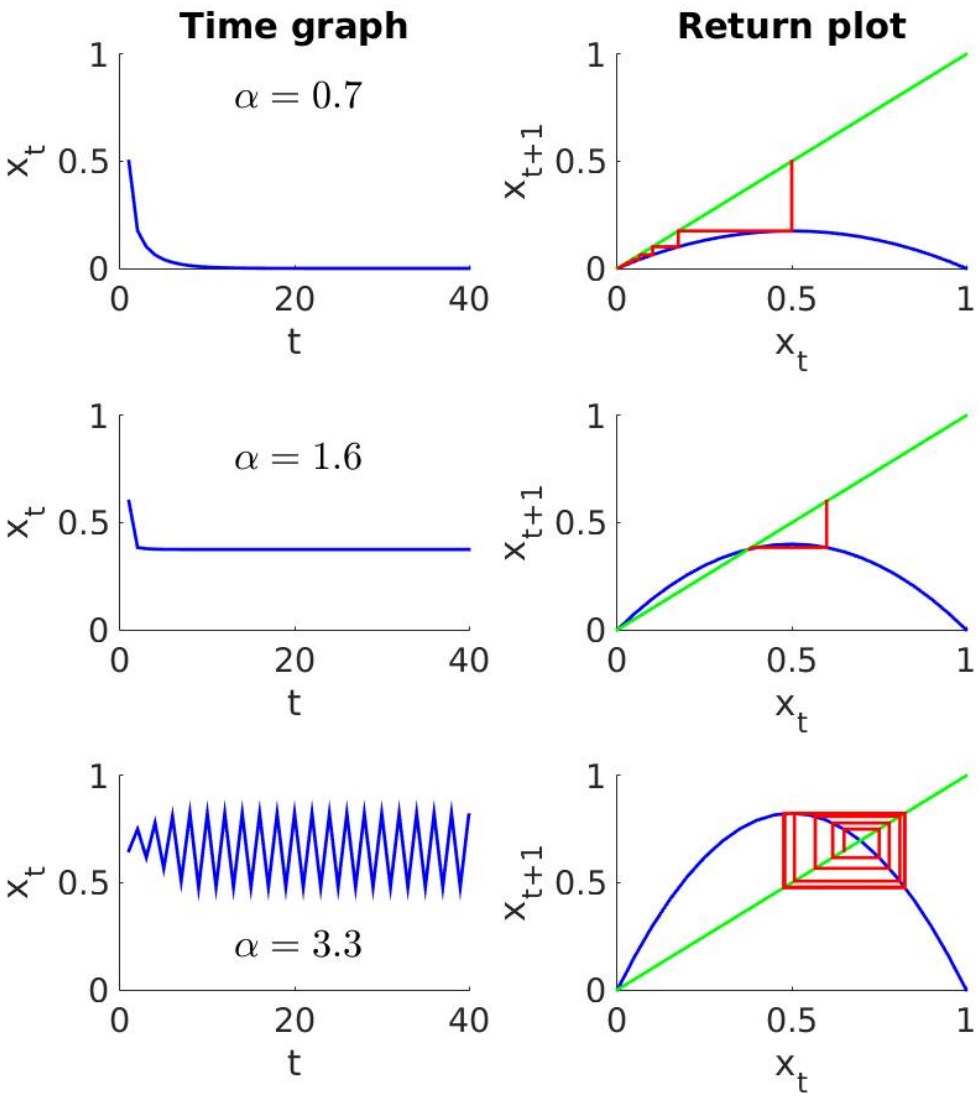

Fig. 9.2. Time graphs (left column) and return maps (right column) of logistic series with α = 0.7 (top row), 1.6 (center), 3.3 (bottom). Blue curve in return plots gives logistic function, bisectrix in green, cobweb (see text) in red. **MATL9_2**.

For the nonlinear map eq. 9.3, the conditions for stability outlined above now apply to the *local* slope of map $F(x)$ in the vicinity of the fixed point: If $|dF(x^*)/dx^*|<1$ at fixed point x^*, the fixed point will be *locally stable* (while a fixed point of a linear map for $|\alpha|<1$ will always be *globally stable*). For $0 \leq \alpha < 1$, logistic map eq. 9.3 has one *globally stable* fixed point at $x^*=0$, as can be verified by either following the cob web (Fig. 9.2, top-right) or evaluating the derivative

$$\frac{dF(x_t)}{dx_t} = \alpha - 2\alpha x_t \tag{9.5}$$

at $x^*=0$. For $\alpha>1$, a second fixed point appears at $x^*=(\alpha-1)/\alpha$, that is initially within infinitesimally small distance from 0. The critical parameter $\alpha_c=1$ at which this second fixed point shows up is called a *bifurcation* point of the system, in this case a *transcritical bifurcation* (Strogatz 1994). For $1<\alpha<3$, map eq. 9.3 has one *globally stable* fixed point

located somewhere (depending on the precise value of α) on the upper branch of $F(x)$, while the fixed point at $x^*=0$ is now *unstable* (Fig. 9.2, center).

The conditions for a fixed point to be stable can also be derived more formally by considering small perturbations ε in the vicinity of the fixed points (Strogatz 1994): If the fixed point is unstable, the perturbations should grow in time, while they should shrink if x^* is stable. Tracking perturbation ε across time one may write (Strogatz 1994)

(9.6) $x^* + \varepsilon_{t+1} = F(x^* + \varepsilon_t) \approx F(x^*) + \varepsilon_t F'(x^*) \quad \Rightarrow \quad \varepsilon_{t+1} \approx \varepsilon_t F'(x^*)$ since $x^* = F(x^*)$ by definition of a fixed point.

The approximation in (9.6) represents a Taylor series expansion around x* in which we have kept only linear (first order) terms, and defined $F'(x)=dF(x)/dx$. If ε is sufficiently small, higher powers (orders) of ε should become vanishingly small compared to the term linear in ε, so that this linear approximation should become valid in the vicinity of x^*. From (9.6) one sees directly that the perturbation ε will grow if $|F'(x^*)|>1$, and will decay away if $|F'(x^*)|<1$, proving the conditions for stability of a fixed point. For $|F'(x^*)|=1$, higher order terms will become relevant in the Taylor expansion eq. 9.6 and the linear approximation (also called a *linearization* around x^*) is no longer valid for determining stability (see Strogatz 1994).

What if $|F(x^*)|>1$ at *all* fixed points of map (9.3)? In that case there are no stable fixed points anymore the map would converge to, yet x_t would *still stay bounded* within the interval [0,1] if $x_0 \in [0,1]$ and $4 \geq \alpha \geq 0$. So where does it go? In fact, for $3.4>\alpha>3$, the system settles into an *oscillation* as illustrated in Fig. 9.2 (bottom), i.e. cycles through in this case two values x_1^* and x_2^* in the limit $t \rightarrow \infty$. This *p-cycle* of order 2 is itself a *stable* object in the system's state space, called a *period 2-cycle attractor*: Whether the system is started outside the cycle, $x_0 > 0.83$ or $x_0 < 0.47$ for $\alpha=3.3$, or within it, $x_0 \in [0.48, 0.82]$, it will always converge to the indefinitely self-repeating sequence (x_1^*, x_2^*). Formally, one may prove that sequence (x_1^*, x_2^*) is indeed a stable *p*-cycle by considering fixed points of the two-times iterated map $F^2(x):=F(F(x))$ (Strogatz 1994). A 2-cycle of map $F(x)$ must be a fixed point of map $F^2(x)$, since $x_1^*=F(F(x_1^*))$ and $x_2^*=F(F(x_2^*))$. If the two fixed points of map $F^2(x)$ are stable, then the 2-cycle of map $F(x)$ must be stable as well. The map

(9.7) $F(F(x_t)) = F^2(x_t) = \alpha[\alpha x_t(1-x_t)](1-[\alpha x_t(1-x_t)]) = -\alpha^3 x_t^4 + 2\alpha^3 x_t^3 - (\alpha^2+\alpha^3)x_t^2 + \alpha^2 x_t$

is a 4-th order polynomial in x_t, as shown in Fig. 9.3. From Fig. 9.3 we sense that the local slopes at fixed points x_1^* and x_2^* of map $F^2(x)$ are of absolute value less than 1, hence stable.

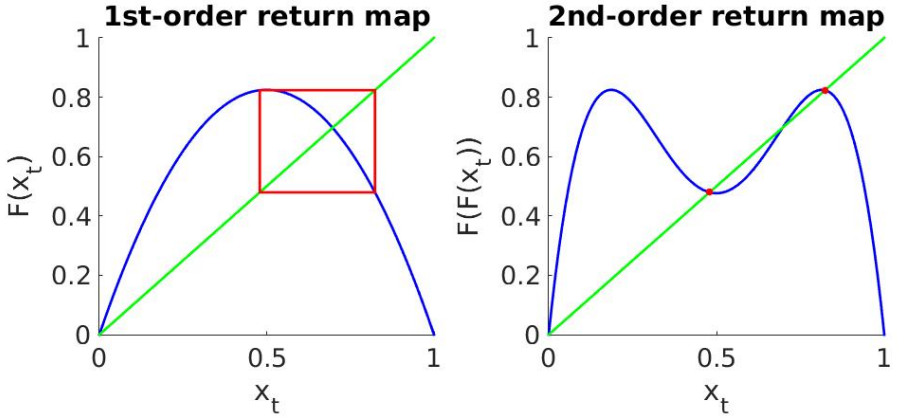

Fig. 9.3. First and second-order return maps of logistic function for α = 3.3, illustrating 2-cycle in first-order map as fixed points of second-order map. See also related discussion and illustrations in Ch. 10 of Strogatz (1994), and in May (1976). **MATL9_3**.

As we keep on increasing α, the map undergoes a series of bifurcations with which the order of the stable cycle will increase further and further (Fig. 9.4, 9.5). Really weird stuff starts to happen once α>3.57: The system's state trajectory starts to densely fill a whole region within the range $x \in$[0.33, 0.9], never precisely repeating itself (Fig. 9.4, center). At this point the *p*-cycle disappears and another stable object appears. In contrast to the completely *periodic*, *regular* nature of a period *p* cycle which will ultimately always precisely retrace itself, however, this new geometrical object is *aperiodic*, *irregular*. Although this irregular time series may appear quasi-random, it is not the result of a noise source into the system, but it is a characteristic of the completely deterministic system eq. 9.3 for α=3.9. This phenomenon has therefore been labeled *deterministic chaos*, and the geometrical objects in state space associated with it, just like fixed points or *p*-cycles, can be either stable or unstable – in the former case it is also called a *strange or chaotic attractor*.

A hallmark of chaotic deterministic systems is their high sensitivity to even minuscule perturbations or differences in initial conditions (the famous 'butterfly effect', as named so by Edward Lorenz). In Fig. 9.4 (bottom), two time series simulated according to model (9.3) with α=3.9 are illustrated with the initial condition x_0 differing just by an order of 10^{-4} (i.e., x_0=0.5 for the red trace and x_0=0.5001 for the blue). Despite this only slight difference in x_0, the two time series quickly *diverge*, unlike time series in the stable *p*-cycle regime would do. The observations from Figs. 9.2 & 9.4 are summarized in the *bifurcation graph* of the logistic map in Fig. 9.5: It shows the distribution of all visited stable states x_t (i.e., after convergence to the attractor) as a function of (control) parameter α, and thus makes explicit at which α values transitions (bifurcations) toward new dynamical behaviors occur.

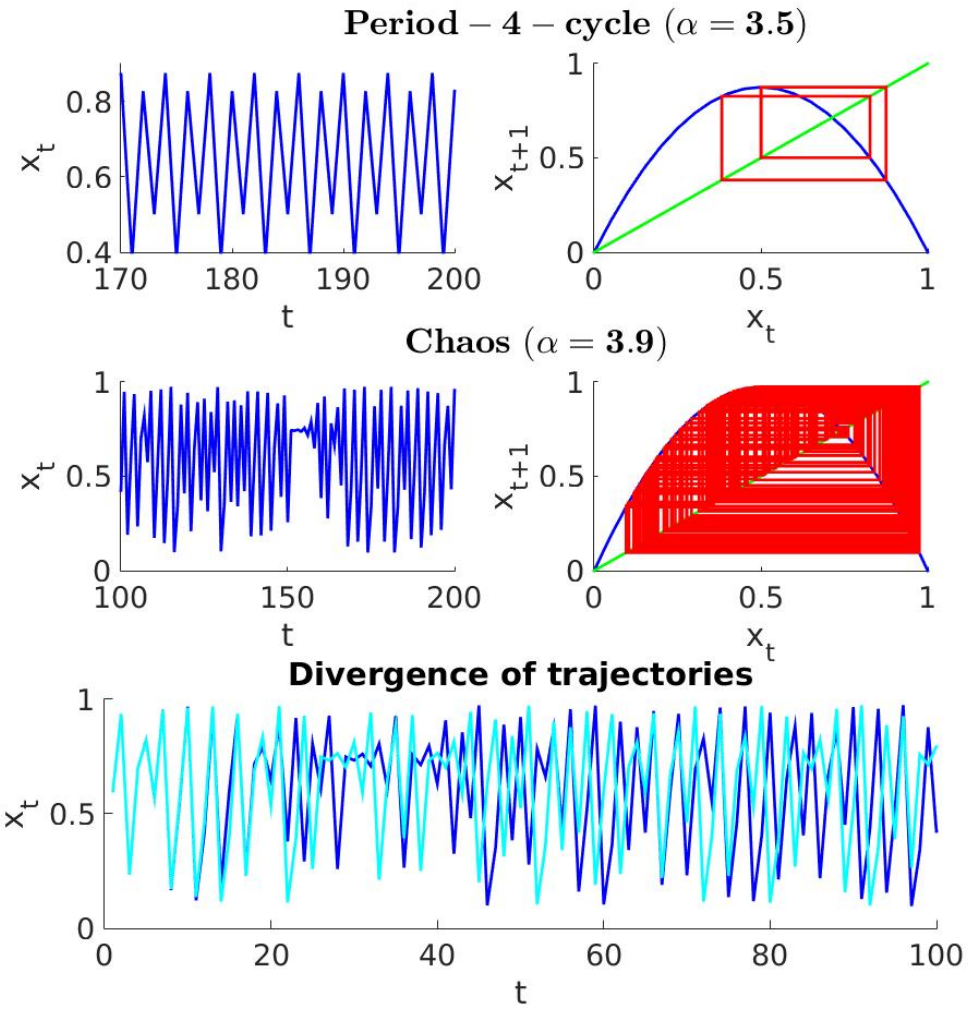

Fig. 9.4. Higher-order cycles (α = 3.5, top row) and chaos (α = 3.9, center) in the logistic map. Bottom graph illustrates quick divergence of time series in chaotic regime for just slightly different (by an ε of 10^{-4}) initial conditions. **MATL9_4**.

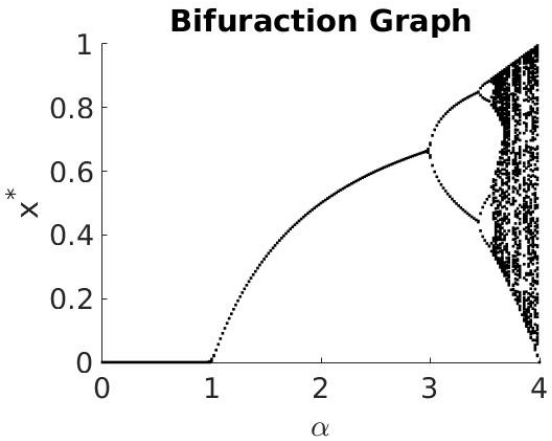

Fig. 9.5. Bifurcation graph of the logistic map. Only stable objects are shown. **MATL9_5**.

There are several important take-home messages here: First, even a completely deterministic system can produce pseudo-random behavior that in many real world situations may actually be difficult to discern from truly probabilistic behavior. Second, although time series generated by chaotic attractors are irregular and highly sensitive to perturbations, they are still spatially confined within some bounded region of state space,

as these objects are still attractors. Hence, unlike sometimes suggested by folklore, chaotic systems may still be statistically *predictable* to a degree given by the spatial extent of the attractors, although on the attractor itself prediction will require exponentially finer knowledge about initial conditions as more steps into the future are to be considered (the *prediction horizon* is determined by the Lyapunov exponent, to be introduced in sect. 9.2.1, eq. 9.34). Third, chaotic systems may produce quite complex distributions as they dwell in many different spots of state space which, paired with their high sensitivity to noise and perturbations, could make it potentially difficult to establish stationarity in the statistical sense as defined in 7.7 or 7.8 (sect. 7.1), at least if only short series are available, albeit the parameters of the system may actually be time-invariant. Fourth, the chaotic nature of a system could make estimating its parameters quite difficult as the LSE or likelihood landscapes of these systems can become quite complicated, fractal, packed with local minima (cf. Fig. 1.4; Abarbanel 2013; Wood 2010). It seems, chaotic systems pose some problems for statistical analysis – even more so as they are likely to be the rule rather than the exception in biology.

9.1.2 Multivariate maps and recurrent neural networks

Now that we have introduced a number of concepts and phenomena in nonlinear dynamics using the univariate logistic map eq. 9.3, we are well equipped to move over to more interesting – from the perspective of neuroscience – multivariate models. We will introduce a general class of multivariate nonlinear maps here termed (discrete-time) *recurrent neural networks* (RNN, illustrated in Fig. 9.6; see Elman 1990; Williams & Zipser 1990; Hertz et al. 1991; RNNs can also be formulated in continuous time as considered in sect. 9.2). A (discrete-time) RNN consists of a collection of units $i = 1..M$ which are described by nonlinear difference equations of the form

$$(9.8) \quad x_{t+1}^{(i)} = G\left(w_{i0} + \sum_{j=1}^{N} w_{ij} x_t^{(j)} + \eta_t^{(i)} \right) ,$$

where the w_{ij} are called connection weights from units j to unit i, w_{i0} is a constant bias term (setting a kind of 'basic activity level' for each unit), $\eta_t^{(i)}$ is a (deterministic, known) external input to unit i at time t (but, in a statistical framework, could also represent a noise term), and G is a nonlinear but usually monotonic 'activation' function, commonly taken to be a sigmoid

$$(9.9) \quad G(y) = \left(1 + e^{-y}\right)^{-1} .$$

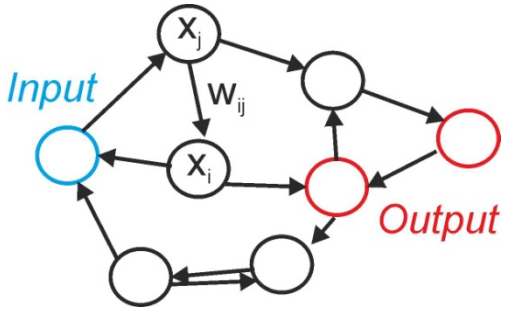

Fig. 9.6. Structure of a recurrent neural network (RNN), with one designated input and two output units. In general, the network can have arbitrary connectivity and different units may be designated as input and/or output units at different time steps.

The individual bias terms w_{i0} for each unit *i* may also be replaced by a common 'activation threshold' θ. In statistical language, (9.8) resembles a generalized AR model, with G^{-1} being the link function. More compactly, this class of models may be rewritten in matrix form as

$$\mathbf{x}_{t+1} = G\left(\mathbf{W}\mathbf{x}_t + \boldsymbol{\eta}_t\right), \tag{9.10}$$

where G operates element-wise. A bias term $\mathbf{w}_0$, if present, may be absorbed as usual into the matrix **W** by adding a leading 1 to column vector $\mathbf{x}_t$.

This model is generally powerful enough to reproduce (m)any kinds of deterministic multivariate time series $\{\mathbf{x}_t\}$, including phenomena like fixed points, *p*-cycles, and chaos (Li 1992; Funahashi & Nakamura 1993; Kimura & Nakano 1998; Chow & Li 2000). It needs to have sufficient degrees of freedom, however, that is the number of units *M* included in the RNN model may have to be larger than the number of variables *N* in the time series. We may designate some of the RNN's units as *output units* (Fig. 9.6) for which we like to have $x_t^{(i)} \overset{!}{=} \xi_t^{(i)}$, $i \le M$, with $\xi_t^{(i)}$ the desired output value for unit *i* at time *t*, while the remaining units serve as *hidden units* that may enable dynamics that the output units on their own would not allow for. Actually, we do not even have to specify for each output unit a desired output at each time step *t*; outputs may just be defined for a subset $\{t^*\} \subset \{1 \ldots T\}$ of target times which could be individual to each unit (de facto turning some of the output units into hidden units at other time steps). Finally, some of the units may explicitly serve as *input units* (Fig. 9.6) receiving inputs $\eta_t^{(i)}$ from the external world. Note that the distinction between input, output, and hidden units is merely conceptual and does not have to be fixed or unique – the same unit may serve different functions at different times. This makes this whole framework very flexible.

Parameter (connection weight) matrix **W** is commonly determined through numerical schemes like gradient descent on the LSE function (see Mandic & Chambers, 2001, for an extensive discussion of different learning algorithms, including recursive least squares [cf. Sussillo & Abbott 2009] and Kalman-filter-like procedures on the weights). Since we are dealing with regression on a vector $\mathbf{x}_t$ that is evolving through time, obtaining the derivatives is a bit more involved than in pure feedforward networks (sect. 2.8), albeit not really any more difficult in principle. Let us start by defining an error term $\varepsilon_t^{(i)}$ for each unit *i* at time *t* as (Hertz et al. 1991)

$$\varepsilon_t^{(i)} = \begin{cases} \xi_t^{(i)} - x_t^{(i)} & \textit{if an output was defined for unit i at time t} \\ 0 & \textit{otherwise} \end{cases}. \tag{9.11}$$

The total LSE function may then simply sum up all these individual error terms, squared, across all units and time:

$$Err(\mathbf{W}) = \frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}(\varepsilon_t^{(i)})^2 = \frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\mathrm{I}\{\varepsilon_t^{(i)} \neq 0\}(\xi_t^{(i)} - x_t^{(i)})^2, \tag{9.12}$$

where I is the indicator function, picking out terms corresponding to those units *i* in the error sum for which a desired output at time *t* was defined. Since error function (9.12) splits up linearly in time, the gradient of *Err*(**W**) w.r.t. parameters **W** does so as well and can be computed time-point-wise as (Williams & Zipser 1990; Hertz et al. 1991)

$$\frac{\partial Err_t(\mathbf{W})}{\partial w_{ij}} = -\sum_{k=1}^{N} \mathrm{I}\{\varepsilon_t^{(i)} \neq 0\}(\xi_t^{(k)} - x_t^{(k)}) \frac{\partial x_t^{(k)}}{\partial w_{ij}}$$

(9.13)

$$\text{with} \quad \frac{\partial x_t^{(k)}}{\partial w_{ij}} = G'\left(w_{k0} + \sum_{l=1}^{N} w_{kl} x_{t-1}^{(l)} + \eta_{t-1}^{(k)} \right) \left[\delta_{ki} x_{t-1}^{(j)} + \sum_{l=1}^{N} w_{kl} \frac{\partial x_{t-1}^{(l)}}{\partial w_{ij}} \right] \quad ,$$

where *G*' is the outer derivative of function *G* (with respect to the argument in eq. 9.13), and δ_{ki} is the Kronecker delta, taking on δ_{ki} = 1 when unit *k* is the receiving unit for that particular w_{ij} (i.e. *i*=*k*) and hence w_{ij} itself occurs in the sum within *G*, and δ_{ki} = 0 otherwise.

Note that eq. 9.13, lower part, defines a recursive relation in time for the derivatives $\partial x_t^{(k)} / \partial w_{ij}$: These depend only on immediately preceding values for $x_{t-1}^{(i)}$ and $\eta_{t-1}^{(i)}$, on the current parameters **W**, and on the derivatives $\partial x_{t-1}^{(k)} / \partial w_{ij}$ *computed in the previous time step*. Thus, the gradients $\partial x_t^{(k)} / \partial w_{ij}$ define a dynamical system by themselves, in principle subject to all the considerations in sect. 9.1. Starting from the reasonable initial condition $\partial x_1^{(k)} / \partial w_{ij} = 0$, the derivatives of the activity values w.r.t. the weights, just like the activation values $x_t^{(i)}$ themselves, can hence be updated and stored iteratively for computing the required weight changes (Pearlmutter 1989, 1990; Williams & Zipser 1990; Hertz et al. 1991). Finally, weights w_{ij} would then be updated moving against their local error gradient as

$$(9.14) \quad w_{ij}^{(new)} = w_{ij}^{(old)} + \gamma \Delta_t w_{ij} \quad , \quad with \;\; \Delta_t w_{ij} = -\frac{\partial Err_t(\mathbf{W})}{\partial w_{ij}} .$$

This updating could proceed 'online', i.e. time-step by time-step, in which case the gradient dynamics (9.13) would be directly coupled with the activity evolution through the weight changes. Or it could occur in 'batch-mode', summing up all changes across time first and then making one final change after a full run (Hertz et al. 1991). The process is then iterated until the total error score and/ or the weights converge. As for gradient descent methods in general (see sect. 1.4.1, 2.8), local minima may plague the process, and one of the simplest (but not necessarily most effective) remedies is to start the process from many different initializations for **W** to obtain a better approximation to the global minimum. Training may also be complicated by the fact that parameter changes can (or should!) drive the system across bifurcations: This may lead to sudden switches in the system's behavior, and hence steep error gradients as the system enters a different dynamical regime. Constructing bifurcation graphs along crucial directions in parameter space (e.g., derived from the eigenvalue spectrum of **W**) may provide some insight. In general, it may be difficult to get away with a constant learning rate γ in (9.14) if the error landscape exhibits strong local variations in gradients (as easily happens in these high-dimensional spaces), with very flat slopes in some areas and very steep ones in others. Adjusting gradients by the absolute values of the Hessian, similar as in the Newton-Raphson procedure (see sect. 1.4.1), may offer some help (various learning algorithms for RNNs, their convergence and stability issues, and remedies are discussed in detail in Mandic & Chambers 2001).

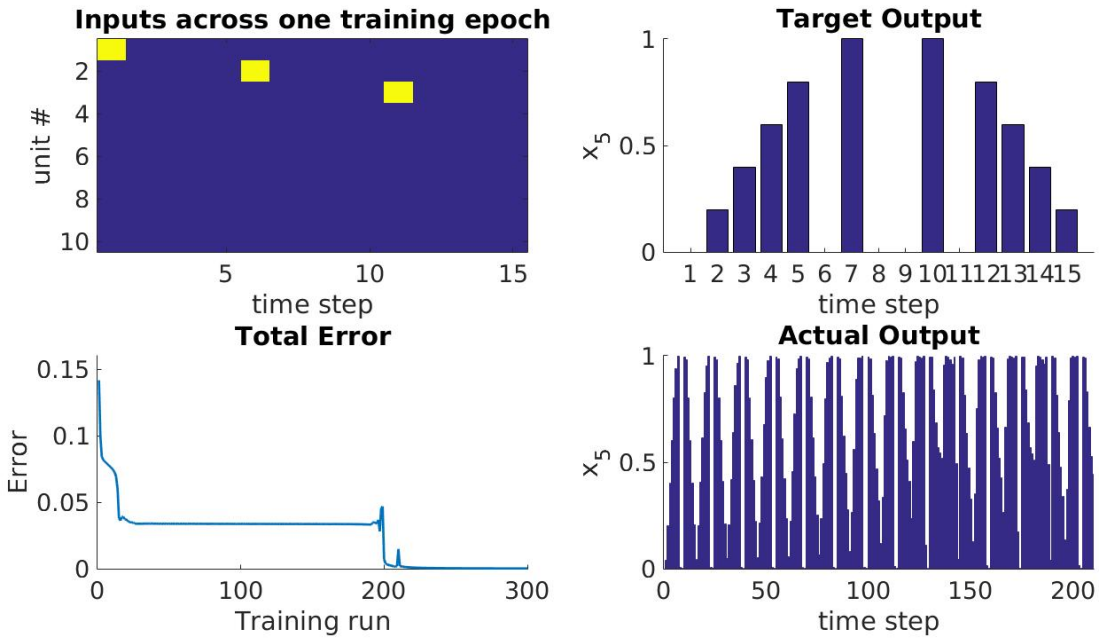

Fig. 9.7. Ten-unit RNN trained on an oscillatory output pattern with temporal gaps. Top left: Input matrix, showing the units (#1,2,3) and time steps (1, 6, 11) at which inputs (indicated by yellow squares) are received during one full training episode ('trial'). Top right: A waxing-waning output pattern with intermediate temporal gaps is requested at unit #5. Bottom left gives the error (9.12) as a function of the number of training episodes (i.e., with 15 time steps each). Note that the error function decays only slowly for longer time periods but then suddenly drops at three time points. Also note the brief positive excursions (or oscillations) near two of these three points (in particular the one around run #200). Rapid changes like these typically occur in the vicinity of bifurcation points at which the system may discretely hop into another dynamical regime, and where the training process becomes highly sensitive to even minimal parameter changes. Bottom right illustrates performance of free-running network (learning turned off) for 210 time steps. Note that initially the reproduction of the training pattern is quite accurate, but then slowly drifts off over time, indicating that the originally trained cycle is unstable. Including slight variations or perturbations around the target cycle in the training process, e.g. through a noise term, may potentially improve stability (Sussillo & Abbott 2009). **MATL9_6**.

Figs. 9.7 (**MATL9_6**) and 9.8 (**MATL9_7**) give several examples where an RNN has been trained to carry out different types of tasks or computations. For instance, in Fig. 9.8 an RNN has been trained to perform a simple two-cue working memory task (see also Zipser et al. 1993). Training RNNs efficiently is in itself a broader field (see Mandic & Chambers 2001), as often one may deal with very rugged error landscapes where the training process gets easily stuck in local minima. Methods like 'simulated annealing' which include probabilistic components so that at least in the limit $t \rightarrow \infty$ the global minimum is approached have been devised for such situations (Aarts & Korst 1988). A potential remedy for RNNs is to force them to stay on the desired output orbit during training by 'forcing' ('clamping') the output units to their target values (Pearlmutter 1990; Hertz et al. 1991; Abarbanel 2013). Thus, the error between true and predicted output is still computed for adjusting the weight parameters, but the respective unit is then explicitly set to the desired output value such that the system is forced back onto the correct cycle. This straightforward strategy helps to bias the space of potential (local) solutions towards the true global minimum by smoothing out the LSE landscape (Abarbanel 2013). Once trained, the stability of fixed points of a RNN and their bifurcations, and perhaps sometimes also that of simple limit cycles, may be assessed by locally linearizing the system as discussed

earlier in this section (see Beer 2006; although fixed points and cycles themselves may have to be determined numerically or graphically).

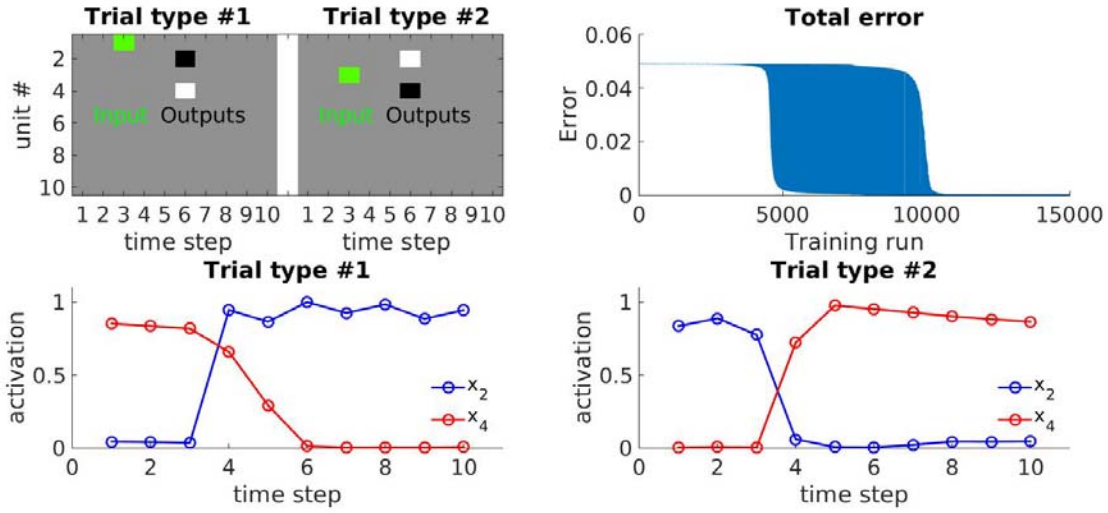

Fig. 9.8. Ten-unit RNN trained on a 'working memory task'. Task setup top left: When an input (green square) is given on unit #1 ('stimulus 1') the network is required to produce an output (of 1) on unit #2 but suppress activity on unit #4 (output of 0) three time steps later (trial type #1). Vice versa, unit #4 (1) but not unit #2 (0) is requested to produce an 'on'-response if unit #3 received an input ('stimulus 2') three time steps earlier (trial type #2). Top right graph gives the error as function of training episodes. Strong, slowly decaying oscillations like those present here could potentially indicate that the system is swept back and forth across a bifurcation. A constant (non-adaptive) learning rate γ in (9.14) appropriate for most of the training process, may become much too high around these sensitive points where error gradients become very steep. Around some bifurcations, the LSE function could exhibit step-like behavior, jumping to new values as the bifurcation point is passed. The reader is encouraged to further examine this. Bottom row shows outputs of units #2 (blue) and #4 (red) for the two different input conditions – as required, at time step 6 either unit #2 is active and unit #4 inactive (left), or vice versa (right). Details are provided in **MATL9_7**.

There are many powerful extensions to the basic RNN framework, some of which should at least be mentioned here. One of them is the concept of 'Long Short-Term-Memory' (LSTM) which allows RNNs to bridge very long delays and to learn long sequences through special gated linear ('integrator') units which maintain activity protected against interference (Hochreiter & Schmidhuber 1997). This is indeed a serious issue in training RNNs as formulated above by gradient descent (recall that error gradients themselves follow a dynamical process): Without a special purpose mechanism, information required for establishing links between temporally widely spaced inputs and outputs may decay too quickly for learning to pick up on them (see also Schmidhuber 2015). The maintenance of error information through constant 'error carousels' across arbitrarily long delays is also a crucial brick in many deep learning networks (Schmidhuber 2015) which start to equal or excel human performance in some domains (Mnih et al. 2015).

Another framework couples a large RNN with fixed (non-trainable) structure/ parameters, but many degrees of freedom and a lot of diversity in the unit's properties and connections, to a simple linear classifier ('readout'). Only the parameters (weights) of the latter are changed by training, such that the training process itself (by virtue of the linearity)

is simple and fast (Maass et al. 2002; Jaeger & Haas 2004; Bertschinger & Natschläger 2004; Sussillo & Abbott 2009). These systems rest on two core ideas: First, the dimensionality of the network is much larger than that of the input space, such that – similarly as with basis expansions in SVMs (sect. 3.5) – inputs are projected into a much higher-dimensional space (via nonlinear operations) which then allows for linear separability (hence, the linear readout is sufficient). Second, meta-parameters of the network are tuned such that it lives 'at the edge of chaos', that is right on a bifurcation from an 'ordered' (complex cycle) to a 'disordered' (chaotic) regime (Bertschinger & Natschläger 2004; Legenstein & Maass 2007; or slightly within the chaotic regime to begin with, see Sussillo & Abbott 2009). The idea is that the intrinsic system dynamic should neither be too simple such that it quickly forgets about the inputs (e.g., fast convergence to a fixed point), nor too chaotic such that it becomes overly sensitive to even tiny differences in inputs. Rather, the system should live in a complex regime that harbors rich temporal structure and possesses long memory time constants. This will ensue at the edge of chaos where the maximal Lyapunov exponent (see next section) is right around 0 (Legenstein & Maass 2007), as on a stable limit cycle or line attractor which also defines directions in state space along which memories of arbitrary inputs will dissipate only slowly (or not at all in the absence of noise).

Similarly to the example in Fig. 9.8, to gain insight into the neural dynamics underlying behavior, several researchers have trained recurrent neural networks on experimental protocols from tasks used with laboratory animals, and then compared unit activities within the trained network to those observed experimentally (e.g. Zipser et al. 1993; Mante et al. 2013). For instance, in Mante et al. (2013) an RNN training algorithm developed by Sussillo and Abbott (2009) was used to explain neural recordings from prefrontal cortex in a context-dependent psychophysical decision making task. The authors discovered that their RNN would develop context-dependent line attractors which could underlie the context-dependent integration of sensory evidence required in the task.

9.2 Continuous-time nonlinear dynamical systems

In this section, we switch from discrete to continuous time, and thus from systems of difference equations to systems of differential equations. In introducing the subject we will at first follow the highly recommended presentation in Strogatz (1994), before moving on to more neuroscientifically motivated examples (see also Wilson, 1999, and Izhikevich, 2007, for neuroscientifically oriented introductions into the field). In the time-continuous domain, a dynamical system is defined through a set of (ordinary) differential equations

$$\text{(9.15)}\quad \dot{\mathbf{x}} \equiv \frac{d\mathbf{x}}{dt} = \begin{bmatrix} dx_1/dt \\ dx_2/dt \\ dx_3/dt \\ \dots \end{bmatrix} = \begin{bmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ f_3(\mathbf{x}) \\ \dots \end{bmatrix} = f[\mathbf{x}(t)]$$

which gives the derivative of state $\mathbf{x}(t)$ with regards to time as a function f of the system's current state. To highlight the system's time-continuous nature, we write $\mathbf{x}(t)$ with $t \in \mathbf{R}$, in contrast to the notation $\mathbf{x}_t$ for discrete-time systems where t is just an integer-index. The system may in addition also *explicitly* depend on time,

(9.16) $\frac{d\mathbf{x}}{dt} = \dot{\mathbf{x}} = f[\mathbf{x}(t), t]$,

where this explicit time-dependence may represent some time-varying input (like an external stimulus) or so-called 'forcing' function into the system. Thus, model (9.16) is a *non-autonomous* while model (9.15) is an *autonomous* system. In principle, one can convert an *n*-dimensional non-autonomous into an (*n*+1)-dimensional autonomous system by including an additional differential equation of the form

(9.17) $x_{n+1} = t$, $\frac{dx_{n+1}}{dt} = 1$.

This may seem a bit like cheating, but formally results in a closed-loop system (Strogatz 1994).

Another point to be mentioned is that the system may contain not only first-order but also higher-order derivatives like d^2x/dt^2 or d^3x/dt^3, i.e. the second or third or even higher-order derivatives with respect to time. A p^{th}-order 1-dimensional ordinary differential equation (ODE) system, however, can always be converted into a first-order *p*-dimensional system (Strogatz 1994). As a simple example, consider the ODE system

(9.18) $\frac{dx}{dt} + a\frac{d^2x}{dt^2} + bx + c = 0$.

By defining $x_1 = x$, $x_2 = \dot{x}_1$, (9.18) may be rewritten as the equivalent first-order 2-dimensional system

(9.19)
$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{1}{a}(x_2 + bx_1 + c) \end{aligned}$$

Finally, the system may also contain derivatives with respect to other variables, e.g. space in addition to time, so that we would have to solve for a function $x(t,y)$ as in

(9.20) $\tau\frac{\partial x}{\partial t} + \lambda^2\frac{\partial^2 x}{\partial y^2} + \mu = 0$.

Such systems frequently occur in neuroscience – in fact, eq. 9.20 is the so-called cable equation in neural biophysics that describes passive current flow along dendritic structures with time constant τ and space constant λ (Koch 1999). These are known as *partial* differential equation (PDE) systems, but will not be covered in this book at all.

The space spanned by all the dynamical variables of an ODE or PDE system is called the system's *state space* (which can be infinite for some systems) – a point within this space uniquely and exhaustively identifies the current state of the system and its future if it is autonomous and deterministic. The set of points (states) within this space that corresponds to a specific realization of the time series $\mathbf{x}(t)$ starting from some initial condition $\mathbf{x}(0)$ is called a *trajectory* (e.g. green line in Fig. 9.11). Trajectories within this space can never intersect unless they terminate in the same limiting state/ orbit (see sect. 9.2.1 below): If they would intersect, that would imply the dynamics is not uniquely defined at the intersection point $\mathbf{x}_s$, violating the basic assumption of an autonomous deterministic

system. Finally, at each state **x**(*t*), the vector *d***x**(*t*)/*dt* of derivatives will point into the direction into which the system is about to move (as it specifies how the system's state changes with time), while the length of that vector specifies the local velocity of movement. The collection of such vectors is called the *flow field* of the system and explicates its dynamics throughout state space (cf. Fig. 9.11).

9.2.1 Review of basic concepts and phenomena in nonlinear systems described by differential equations

As in the discussion of maps, subsequent matters will be facilitated by introducing basic concepts along linear ODE systems first. For those readers not so familiar with solving differential equations (or just as a reminder), a single linear differential equation is solved straightforwardly by separation of variables as

$$(9.21)\quad \frac{dx}{dt} = \dot{x} = \alpha x \quad \Rightarrow \quad \int \frac{dx}{x} = \alpha \int dt \quad \Rightarrow \quad \log(x) + c = \alpha t \quad \Rightarrow \quad x(t) = c' e^{\alpha t},$$

where the integration constant *c´* is determined through the initial condition $x(0)=x_0$. Note that $x(t)$ decays away in time if $\alpha<0$ (such that $x^*=0$ is a *stable fixed point* of eq. 9.21), while it exponentially explodes for $\alpha>0$ (making $x^*=0$ an *unstable fixed point*). Transferring this to a *system* of linear differential equations

$$(9.22)\quad \dot{\mathbf{x}} = \mathbf{A}\mathbf{x},$$

we may propose the ansatz (see Strogatz 1994)

$$(9.23)\quad \mathbf{x}(t) = e^{\lambda t}\mathbf{v} \quad , \mathbf{v} \neq \mathbf{0}.$$

Differentiating both sides with respect to *t* one gets

$$(9.24)\quad \mathbf{A}(e^{\lambda t}\mathbf{v}) = \lambda e^{\lambda t}\mathbf{v} \quad \Rightarrow \quad (\mathbf{A} - \lambda \mathbf{I})\mathbf{v} = \mathbf{0}.$$

Thus, the solution is given in terms of an eigenvalue problem and can generally (assuming distinct eigenvalues $\lambda_j \neq \lambda_k$) be expressed as a linear combination of all the eigenvectors $\mathbf{v}_k$ of **A**:

$$(9.25)\quad \mathbf{x}(t) = \sum_{k=1}^{p} c_k e^{\lambda_k t}\mathbf{v}_k,$$

where the integration constants c_k are obtained from the initial condition

$$(9.26)\quad \mathbf{x}(0) = \sum_{k=1}^{p} c_k \mathbf{v}_k = \mathbf{x}_0.$$

Note that the eigenvalues λ_k may be complex numbers, i.e. consist of a real and an imaginary part. As the imaginary function $e^{ix} = \cos(x) + i\sin(x)$, where $i = \sqrt{-1}$ is the imaginary number *i*, can be rewritten in terms of a sine and a cosine function, the general solution may also be expressed as

$$(9.27)\quad \mathbf{x}(t) = \sum_{k=1}^{p} c_k e^{[\mathrm{re}(\lambda_k)+i\,\mathrm{im}(\lambda_k)]t}\mathbf{v}_k = \sum_{k=1}^{p} c_k e^{\mathrm{re}(\lambda_k)t}[\cos(\mathrm{im}(\lambda_k)t) + i\sin(\mathrm{im}(\lambda_k)t)]\mathbf{v}_k\ ,$$

from which we see that the linear ODE system (9.22) may have (damped, steady, or increasing) oscillatory solutions. In passing, we also note that an *inhomogeneous* system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}$ could be solved by a change of variables $\mathbf{z} = \mathbf{x} + \mathbf{A}^{-1}\mathbf{B}$, solving the *homogeneous* system $\dot{\mathbf{z}} = \mathbf{A}\mathbf{z}$ first, and then reinserting to solve for **x** (e.g. Wilson 1999).

The linear ODE system eq. 9.22 has a fixed point at **x*=0** (obtained by setting $\dot{\mathbf{x}} = \mathbf{0}$) with characteristic behavior in its vicinity given by the real and imaginary parts of the eigenvalues (Fig. 9.9). Specifically, in extension of the logic laid out above for a single ODE, the fixed point will be stable if the real parts of *all* its eigenvalues have $\mathrm{re}(\lambda_k) < 0$, while it will be unstable if *any one* of the distinct eigenvalues has $\mathrm{re}(\lambda_k) > 0$. The imaginary parts of the eigenvalues, in contrast, will determine how specifically the trajectory tends to approach the fixed point or strays away from it: If there is at least one eigenvalue with imaginary part $\mathrm{im}(\lambda_k) \neq 0$ (in fact, they will always come in pairs), the system has a damped *oscillatory* solution with the trajectory cycling into the point if stable, or spiraling away from it if unstable (Fig. 9.9). In this case the fixed point is called a *stable* or *unstable spiral point*, respectively, while it is called a *stable* or *unstable node* (Fig. 9.9), respectively, if all imaginary parts are zero, $\forall k : \mathrm{im}(\lambda_k) = 0$. If the system has eigenvalues with pure imaginary (i.e., zero real, $\forall k : \mathrm{re}(\lambda_k) = 0$) parts, it will be *neutrally stable* along those directions and generate a pure *sinusoidal* or *harmonic oscillation* (Fig. 9.10). Such a point is called a *center*. Note that because of the neutrally stable character, the amplitude of the oscillation will depend on where precisely we put the system in state space, and will be changed to new (neutrally) stable values by any perturbations into the system. The pure sinusoidal and neutrally stable character is a hallmark of *linear oscillators*. Finally, in case of a fixed point toward which the system's state converges along some directions but diverges along others (i.e. $\exists k : \mathrm{re}(\lambda_k) < 0 \wedge \exists l : \mathrm{re}(\lambda_l) > 0$), we are dealing with a so-called *saddle node*.

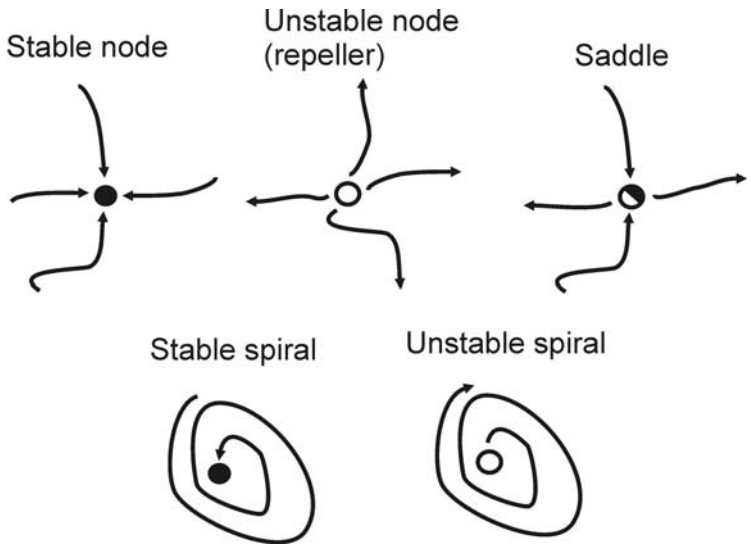

Fig. 9.9. Classification of different types of fixed points (see text).

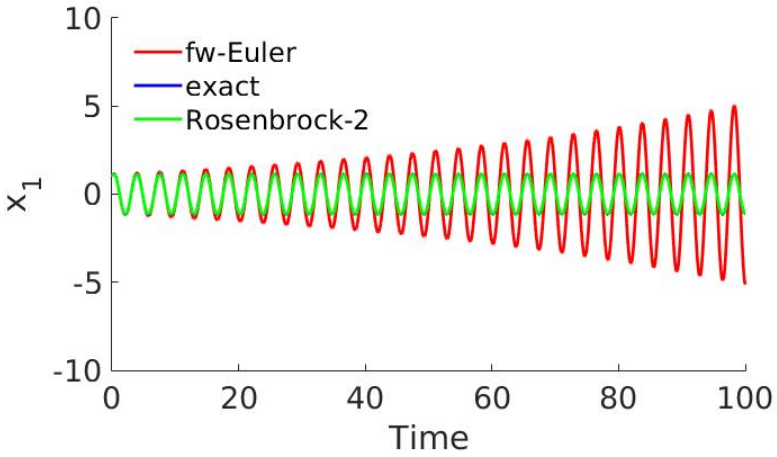

Fig. 9.10. Neutrally stable harmonic oscillations in linear ODE system. The graph illustrates that the choice of numerical ODE solver is important when integrating ODEs numerically: While the exact analytical solution (blue) and that of an implicit 2nd-order numerical solver (Rosenbrock-2, green) tightly agree (in fact, a blue curve is not visible since the green curve falls on top of it), a simple forward Euler scheme (red) diverges from the true solution. See **MATL9_8** for details.

We will now start the treatment of *nonlinear* ODE systems with a neuroscientifically motivated example (based on the Wilson-Cowan model; Wilson & Cowan, 1972, 1973), a 2-dimensional system consisting of a population of excitatory (pyramidal) neurons and one of inhibitory (inter-) neurons, each described by one ODE for the temporal evolution of the average population firing rates (cf. Wilson & Cowan 1972; Wilson 1999; Pearlmutter 1989, 1990):

$$\begin{aligned} \tau_E \dot{\nu}_E &= -\nu_E + \left[1 + e^{\beta_E (\theta_E - w_{EE}\nu_E + w_{IE}\nu_I - \eta_E(t))}\right]^{-1} \\ \tau_I \dot{\nu}_I &= -\nu_I + \left[1 + e^{\beta_I (\theta_I - w_{EI}\nu_E)}\right]^{-1} \end{aligned} \tag{9.28}$$

Note that this system comprises two feedback loops through the sigmoid-function terms in square-brackets: One positive feedback loop through recurrent self-excitation of the excitatory neurons with strength w_{EE}, and one negative via excitation of the inhibitory neurons which feed back inhibition through parameter w_{IE}. Term $\eta_E(t)$ could accommodate an external input to the excitatory neuron population, but we will assume it to be zero in most of the following. Slope parameters β_E, β_I, are included here more for conceptual reasons: At least β_I is, strictly, redundant in (9.28) (and should thus be fixed or omitted if such a model were to be estimated from empirical data). In the absence of any feedback or external input, firing rates would exponentially decay with time constants τ_E and τ_I, respectively.

Fig. 9.11 (center column) illustrates the 2-dimensional (ν_E, ν_I) *state space* of this system (also called a phase plane in this case) with parameter values {τ_E=30, τ_I=5, β_E=4, θ_E=0.4, β_I=1, θ_I=0.35, w_{EE}=1.15, w_{IE}=0.5, w_{EI}=0.5}, including its *flow field*, that is the direction of the temporal derivatives across a grid of state points. The red and blue lines highlight two special sets of points within this space, the system's *nullclines*, at which the temporal derivatives of one of the system's variables vanish. Thus, the blue curve gives the set of all points for which $\dot{\nu}_E = 0$ (the ν_E *-nullcline*) and the red curve the set of points for which $\dot{\nu}_I = 0$ (the ν_I *-nullcline*). For system (9.28) these nullclines can be analytically obtained by setting the derivatives to zero and solving for ν_E and ν_I, respectively:

(9.29)

$$\tau_E \dot{\nu}_E = -\nu_E + \left[1 + e^{\beta_E(\theta_E - w_{EE}\nu_E + w_{IE}\nu_I)}\right]^{-1} = 0 \quad \Rightarrow \quad \nu_I = \frac{1}{w_{IE}}\left[\frac{1}{\beta_E}\log(\nu_E^{-1} - 1) - \theta_E + w_{EE}\nu_E\right] = f_E(\nu_E)$$

$$\tau_I \dot{\nu}_I = -\nu_I + \left[1 + e^{\beta_I(\theta_I - w_{EI}\nu_E)}\right]^{-1} = 0 \quad \Rightarrow \quad \nu_I = \left[1 + e^{\beta_I(\theta_I - w_{EI}\nu_E)}\right]^{-1} = f_I(\nu_E)$$

Note that for mathematical convenience we have solved not only the second, but also the first equation in (9.29) for variable ν_I (instead of ν_E), as variable ν_I occurs only in the exponential term, while the first differential equation includes both linear and exponential functions of ν_E (cf. Strogatz 1994). For graphing these curves it obviously does not matter which variables we solve for – in this case one would chart the pairs (ν_E,$f_E(\nu_E)$) and (ν_E,$f_I(\nu_E)$) for the ν_E- and ν_I -nullclines, respectively.

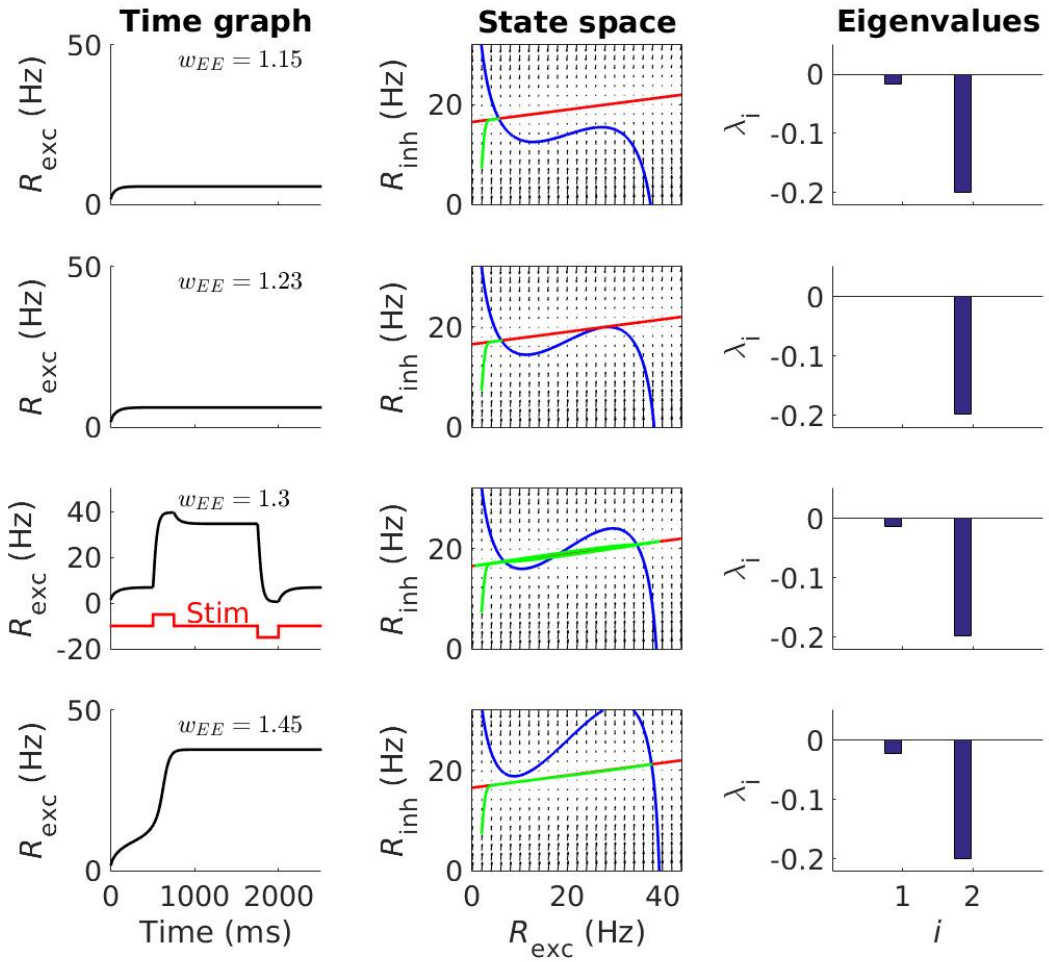

Fig. 9.11. Time graphs (left column), phase spaces (center column), and eigenvalues (right column) for system (9.28) for w_{EE} = 1.15, 1.23, 1.3, and 1.45 (from top to bottom). R_{exc}=40×ν_E, R_{inh}=40×ν_I. See **MATL9_9** for settings of other parameters and implementational details. For the simulation in the third row, a stimulus (illustrated by the red curve) was applied to switch the system among its two fixed points. ν_E-nullclines in blue in the phase portraits, ν_I-nullclines in red, system trajectory in green, flow field as black arrows (flow vectors become very tiny in the vicinity of the ν_I-nullcline when not normalized; see Fig. 9.12C for a clearer indication of direction of flow). Eigenvalues are the ones for the rightmost fixed point, with real parts in blue (imaginary parts are 0 in all cases shown here). **MATL9_9**.

The fixed points of the system are readily apparent as the intersection points of the

nullclines, since for those one has both $\dot{\nu}_E = 0$ and $\dot{\nu}_I = 0$. Furthermore, the plotted flow field already gives an indication of whether these are stable or not: The flow seems to converge at the lower left and upper right fixed point in Fig. 9.11 (3rd row; Fig. 9.12C), but to diverge from the fixed point in the center of the graph. In fact, the two nullclines divide the whole phase plane into regions of different flow, as the sign of the temporal derivative must change for a variable across its nullcline: To the right from the f_E-curve the flow is oriented to the left, while left from f_E it is oriented to the right. Likewise, above the ν_I-nullcline the ν_I variable tends to decay, while below it the flow points upwards. Thus, the beauty of such a state space representation is that it provides a whole lot of information and insight about the system dynamics in one shot.

To assess the stability and nature (node vs. spiral) of the fixed points precisely, one can follow a similar approach as outlined for nonlinear maps in sect. 9.1.1. That is, one would 'linearize' the system around the fixed points, setting up a differential equation for a small perturbation ε from the fixed point and approximating it by the linear terms in a Taylor series expansion (Strogatz 1994). For a *p*-dimensional nonlinear ODE system

$$\dot{\mathbf{x}} = \mathrm{f}(\mathbf{x}) = \left[f_1(x_1..x_p), f_2(x_1..x_p), ..., f_p(x_1..x_p)\right]^T \tag{9.30}$$

this leads to a matrix of first derivatives of the right hand sides $f_i(\mathbf{x})$ with respect to all its component variables x_i, i=1..p:

$$\frac{\partial \mathrm{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \partial f_1(\mathbf{x})/\partial x_1 & \partial f_1(\mathbf{x})/\partial x_2 & \partial f_1(\mathbf{x})/\partial x_3 & \cdots & \partial f_1(\mathbf{x})/\partial x_p \\ \partial f_2(\mathbf{x})/\partial x_1 & \partial f_2(\mathbf{x})/\partial x_2 & \partial f_2(\mathbf{x})/\partial x_3 & \cdots & \partial f_2(\mathbf{x})/\partial x_p \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ \partial f_p(\mathbf{x})/\partial x_1 & \cdots & \cdots & \cdots & \partial f_p(\mathbf{x})/\partial x_p \end{pmatrix}. \tag{9.31}$$

This is called the *Jacobian matrix*, and exactly as for the purely linear case (eq. 9.22), its eigenvalue decomposition has to be examined at the fixed points x^* to determine whether we are dealing with saddles, nodes ($\forall k : \mathrm{im}(\lambda_k) = 0$), or spirals ($\exists k : \mathrm{im}(\lambda_k) \neq 0$), and, in the latter two cases, whether they are stable ($\forall k : \mathrm{re}(\lambda_k) < 0$) or unstable ($\exists k : \mathrm{re}(\lambda_k) > 0$). For the parameter values given above (Fig. 9.11, 3rd row), it turns out that the lower left and upper right fixed points of eq. 9.28 are indeed stable nodes, while the one in the center is a (unstable) saddle node with *stable* and *unstable manifolds*. System (9.28) thus has *two fixed point attractors* which *coexist* for the same set of parameters, a phenomenon called *bi-stability* (or *multi-stability* more generally if several attracting sets co-exist). The unstable manifold of the saddle divides the phase plane into two regions from which the flow either converges to the one fixed point attractor or the other, called their *basins of attraction* (Fig. 9.12C). Thus, a basin of attraction is the set of all points from which the system's state will finally end up in the corresponding attractor.

Let us next explore how the system dynamic changes as parameters of the system are systematically varied. Fig. 9.11 illustrates the system's phase plane for four different values of parameter w_{EE}, the strength of the recurrent excitation (positive feedback) term. As evident in Fig. 9.11 (top row), for small w_{EE} the system has just one stable fixed point associated with low firing rates of both the excitatory and inhibitory populations. This makes sense intuitively – if self-excitation is not strong enough, the system will not be able to maintain high firing rates for longer durations. As w_{EE} is increased, the system undergoes a so-called *saddle node bifurcation* at which the f_E and f_I nullclines just touch

each other forming a new object, a single saddle node (Fig. 9.11, 2nd row). The saddle node bifurcation leads into bistability as w_{EE} is increased further (as was shown in Fig. 9.11, 3rd row). Finally, if self-excitation w_{EE} is increased to very high values, the lower rate stable fixed point vanishes (Fig. 9.11, bottom) in another saddle node bifurcation, coalescing with the center fixed point. Excitatory feedbacks in this simple network are now so strong that it will always be driven to high firing rates.

Plotting the set of stable and unstable fixed points against (control) parameter w_{EE} gives the *bifurcation graph* (Fig. 9.14, left): It summarizes the different dynamical regimes a system can be in as a function of one or more of the system's parameters. Stable objects are often marked by solid lines and unstable ones by dashed lines in such plots. From the bifurcation graph Fig. 9.14 (left) we see that system eq. 9.28 exhibits bistability only within a certain parameter regime, for $w_{EE} \in [1.23, 1.39]$.

Within this regime, the system exhibits an important property that engineers and physicists call *hysteresis*: An external stimulus or perturbation may switch the system from its low-rate attractor into its high-rate attractor or vice versa (Fig. 9.11, 3rd row). Once driven into the 'competing' basin of attraction, the system will stay there even after withdrawal of the original stimulus, unless another perturbation occurs. This hysteresis property is reminiscent of neural firing rate changes that have been observed in the prefrontal cortex during working memory tasks (Fig. 9.12A). In a working memory task, one of several cues is presented to the animal, followed by a delay period of usually one or more seconds during which no discriminating external information is present. After the delay is over, the animal is confronted with a stimulus display which offers several possible choices (Fig. 9.12A), with the correct choice depending on the previously presented cue. Since the cue may change with each trial and the task cannot be solved based on the information given in the choice situation alone, the animal has to maintain a short-term representation of it throughout the whole delay period. A hallmark finding in these tasks, first made by Joaquin Fuster (1973), is that some neurons switch into a cue-dependent elevated firing rate during the delay, and reset back to their baseline firing upon the choice (Fig. 9.12B; Funahashi et al. 1989; Miller et al. 1996). This stimulus-specific enhancement of firing rates during the delay has been interpreted as an active short-term memory representation of the cue, and variants of model (9.28) have been advanced as a simple neuro-dynamical explanation for this experimental finding. More generally, it may be worth pointing out that the response of a nonlinear system like (9.28) to an external stimulus will strongly depend on the system's current state, including, e.g., the oscillatory phase it's currently in (see below). This deterministic dependence on the current state or phase may account for some of the apparently random trial-to-trial fluctuations in neural responses that have long been debated in neuroscience (Shadlen & Newsome 1998; Kass et al. 2005).

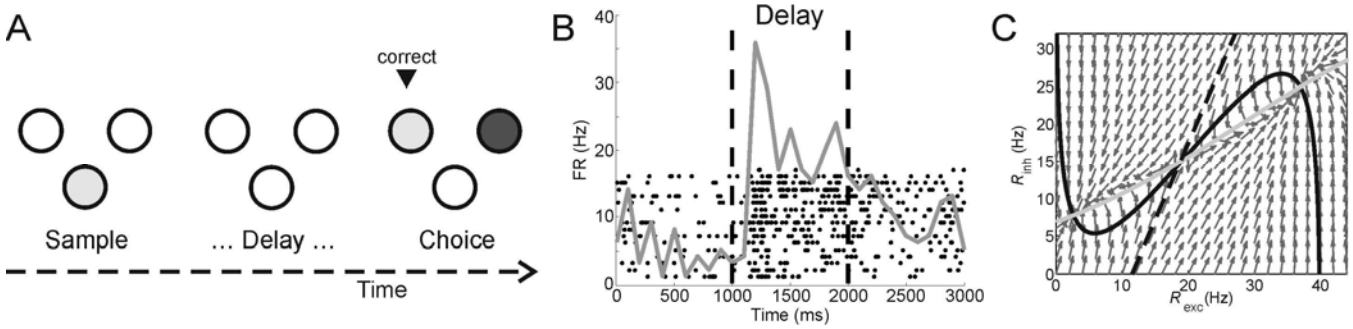

Fig. 9.12. A) Setup of a 'delayed-matching-to-sample' working memory task as used

experimentally: A sample stimulus is present (randomly chosen on each trial), followed by a delay period of one to several seconds (with all external cues removed), followed by a choice period. B) Spike histogram (gray curve) and single trial responses (black dot spike rasters; each line corresponds to a separate trial) for a prefrontal cortex neuron (recordings kindly provided by Dr. Gregor Rainer, Visual Cognition Laboratory, University of Fribourg). Delay phase of 1 s demarked by dashed black lines. C) State space representation of system (9.28) illustrating basins of attraction of the low and high rate attractor states. Reprinted from Durstewitz et al. (2009), Copyright (2009) with permission from Elsevier.

The behavior and phase plane of system (9.28) changes quite dramatically if we change the time constant of the inhibitory feedback. In all the analyses above, the inhibitory feedback was taken to be much faster than the excitation, pressing the flow towards the f_I nullcline. So let us consider what happens if, on the contrary, the inhibition is much slower than the excitation (for this we set τ_E=30 < τ_I=180; also β_I was changed to 4.5, and we start with w_{EE}=1.2, w_{IE}=0.5). This time we will examine how the system's dynamic changes as we slowly increase w_{EI} as a control parameter, i.e., the strength of the excitatory inputs to the inhibitory population. If w_{EI} is small, the inhibitory neurons are not driven strongly enough by the excitatory ones, and hence feedback inhibition is relatively low. In this case, the system settles into a stable fixed point associated with high firing rates (Fig. 9.13, top). But note that this time the flow does not straightly converge from all sides into the fixed point, but approaches it in a kind of damped oscillation (Fig. 9.13, top-center) - hence, this time we are dealing with a *stable spiral point*, not a node! This is also evident from the fact that the fixed point comes with nonzero imaginary parts (Fig. 9.13, top-right). As w_{EI} is increased, the damped oscillatory activity becomes more pronounced (Fig. 9.13, 2nd row), while the real parts of the fixed point's eigenvalues shrink (Fig. 9.13, right). At some point, the fixed point looses stability altogether and activity spirals outward (diverges) from it (Fig. 9.13, 3rd row). Since at this point the system has only one *unstable* fixed point, yet the variables are bounded ($\nu_E \in [0,1]$, $\nu_I \in [0,1]$, scaled by a factor of 40 in Fig. 9.13), according to the *Poincare-Bendixson theorem* it must have a stable limit cycle (Strogatz 1994, Wilson 1999). More generally, according to this theorem, a stable limit cycle in the phase plane can be proven by constructing a 'trapping region' such that from anywhere outside that region trajectories converge into it, while at the same time the region does not contain any fixed points, so that the flow has no other way to go (Strogatz 1994, Wilson 1999).

It might be worth noting that we can obtain these dramatic changes in the system dynamics (from fixed point bistability to stable oscillations) solely (or mainly in this specific case) by altering its *time constants*, i.e., without necessarily any change in the nullclines. Thus, the time scales on which a dynamical system's variables develop relative to each other are an important factor in determining its dynamic, even if the functional form of the nullclines, or other parameters, do not change at all.

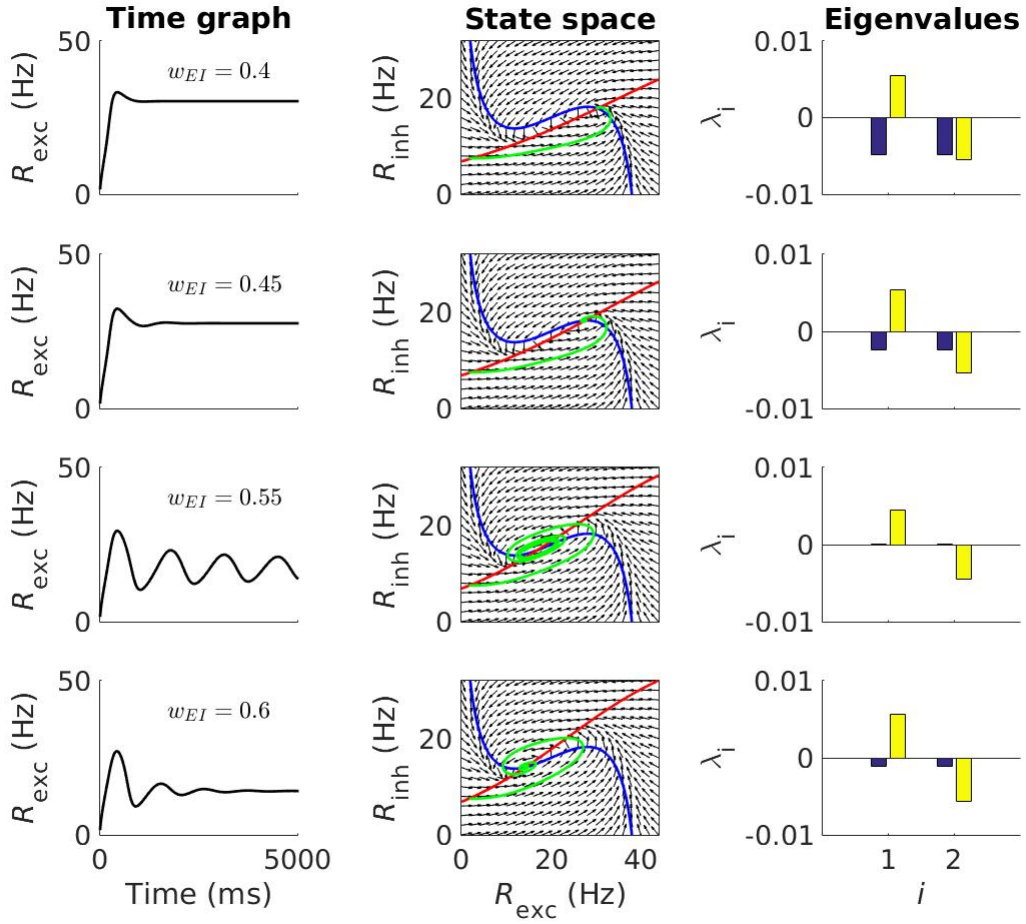

Fig. 9.13. Time graphs (left column), phase spaces (center column; ν_E-nullclines in blue, ν_I-nullclines in red, system trajectory in green), and eigenvalues (right column; real parts in blue, imaginary parts in yellow) for system (9.28) for w_{EI} = 0.4, 0.45, 0.55, and 0.6 (from top to bottom). To better visualize the swirling flow around the spiral point, all flow vectors were normalized to unity length. See Fig. 9.11 for more explanation. See **MATL9_10** for other parameter settings. R_{exc}=40×ν_E, R_{inh}=40×ν_I.

Fig. 9.14 (right) gives the bifurcation graph for the parameter configurations investigated above. At w_{EI} ≈0.497 the stable spiral point looses stability and gives rise to a stable limit cycle slowly growing in amplitude as w_{EI} is further increased. A limit cycle is an isolated closed orbit in state space, unlike the closed orbits encountered in linear systems which are neutrally stable and densely fill the space. This type of bifurcation is called a *supercritical Hopf bifurcation* (Strogatz 1994; Wilson 1999): Its characteristics are that we have a spiral point which changes from stable to unstable, i.e. at the bifurcation point itself we have a pair of purely imaginary eigenvalues, from which a stable limit cycle emerges with infinitesimally small amplitude but relatively constant frequency (Fig. 9.14, right). In contrast, in a *subcritical Hopf bifurcation* an unstable limit cycle develops around a stable spiral point. Passing the point where stable fixed point and limit cycle annihilate each other, this could have dramatic consequences for the system dynamics, as it suddenly has to move somewhere else. In the neural context, the system would usually pop onto a stable limit cycle with finite and relatively constant amplitude but steadily growing frequency as a control parameter is changed. Thus, systems with a subcritical Hopf bifurcation usually exhibit another form of bistability, with a stable spiral point and a stable limit cycle separated by an unstable limit cycle which coalesces and annihilates with the

stable spiral at the Hopf bifurcation point.

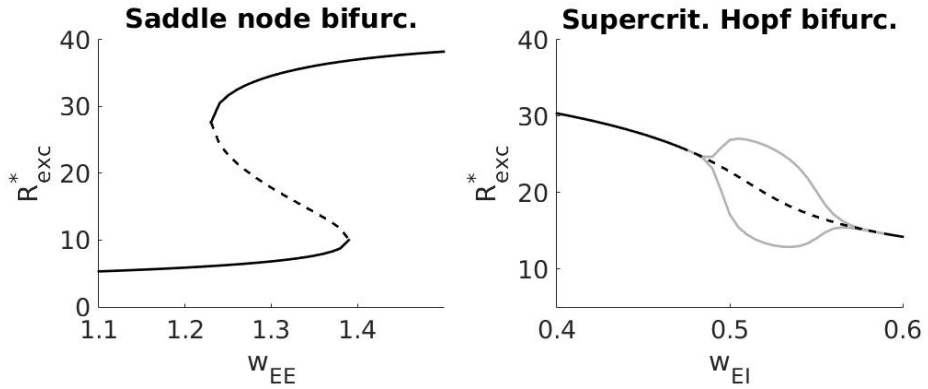

Fig. 9.14. Bifurcation diagrams for system (9.28) for the parameter settings illustrated in Fig. 9.11 (left) and Fig. 9.13 (right). Black solid line = stable fixed points, black dashed line = unstable fixed points, gray = minimum and maximum values of limit cycle. **MATL9_11**.

Experimental evidence for both sub- and supercritical Hopf bifurcations has been obtained in the spiking behavior of various cortical and subcortical neurons. Continuous spiking represents a limit cycle in dynamical terms, and real neurons may undergo various types of bifurcation from stable subthreshold behavior to spiking as, for instance, the amplitude of an injected current into the neuron is increased (Izhikevich 2007). For instance, in many neurons (e.g. Holden & Ramadan 1981) the spike amplitude steadily decreases as an injected current is increased until the membrane potential finally converges to a stable depolarized state ('depolarization block'), characteristics of a supercritical Hopf bifurcation (see also Fig. 9.16C). Many neurons, especially in the auditory system, also exhibit damped subthreshold oscillations within some range, and then eventually suddenly hop into spiking behavior with relatively fixed amplitude but steadily growing frequency as the injected current passes a critical threshold (as in a subthreshold Hopf bifurcation; Hutcheon & Yarom 2000; Stiefel et al. 2013). There are a number of other interesting bifurcations that can give rise to limit cycles and appear to be very common in neural systems. The interested reader is referred to Izhikevich (2007) and Wilson (1999) for further examples and a systematic treatment of dynamical systems in the context of neuroscience.

A final point here concerns the question of how to detect and prove the stability of limit cycles. For fixed points this was relatively straightforward: Set all derivatives to zero, solve for the system's variables, and examine the eigenvalues of the Jacobian matrix at the fixed points. For limit cycles there is no such straightforward approach. We have already mentioned the Poincare-Bendixson theorem, but its applicability is restricted to flows on a (2-dimensional) plane. A simple trick to make *unstable* limit cycles visible (provided the flow is diverging from all directions) is to *invert the time*, i.e. replace $\dot{\mathbf{x}} = \mathrm{f}(\mathbf{x},t)$ by $\dot{\mathbf{x}} = -\mathrm{f}(\mathbf{x},-t)$ (Strogatz 1994). Inverting the flow this way means that unstable objects now become stable and hence visible when the system is started from somewhere within their basin of attraction. Another way to assess the stability of a limit cycle is to convert the continuous flow into a map $\mathbf{x}_t = \mathrm{F}(\mathbf{x}_{t-1})$ by recording only specific points $\mathbf{x}_t$ on the trajectory – for instance, these might be the intersections with a plane strategically placed into the state space, a so-called Poincaré-section, or consecutive local maxima of the time series (Strogatz 1994). In some cases it may be possible to obtain F analytically, or one may approximate it empirically by fitting a model to the pairs (x_{t+1}, x_t) (Fig. 9.15; Strogatz 1994). A fixed point of this map, for which stability may be much easier to prove (see sect. 9.1.1),

would correspond to a limit cycle (or fixed point) within the continuous-time ODE system; hence stability of the fixed point would imply stability of the limit cycle. There are other ways to prove the existence and stability of limit cycles beyond the scope of this brief introduction (Strogatz 1994; Izhikevich 2007).

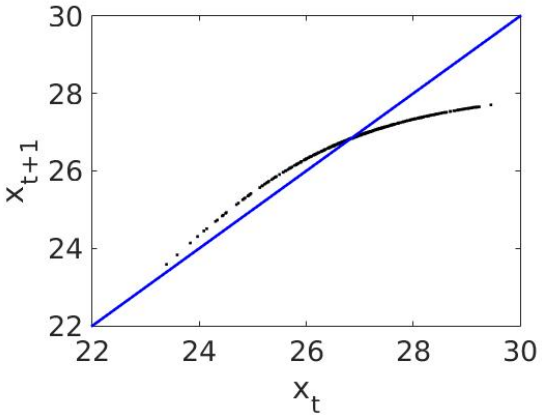

Fig. 9.15. Numerically obtained first-return plot on Poincare section (defined as consecutive maxima x_t of variable v_E) for the stable limit cycle of system (9.28) shown in the third row of Fig. 9.13. The graph indeed indicates stability of the limit cycle with a slope < 1 in absolute value at the fixed point. Note that the system may have to be run from different initial conditions or perturbed multiple times to fill in the map. **MATL9_12**.

Just like nonlinear maps (sect. 9.1), nonlinear ODE systems can also exhibit chaotic phenomena. However, while for a map one (nonlinear!) equation is sufficient to produce chaos, in smooth, continuous ODE systems at least 3 dimensions are needed. In a univariate ODE system, one can have fixed points but no limit cycles (and consequently no chaos either): On an infinite line, the flow cannot cycle as this would violate the condition that the direction of flow has to be uniquely determined at any point. While 2-dimensional state spaces allow for isolated closed orbits, a chaotic attractor requires one dimension more to unfold since the trajectory will have to infinitely cycle within a *bounded* spatial region without ever crossing (repeating) itself – geometrically speaking, this is just not possible in only 2 dimensions (see Strogatz, 1994, and illustrations therein). It cannot grow forever nor shrink toward a single point, since otherwise it would not be a chaotic attractor (but either unstable or a stable fixed point).

For illustration, we will again borrow a 3-ODE example from neuroscience (Durstewitz 2009; based on models in Izhikevich, 2007), representing a simple spiking neuron model with one ODE for the neuron's membrane potential (V), one implementing fast K^+-current driven adaptation (n), and one for slow adaptation (h):

$$
\begin{aligned}
-C_m\dot{V} &= I_L + I_{Na} + I_K + g_M h(V-E_K) + g_{NMDA}\sigma(V)(V-E_{NMDA}) \\
I_L &= g_L(V-E_L) \\
I_{Na} &= g_{Na} m_\infty(V)(V-E_{Na}), \quad m_\infty(V) = \left[1+\exp\left((V_{hNa}-V)/k_{Na}\right)\right]^{-1} \\
I_K &= g_K n(V-E_K) \\
\sigma(V) &= [1+0.33\exp(-0.0625V)]^{-1}
\end{aligned}
\tag{9.32}
$$

$$
\dot{n} = \frac{n_\infty(V)-n}{\tau_n}, \quad n_\infty(V) = \left[1+\exp\left((V_{hK}-V)/k_K\right)\right]^{-1}
$$

$$
\dot{h} = \frac{h_\infty(V)-h}{\tau_h}, \quad h_\infty(V) = \left[1+\exp\left((V_{hM}-V)/k_M\right)\right]^{-1}
$$

This model generates spikes due to the very fast I_{Na} nonlinearity, driving voltage upwards once a certain threshold is crossed, and the somewhat delayed (with time constant τ_n) negative feedback through I_K (Fig. 9.16A). The interplay between these two currents produces a stable limit cycle for some of the system's parameter settings which corresponds to the spiking behavior. Let us first examine the behavior of the system as a function of the much slower feedback variable h (fixing g_{NMDA}), that is we will treat slow variable h for now like a parameter of the system, a methodological approach called *separation of time scales* (Strogatz 1994; Rinzel & Ermentrout 1998). A bifurcation graph of the system's stable and unstable objects as h is varied is given in Fig. 9.16C. Reading the graph from right to left, as h and thus the amount of inhibitory current I_h decreases, the spike-generating limit cycle comes into existence through a *homoclinic orbit bifurcation* (Izhikevich 2007). A homoclinic orbit is an orbit that originates from and terminates within the same fixed point, in this case the saddle node in Fig. 9.16C (a *heteroclinic orbit*, in contrast, is one which would connect *different* fixed points). The limit cycle vanishes again in a supercritical Hopf bifurcation for $h \approx 0.043$. For $h \in [0.043, 0.064]$, the system exhibits bistability between a stable fixed point and the spiking limit cycle.

Now, as h is truly a variable and not a parameter in the fully coupled system eq. 9.32, it will wax and wane during a spiking process and thus move the (V,n)-system back and forth along the h-axis in Fig. 9.16C. In fact, if the amplitude of these variations in h is sufficiently large (as determined by parameter g_M for instance), it may drive the (V,n)-system back and forth across the whole *hysteresis-region* in Fig. 9.16C defined by $h \in [0.043, 0.064]$. In consequence, the system will cycle through phases of repetitive spiking activity once the fixed point lost stability at low h, interrupted by 'silent phases' as h sufficiently increases during the repetitive spiking to drive the (V,n)-system back into the regime where the limit cycle vanishes (in the homoclinic orbit bifurcation), leaving only the stable fixed point. Thus, this is one of the dynamical mechanisms which may give rise to *bursting activity* in neurons (Fig. 9.16A, top; see Rinzel & Ermentrout, 1998, for in depth treatment).

While the slow adaptation variable h may cause bursting by driving model (9.32) back and forth across the hysteresis region, the bistability (hysteresis) regime itself owes its existence in large part to the nonlinearity of the NMDA current. It would cease to exist if NMDA currents were *linear* in the model (Durstewitz & Gabriel 2007; Durstewitz 2009). Fig. 9.16B shows a different type of bifurcation graph where all inter-spike intervals of (9.32) were plotted as a function of the NMDA conductance strength, g_{NMDA}. For low g_{NMDA}, model eq. 9.32 exhibits bursting, indicated by at least two different inter-spike intervals (short ones during the burst and long ones in between). For very high g_{NMDA}, the strong excitatory drive provided by this current places the system into a regime of regular repetitive spiking at high rate (Fig. 9.16A, bottom), marked by a single inter-spike interval in the bifurcation graph Fig. 9.16B. However, somewhere between the oscillatory bursting and repetitive spiking regimes, highly irregular spiking behavior appears, mixing repetitive and bursting phases of different durations (Fig. 9.16A, center). Thus chaos reigns at the transition from clearly bursting to clearly regular single-spiking activity, a phenomenon quite common in systems like the present one (Terman 1992). The interested reader is referred to the brilliant textbook by Steven Strogatz (1994) for an in depth discussion of the probably most famous of all chaotic systems, the Lorenz attractor, and the different routes to chaos (see also Ott 2002).

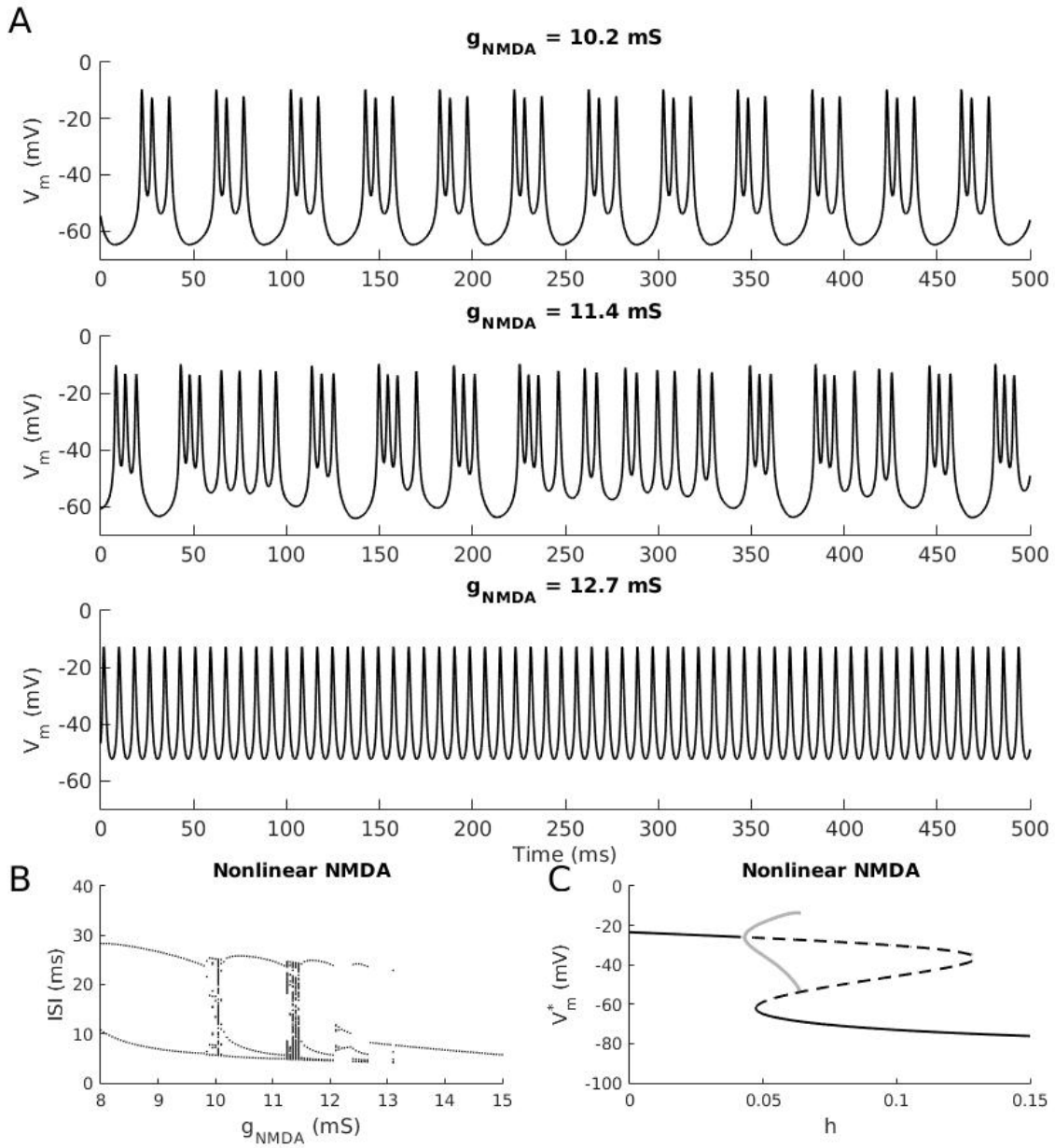

Fig. 9.16. Time series and bifurcation graphs of system (9.32) for different values of parameter g_{NMDA}. B: Note the second- or higher-order limit cycles interrupted by chaotic regimes as g_{NMDA} is increased. C: The stable (spiking) limit cycle (gray curve) in this model arises from a homoclinic orbit bifurcation where the limit cycle terminates on the unstable branch (dashed curve) of the center fixed point. Reprinted from Durstewitz et al. (2009), Copyright (2009) with permission from Elsevier. **MATL9_13**.

Model (9.32), like the 2-ODE firing rate model eq. 9.28 introduced further up, bears a direct relationship to experimental observations: Stimulating pyramidal neurons in a prefrontal cortex brain slice preparation with NMDA seems indeed to induce all of the three different dynamical regimes discussed above. In the presence of NMDA, these neurons can exhibit bursting, repetitive spiking, and chaotic mixtures of repetitive spiking and bursting with many of the same signatures (e.g. in the membrane potential distribution) as observed for their model counterparts (Fig. 9.17). In the absence of NMDA, in contrast, these cells only fire regular, repetitive spikes with just one dominant interspike-interval (ISI), with ISI length depending on the amount of injected current (Durstewitz & Gabriel

2007).

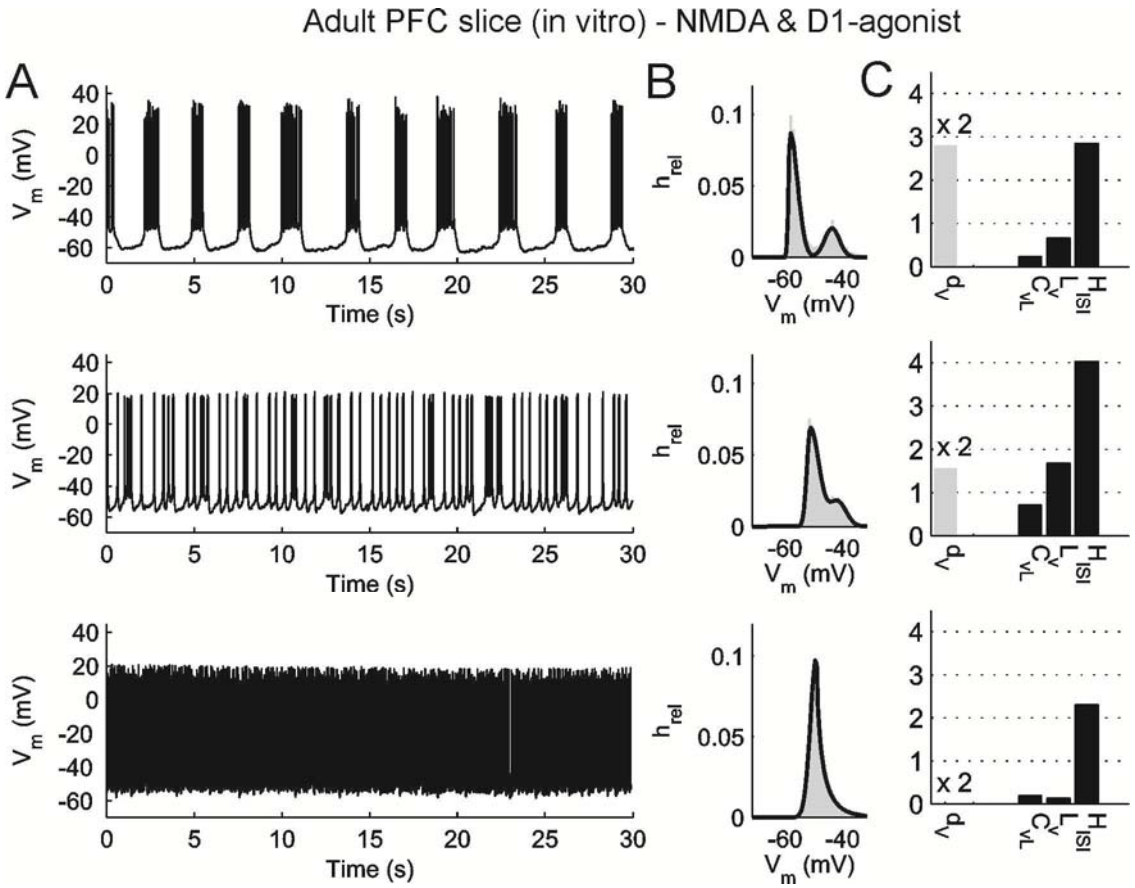

Fig. 9.17. In-vitro spike train recordings from prefrontal cortex neurons driven by an NMDA agonist that exhibit similar dynamical regimes as model (9.32). Panel B shows the membrane potential distributions corresponding to each of the voltage recordings in A. Panel C gives an index of bimodality in the V_m distributions (d_V) and various measures of irregularity in the interspike-interval series (C_{vL}: C_v computed locally; L_V: see Shinomoto et al. 2003; H_{ISI}: entropy of ISI distribution). Reproduced from Durstewitz & Gabriel (2007) by permission of Oxford University Press, with slight modifications.

A quantitative measure for the kind of dynamical regime we may be dealing with that could (in principle – in practice noise is a big problem) also be applied to experimental observations is the *(maximal) Lyapunov exponent* (a *d*-dimensional dynamical system will have *d* Lyapunov exponents, but often only the maximal is of interest). It measures the degree (speed) of convergence or divergence of trajectories that started close to each other in state space. Specifically, assuming an initial distance d_0 between trajectories, this will usually grow or decay exponentially as

$$d(\Delta t) \approx d_0 e^{\lambda \Delta t} . \tag{9.33}$$

The (maximal) *Lyapunov exponent* is λ in the limit of this expression for $\Delta t \to \infty$ and $d_0 \to 0$ (Strogatz 1994; Kantz & Schreiber 2004):

$$\lambda := \lim_{\substack{\Delta t \to \infty \\ d_0 \to 0}} \log\left(\frac{d(\Delta t)}{d_0}\right)\frac{1}{\Delta t} . \tag{9.34}$$

Theoretically, it may sometimes be possible to compute this exponent explicitly by

integrating temporal derivatives along trajectories (e.g. Wilson 1999). Empirically (for a time series sampled at discrete times *t*), one moves along the trajectory, forming a spatial neighborhood $H_\varepsilon(\mathbf{x}_t) = \{\mathbf{x}_\tau \mid d(\mathbf{x}_t, \mathbf{x}_\tau) \le \varepsilon\}$ for each point $\mathbf{x}_t$ and taking the average (Kantz & Schreiber 2004)

$$(9.35)\quad \hat{d}(n\Delta t) = \frac{1}{|H_\varepsilon(\mathbf{x}_t)|} \sum_{\mathbf{x}_\tau \in H_\varepsilon(\mathbf{x}_t)} \left\| \mathbf{x}_{t+n\Delta t} - \mathbf{x}_{\tau+n\Delta t} \right\|$$

across this neighborhood. Averaging $\log[\hat{d}(n\Delta t)]$ along the time series, and plotting this quantity against $n\Delta t$, may reveal a linear regime whose slope can be taken as an estimate of the maximum Lyapunov exponent (usually, the graph will initially exhibit a steep rise with *n* due to noise, and will plateau at some point when the full spatial extent of the attractor [the maximal data range] is reached; see the monograph by Kantz & Schreiber (2004) for more details). If this maximal Lyapunov exponent $\lambda_{max}<0$, we have exponential convergence and thus a system governed by fixed point attractor dynamics. If $\lambda_{max}\approx 0$, this means we may be dealing with a limit cycle, as along the direction of the limit cycle the system is neutrally stable, i.e. a perturbation along this direction will neither decay nor grow but stay (this is actually what facilitates synchrony among phase oscillators, see next section). If $\lambda_{max}>0$, we have exponential divergence of trajectories along at least one direction, thus chaos if the system dynamic is still confined within a bounded region of state space. In a highly noisy system, $\lambda_{max} \to \infty$, as noise will quickly push trajectories apart, at least initially (that is for $n\Delta t$ not too large).

9.2.2 Nonlinear oscillations and phase-locking

Oscillations are ubiquitous in nervous systems and have been claimed to be of prime importance for neural coding and information processing (cf. sect. 7.1), so we will discuss them separately here. Nonlinear oscillations correspond to limit cycles, and hence both linear and nonlinear oscillations may be thought of as movements on the circle, with the precise angular position $\varphi(t)$ on the circle defining the current phase of the oscillator (Strogatz 1994). Let us introduce the topic gently, along similar expositions in Strogatz (1994) and Wilson (1999), with two *uncoupled* linear oscillators moving with constant angular velocities ω_1 and ω_2 around the circle:

$$(9.36)\quad \begin{aligned} \dot{\varphi}_1 &= \omega_1 \\ \dot{\varphi}_2 &= \omega_2 \end{aligned}$$

If the ratio between the two angular velocities $\omega_1/\omega_2 = p/q$, $p, q \in \mathbf{N}_+$, is a rational number, this implies that oscillator φ_1 catches up with φ_2 after exactly *p* turns while the other oscillator did *q* cycles. Geometrically, one may think of this joint phase dynamic as movement on a 'donut' (torus) where φ_2 evolves perpendicularly to φ_1. If p/q is an *irrational* number, this means that the phase difference $\phi = \varphi_1 - \varphi_2$ will constantly drift and the two oscillators are not aligned: In this case, the trajectory (φ_1, φ_2) will densely fill the whole torus, never precisely repeating itself. This phenomenon is called *quasi-periodicity*, and although the joint (φ_1, φ_2)-dynamics is not strictly regular, it is distinct from chaos (for instance, the *component processes* are still strictly periodic in this example).

Let us now introduce coupling between the two oscillators (leading into the wide theoretical field of dynamics in *coupled oscillators*; Pikovsky et al. 2001) by adding a coupling term to eq. 9.36 with amplitude *a*:

(9.37) $$\begin{aligned}\dot{\varphi}_1 &= \omega_1 + a\sin(\varphi_1 - \varphi_2) \\ \dot{\varphi}_2 &= \omega_2 + a\sin(\varphi_2 - \varphi_1)\end{aligned},$$

that is the strength and direction of coupling is taken to be a function of the phase difference $\phi = \varphi_1 - \varphi_2$ between the oscillators (Wilson 1999; Strogatz 1994). A model of this kind was used by Cohen et al. (1982) to explain the coordination among segments of the lamprey spinal cord (assumed to be phase oscillators) necessary to produce coherent movement. Although the particular type of functional relationship may differ, some kind of phase-dependency will generally apply to many real-world oscillators: For instance, neurons interact with others only near the time of spike emission, and the impact a spike has on the postsynaptic target will strongly depend on its current phase, for instance whether it is just in its refractory period (e.g., Rinzel & Ermentrout 1998). Let us examine how the *phase difference* ϕ between the oscillators evolves in time by writing down a differential equation for it (Cohen et al. 1982; Strogatz 1994; Wilson 1999):

(9.38) $$\dot{\phi} = \dot{\varphi}_1 - \dot{\varphi}_2 = \omega_1 + a\sin(\varphi_1 - \varphi_2) - \omega_2 - a\sin(\varphi_2 - \varphi_1) = \omega_1 - \omega_2 + 2a\sin(\phi).$$

Fig. 9.18 (top) shows the $(\phi, \dot{\phi})$-space of this system for the situation a=2 and ω_1=ω_2, i.e. the two oscillators having the same intrinsic frequency. The system has two fixed points at ϕ=0 (closing up with ϕ=2π), and at ϕ=π, graphically given by the intersections of curve $\dot{\phi} = f(\phi)$ with the abscissa ($\dot{\phi} = 0$). The center fixed point at ϕ=π is obviously stable as $f(\phi) > 0$ to the left from it and $f(\phi) < 0$ right from it, while the other one is unstable. Hence, any initial phase difference between the two oscillators will converge to π as time goes by – the two oscillators are said to be *phase-locked* with a phase lag of ϕ=π (for a=-2, the two oscillators would be exactly *synchronous*, or "in phase", and any initial phase difference would shrink back to 0). The important take-home here is that *phase-locking corresponds to an attractor state* of the phase difference dynamic. More generally, this does not have to be a fixed point attractor as for the simple 1-dimensional system eq. 9.38, but could as well be a cycle with the phase difference periodically changing but still bounded. In general, *p*:*q* phase-locking is thus defined (Pikovsky et al. 2001) by

(9.39) $$\left| p\varphi_1 - q\varphi_2 \right| < \varepsilon .$$

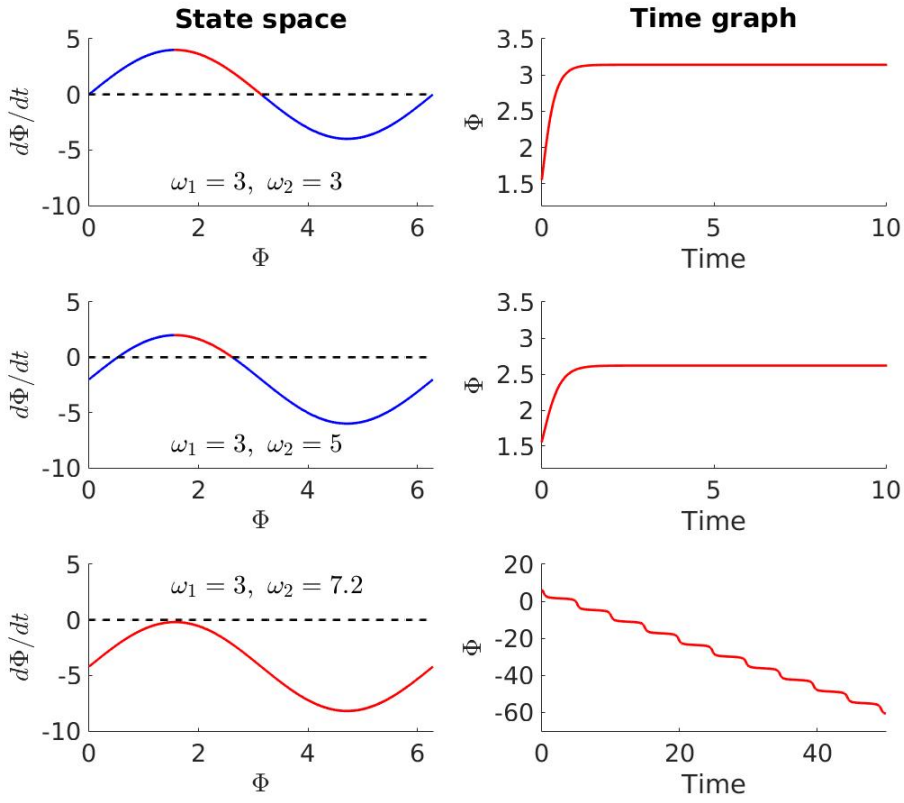

Fig. 9.18. Phase plots (left) and time graphs (right) for the coupled phase oscillators (9.38) for different levels of frequency detuning as indicated. Red portions of curve on left indicate trajectory corresponding to time graphs on the right. Bottom graph: Note the slowly shifting phase difference while the trajectory passes the 'attractor ghost' (or 'ruin') interrupted by fast 'phase slips' (for the time graph, phase was not reset to 0 after each revolution, to better illustrate the constant drift). **MATL9_14**.

As we start to *detune* the two oscillators eq. 9.38 by increasing the difference between their intrinsic frequencies, ω_1-ω_2, the curve $f(\phi) = d\phi/dt$ will move up (or down) in Fig. 9.18. There might still be a stable fixed point, but it will shift along the abscissa, such that phase-locking will occur with a phase $\phi \notin \{0, \pi\}$, with one oscillator leading the other. If the amount of detuning becomes too large (Fig. 9.18, bottom), the stable fixed point disappears in a saddle node bifurcation and there will be constant phase drift; the two oscillators become unlocked. Even with a large difference in intrinsic frequencies, phase locking may be reestablished by increasing the amplitude *a* of the interaction in eqn. 9.37-9.38. More generally, regions of stable phase-locking in the (ω_1-ω_2,*a*) parameter plane form triangle-shaped areas called 'Arnold tongues' (Pikovsky et al. 2001). These will be relatively wide for low-order (like 1:1) locking and decrease in width for higher-order locking (e.g. 5:3). Eventually, as the coupling becomes very strong, this may lead to *complete synchronization* with one variable essentially mimicking the other (Pikovsky et al. 2001).

Another interesting scenario occurs very close to the saddle node bifurcation when the fixed point just lost stability (Fig. 9.18, bottom): The impact of this saddle node '*ghost*' (or '*ruin*') can still be felt by the system as the trajectory will be slowed down as it passes through the narrow channel between the maximum of $f(\phi)$ and the abscissa. This leads to two different time scales (largely independent of the system's intrinsic time constants), with nearly constant phase difference for longer times interrupted by brief *phase slips* (Fig. 9.18, bottom-right).

Phase-locking of oscillators within or between brain areas has been proposed to represent a central means for coding and establishing communication channels (Singer & Gray 1995; Buzsaki 2011). In fact, this is such a huge area of research in both experimental and theoretical neuroscience that this little paragraph can only give a very brief glimpse. For instance, as already reported in sect. 7.1, prefrontal cortex and hippocampus tend to phase-lock their oscillatory activities during choice periods of working memory tasks (Jones & Wilson 2005). Synchronization among brain areas has been suggested to underlie information transfer and coordination during visual processing and selective attention (Engel et al. 2001; Fries et al. 2001). Synchronization among neural spike trains may also bind different sensory features into a coherent percept (Singer & Gray 1995). Fig. 9.19A illustrates the basic idea of phase coding (Hopfield 1995; Brody & Hopfield 2003): Different specific spike time patterns with respect to the underlying population oscillation embody the representation of different sensory objects. Such a representational format has various computational advantages. First, as illustrated in Fig. 9.19A, the very same set of neurons can be utilized to represent a large variety of different objects, thus getting around the frequently discussed 'grandmother-cell issue' (the idea that each specific sensory object in the world is represented by its own specialized [set of] neuron[s]; Singer & Gray 2005). Second, this type of code is fast and temporally very compact – in principle, only a single spike per neuron within a short time window is needed to convey the nature of the sensory object. Third, various computational properties basically come for free with this coding scheme, for instance scale-invariance (Hopfield 1995; Hopfield & Brody 2000; Brody & Hopfield 2003): Assuming a certain form for the neural transfer function, varying the size or intensity of the object may simply push the whole spiking pattern back or forward in phase (Fig. 9.19B), without altering its relational composition (Hopfield 1995). The scale-invariant read-out could be accomplished by a 'coincidence detector' neuron (a "grandmother cell") that collects synaptic signals from the coding neurons with the right set of temporal lags (Fig. 9.19C; Hopfield 1995; Gerstner et al. 1996).

The increased tendency of (neural) oscillators to synchronize as the frequency detuning among them is diminished has been exploited in a number of computationally efficient, elegant, scale-invariant neural pattern recognition devices (Hopfield & Brody 2001; Brody & Hopfield 2003). The core feature of these is that the spiking activity of the neurons is naturally desynchronized due to the frequency detuning caused by the differences in background currents into these neurons. A sensory object that is to be detected elicits a complementary pattern of synaptic inputs that removes the detuning among a subset of receiving neurons (a 'key-fits-lock' principle), which therefore synchronize and signal the detection of the object (Brody & Hopfield 2003). Varying intensities of the stimulus would just uniformly scale up or down the pattern of synaptic inputs, thus not affecting the match to the receiving set per se. Since the synchronization among the neurons goes in hand with increased local field potential oscillations, this mechanism could provide an explanation for the experimentally observed oscillatory activity triggered by biologically relevant but not irrelevant odors in the honeybee olfactory mushroom body (Stopfer et al. 1997).

Putative evidence of phase coding was obtained in various brain areas. In hippocampus, for instance, place cells may indicate a rat's current position on a track by emitting spikes during a particular phase of the hippocampal theta rhythm (Buzsaki & Draguhn 2004). Or, in higher visual areas, object-specific phase codes have even been described during delay periods of a working memory task (Lee et al. 2005).

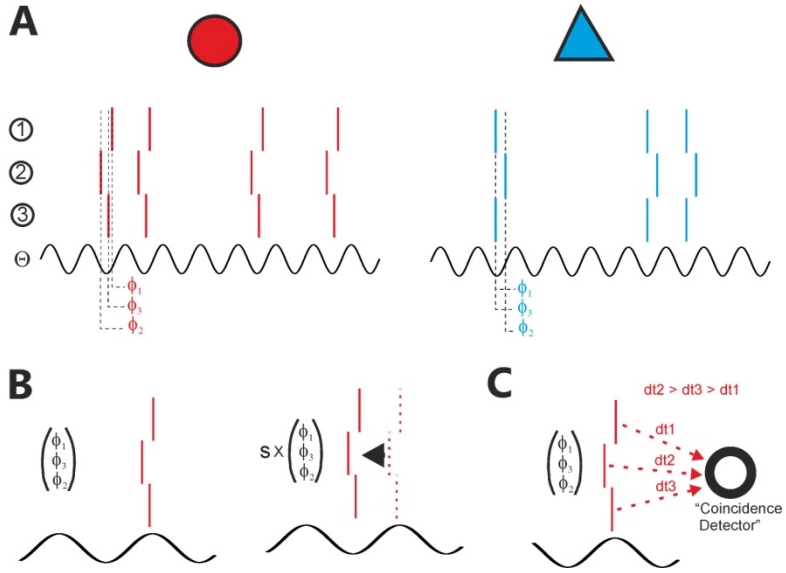

Fig. 9.19. Principle of phase coding. A) Different objects (red square vs. blue triangle) are encoded by a pattern of spike time relations in three units relative to the underlying θ-phase. The pattern may repeat at regular or irregular intervals at the same θ-phase. All the information about the object is encoded in the phase vector (ϕ_1 ϕ_2 ϕ_3). Modified from Durstewitz & Seamans (2006), Copyright (2005) IBRO, with permission from Elsevier. B) Increasing the intensity of the stimulus under certain conditions may lead to a uniform phase shift of the whole spike time pattern, without destroying the spike time relations themselves (Hopfield 1995). C) A coincidence detector may read out the presence of a given object from the simultaneous arrivals of all spikes from the phase coding neurons, which may be achieved if the axonal delays were accordingly adjusted (Gerstner et al. 1996; Brody & Hopfield 2003).

Empirically, phase-locking and -coding can be assessed by graphical representations such as the phase-stroboscope and phase-histogram (Fig. 9.20; Pikovsky et al. 2001), and statistical tests based on these (Hurtado et al. 2004). In fact, numerous approaches have been advanced for detection and statistical testing of phase relations or repeating spike time patterns within or across sets of recorded neurons, some of which we will only briefly summarize here. For instance, unitary event analysis scans simultaneously recorded spike trains for precise spike co-occurrences that exceed the joint spike probability predicted from independent Poisson processes with the same local rate (Grün et al. 2002a, 2002b). Although this could theoretically be extended to any configuration of temporal lags between spiking times, this becomes practically infeasible due to the combinatorial explosion of potential temporal patterns as arbitrary spike time relations are considered. In another approach based on the cumulants of the population spike density of all simultaneously recorded neurons, Staude et al. (2009, 2010) developed a method and stringent statistical test for checking the presence of higher-order (lag-0) correlations among neurons (this approach does not, however, reveal the identity of the underlying 'cellular assemblies'). Another recent ansatz by Shimazaki et al. (2012) builds on the state-space model for Poisson point processes developed by Smith and Brown (2003; see sect. 7.5.3) to extract higher-order spike synchrony from simultaneous spike trains recordings under non-stationary conditions (by allowing parameters to vary according to a latent process). Smith et al. (2010, Smith & Smith 2006) address the problem of testing significance of recurring spike time sequences like those observed in hippocampal place cells (Buzsaki & Draguhn 2004). Their approach only makes use of the order information in the neural activations and hence neglects the exact relative timing of the spikes, or even

the number of spikes emitted by each neuron in the ordered sequence of activations. This allows the derivation of exact probabilities for the events based on combinatorial considerations, while at the same time being able to detect recurrences despite 'time-warping'. In a similar vein, Sastry & Unnikrishnan (2010) employ data mining techniques like 'market-basket analysis' and the a-priori-algorithm (see Hastie et al. 2009) to combat the combinatorial explosion problem in sequence detection. Time series translated into event sequences are scanned first for significant sequential pairs, then for triplets based on this subset of pairs, then quadruplets, and so on, iteratively narrowing down the search space as potential sequences become longer. Their approach also takes the temporal lags among the events in a sequence into account. Finally, rather than working directly on the multivariate point process series defined by the spike recordings, Humphries (2011) applied graph-theoretical approaches to covariance or – more generally – similarity matrices to extract 'spike train communities' (clusters with high within-similarity; see also sect. 6.4).

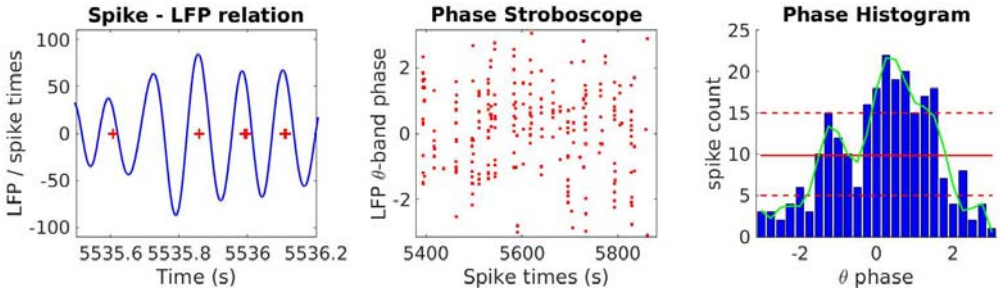

Fig. 9.20. Phase stroboscope and phase histogram from a hippocampal neuron recorded *in vivo* (recordings kindly provided by Dr. Matt Jones, School of Physiology, Pharmacology and Neuroscience, University of Bristol). Left graph shows occurrence of spikes (red crosses) in relation to the local field potential (LFP) filtered in the θ-band (~5-10 Hz). Spike times tend to occur near the peaks of the theta rhythm, avoiding the troughs. The phase stroboscope (center) plots the spike times (red dots) as a function of time (x-axis) and phase of the LFP θ-band (y-axis). Spike times appear to cluster broadly slightly above zero within the temporal interval shown, with possibly a second band below -1, potentially indicating 2:1 locking. Aggregating this information across time gives the phase histogram on the right which confirms preferential spiking near the zero-θ-phase (and possibly a second peak close to -1). Indeed, the spike count leaves the 90% confidence band (dashed red) computed from the binomial distribution for several bins around and right above zero. Green curve illustrates a cubic spline fit to the histogram. **MATL9_15**.

9.3 Statistical inference in nonlinear dynamical systems

Dynamical system models play a pivotal role in (computational) neuroscience as explanatory tools. Biological neural networks are complex nonlinear dynamical systems, and hence nonlinear dynamical models are often required to gain deeper insight into the mechanisms underlying many interesting dynamical phenomena like cellular spiking patterns, oscillatory population activity, cross-correlations and phase-coding, multi-stability, apparent phase transitions, chaotic activity patterns, or behavioral phenomena that evolve over time such as learning. Transitions among dynamical regimes are frequently observed as biophysical parameters of the system (e.g. NMDA conductances) are changed (for instance, by pharmacological or genetic means), and dynamical models can help to understand why and how system dynamics change as biophysical parameters are

modified. They can further provide important insights into the computational significance of such changes, for instance their potential role in pattern formation and completion, memory storage and retrieval, object or speech recognition, motor pattern generation, and so on (Hopfield & Brody 2000, 2001; Brody & Hopfield 2003; Machens et al. 2005; Buonomano 2000; Brunel & Wang 2001; Wang 2002; Fusi et al. 2007; Mongillo et al. 2008; Tsodyks 2005; Gütig & Sompolinsky 2006; Lisman et al. 1998; Sussillo & Abbott 2009; Durstewitz et al. 2000; Durstewitz & Seamans 2002, 2008; Durstewitz & Gabriel 2007). Topics like these define the field of computational neuroscience in its core, and it is way too large an area in its own right to be nearly covered in this book – see e.g. the monographs by Dayan & Abbott (2001), Hertz et al. (1991), Koch (1999), or Izhikevich (2007). The preceding sections hardly scratched the surface in this regard.

Although dynamical system models have been used with huge success in explaining and predicting various dynamical and computational phenomena, this usually remains at a more qualitative and less quantitative level, as in most of the examples in the previous two sections. Dynamical system models of neurons or networks are frequently tuned 'by hand' to get a rough match to their empirical counterparts, for instance in mean firing rate and inter-spike-interval variations. After some initial tuning, they are then directly applied to the empirical observations they are supposed to account for (e.g. Durstewitz et al. 2000; Brunel & Wang 2001; Wang 2002). This has to do with their complexity, the strong nonlinearities these models usually contain and their sometimes highly chaotic behavior, and the many parameters and equations (sometimes on the order of thousands) that might be needed to represent 'biological reality' in sufficient detail for investigating a particular physiological phenomenon; factors that seem to impede more systematic, principled, or even analytical approaches to estimation, prediction, and statistical inference.

However, if neurocomputational models could be utilized more directly as data-analytical tools, embedded within a statistical framework, they may enable researchers to dive much deeper beyond the data surface, and to gain a much more thorough theoretical understanding of the experimental observations. This role neurocomputational models could only fill in if more systematic means for parameter estimation and statistical inference were available. Embedding neuro-computational models into a statistical framework will not only equip them with principled ways of estimating their parameters from neural and behavioral data, rather than performing laborious and subjective trial-and-error search in parameter space. It will also come with strict statistical criteria, based e.g. on likelihood functions or prediction errors, according to which their quality as explanatory tools could be formally judged and compared. A good computational model would be one that can predict, in an out-of-sample sense (Ch. 4), physiological or behavioral test data not used for estimating the model (e.g. Hertäg et al. 2012). Also, explicit statistical testing of different hypotheses regarding the underlying neuro-computational processes and mechanisms would become feasible this way. Various computational hypotheses could be formulated in terms of neuro-computational models which could be directly contrasted on the same data set with respect to their predictive power, their Bayesian posteriors, or, if nested, using formal test statistics like the log-likelihood ratio.

9.3.1 Nonlinear dynamical model estimation in discrete and continuous time

Moving back into the realm of statistics, one may equip structural models of the form eq. 9.10 or eq. 9.16 with probability assumptions, and directly estimate their parameters from the neural and/ or behavioral data at hand (as opposed to just exposing the model to the same task setup as used for the experimental subjects, as in the studies cited in sect. 9.1.2). Such a statistical framework would allow moving dynamical system models in neuroscience away from being mainly exploratory, qualitative tools, toward being truly quantitative, data-analytical tools. Experimentally, commonly only a subset of those

variables specifying the model dynamics is observed, or even only quantities like spike counts that indirectly relate to the underlying system dynamics without forming a dynamical process by themselves. Thus, in this domain we will almost inevitably have to deal with latent variable models which will be collectively referred to as *nonlinear state space models* in the following. The term nonlinear here denotes the type of *transition dynamics* (note that the logistic map (9.3) is in fact *linear* in its parameter α, thus may be interpreted as a kind of basis expansion in *x* from a purely statistical perspective). Within this class of models, exact inference is generally impossible, and one has to retreat to approximate and/or numerical sampling methods.

We will start by discussing inference in discrete-time dynamical systems before going into methods specific for continuous-time systems. Note that we can always discretize a continuous-time dynamical system; in fact that's the way ODE or PDE systems are solved numerically on the computer anyway, by numerical integration schemes that progress in discrete time (and/or space) steps (e.g. Press et al. 2007). And obviously experimental data come sampled at discrete time steps as well. Thus, the discrete-time methods discussed next can be and have been (e.g. Paninski et al. 2012, Lankarany et al. 2013) applied to continuous-time dynamical systems as well.

Assume we have observed an *N*-variate time series of spike counts $\mathbf{c}_t = (c_{1t} ... c_{Nt})^T$ from which we wish to estimate a RNN model like (9.10). If the activation function *G* of the RNN is a sigmoid as in (9.8)-(9.10), one could take the approach of interpreting $\mathbf{x}_t$ directly as a vector of spiking probabilities since these values are confined in [0, 1] (although, more generally, the outputs may always be re-scaled to map onto the required output range). In that case, one would take the bin width small enough such that the empirical counts c_{it} become binary numbers $\in \{0,1\}$. Suppose the transition dynamic itself is *deterministic* (no noise term), as in eq. 9.10, and each output could be assigned to exactly one RNN unit, then the data likelihood can be expressed through the Bernoulli process as

$$(9.40)\quad L_{\mathbf{C}}(\mathbf{W}) = p(\{\mathbf{c}_t\} \mid \mathbf{W}) = \prod_{t=1}^{T} \prod_{i=1}^{N} x_{it}^{c_{it}} (1 - x_{it})^{1-c_{it}} ,$$

where **W** is the weight matrix in model (9.10). (Assuming the RNN process is deterministic, for given **W** and initial state $\mathbf{x}_0$ the RNN trajectory $\{\mathbf{x}_t\}$ is completely determined and we do not need to integrate the r.h.s. of eq. 9.40 across all possible paths as in state space models, sect. 7.5.)

So we have interpreted the x_{it} as continuously-valued spiking probabilities which evolve as the (deterministic) latent process underlying the observed series of binary spike events. This approach might not work well in practice (although I never tried), since the RNN activities $\mathbf{x}_t$ would have to be very low most of the time, producing long series of 0's, with occasional jumps to higher values. Also, assuming that the underlying dynamics is itself deterministic is unrealistic and may lead to identification of the wrong underlying model. So here is an alternative interpretation: We make the network states binary themselves, $x_{it} \in \{0,1\}$, and take the output from the RNN sigmoid eqn. 9.8-9.9 as the probability with which unit *i* will find itself in the 'off' (0) vs. the 'on' (1) state at the next time step, $pr(x_{i,t+1} = 1) = G\left(w_{i0} + \sum_{j=1}^{N} w_{ij} x_{jt}\right)$ with *G* defined as in (9.9). Thus, we obtain a RNN dynamics which is by itself *stochastic*, with units switching probabilistically between just two states, and the switching probability determined by the same sigmoid-type function and network input as used for model (9.8)-(9.10). Note that in this formulation the RNN dynamics $\{\mathbf{x}_t\}$ is not (necessarily) a latent process anymore, but we could plug the observed spiking activities $c_{it} \in \{0,1\}$ right away into the activation function *G* for the x_{it}

(again facilitating estimation). In fact, this gives kind of a logistic (see sect. 3.3) auto-regressive model, similar to the Poisson auto-regressive model discussed in sect. 7.3. This type of stochastic RNN is known in the neural network community as a *Boltzmann machine*, since the joint probability distribution over network states will eventually (in steady state) reach a Boltzmann distribution (Hinton & Sejnowski 1986, Aarts & Korst 1988; closely related to *Ising models* and *Hopfield networks*, see Hertz et al. 1991; strictly, a Boltzmann machine also comes with symmetric connectivity, $w_{ij}=w_{ji}$ for all (i,j), and zero self-connections, $w_{ii}=0$ for all i). Concepts along these lines have frequently been used in neuroscience to infer the underlying connectivity or the significance of unit interactions in neural coding (e.g. Schneidman et al. 2006).

We will now discuss a more general RNN approach, including probability assumptions for the latent process, and allowing for larger bin widths, such that variables c_{it} can assume larger (integer) values and do not form a sparse series anymore. (In fact, choosing the binning large enough may also avoid other issues in the estimation of nonlinear latent variable models by forming 'summary statistics', to be discussed in sect. 9.3.3.) We will also give up the assumption we had made above that there is a 1:1 mapping between RNN units and observed units. For instance, since the neurons observed are usually only a small sample from a much larger network, to fully account for the observed spiking dynamics, we may have to extend the RNN through hidden units beyond those that had been explicitly observed. Doing so may reduce the danger of misattributing observed dynamics directly to interactions among observed units, while they were really caused through interactions with non-observed variables. Or, the other way round, we may want to reduce the observed space to a much lower-dimensional unobserved latent space, as in Gaussian Process Factor Analysis (sect. 7.5.2). This may be desired for purposes of visualization, or if we suspect the true underlying network dynamics to be lower-dimensional than the nominal dimensionality of the observed process. Often it is these essential dynamical features of the underlying dynamical system we may want to extract from the observed recordings.

In general, we will link the observed spike count series $\{\mathbf{c}_t \in \mathbf{Q}^{N\times 1}\}, t=1...T,$ to the latent dynamics through a Poisson model with the conditional mean a function of $\{\mathbf{x}_t \in \mathbf{R}^{M\times 1}\}$, and include a Gaussian white noise term in the latent process. Such a model was introduced by Yu et al. (2005) who included a nonlinearity in the hidden state equations to yield a stochastic RNN similar in form to (9.10). Slightly modified from Yu et al. (2005), it is given by

$$
\begin{aligned}
& c_{it} \mid \mathbf{x}_t \sim Poisson(\lambda_{it}) \;,\; \lambda_{it} = \exp[\log \beta_{0i} + \boldsymbol{\beta}_{1i}\mathbf{x}_t] \\
& \mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{W}\phi(\mathbf{x}_{t-1}) + \mathbf{S}_t + \boldsymbol{\varepsilon}_t \\
& \phi(\mathbf{x}_t) = \left[1 + e^{\gamma(\delta - \mathbf{x}_t)}\right]^{-1} \\
& \boldsymbol{\varepsilon}_t \sim N(\mathbf{0}, \boldsymbol{\Sigma}) \;,\; \mathbf{A} = diag[\alpha_j] \;,\; \boldsymbol{\Sigma} = diag[\varepsilon_j] \\
& \mathbf{x}_1 \sim N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma})
\end{aligned}
\qquad (9.41)
$$

Note that, unlike model (9.10), the transition equation for $\mathbf{x}_t$ takes the form of an AR(1) model with a kind of *basis expansion* in $\mathbf{x}_{t-1}$ given by $\phi(\mathbf{x}_{t-1})$; that is, in contrast to (9.10), model (9.41) is *linear* in parameters $\mathbf{A}$ and $\mathbf{W}$ (but the dynamics is still nonlinear!). This linearity in parameters profoundly simplifies the maximization step in EM (see also Ghahramani & Roweis 1999) without limiting the dynamical repertoire of the model (in fact, note that transition dynamics (9.41) may be rewritten by substituting $\mathbf{x}_t \mapsto \mathbf{W}\mathbf{y}_t$ and multiplying through by $\mathbf{W}^{-1}$, Beer 2006; see also Funahashi & Nakamura 1993; Kimura &

Nakano 1998; Chow & Li 2000, for the dynamical versatility of this class of models). Constants α_j (arranged in diagonal matrix **A**) regulate the temporal decay of unit activities x_{jt} and are thus related to the time constants of the system, while parameter matrix **W** weighs the inputs from the other network units. $\mathbf{S}_t$ represents external inputs (like sensory stimuli) into the system and is assumed to be fixed by the experimental conditions, i.e. not subject to estimation. One may think of $\mathbf{x}_t$ as a set of underlying membrane potentials, which are translated by row vectors $\boldsymbol{\beta}_{1i}$ into observed spike counts c_{it} for each unit *i* in time bin *t* according to a Poisson process with time-dependent rate λ_{it} (cf. eq. 7.81 in sect. 7.5.1). Parameter β_{0i} represents the base line spike rate. Note that Gaussian noise matrix $\boldsymbol{\Sigma}$ is restricted to be diagonal, such that all correlations among units must be due to dynamical interactions through connectivity matrix **W** and latent state mixing through vectors $\boldsymbol{\beta}_{1i}$ (scalar and unique β coefficients may be used, if one wants to attribute all correlations to the network connectivity **W** and avoid potential model identification issues). Yu et al. (2005) used this formalism for reconstructing neural state trajectories from multiple single unit recordings from the primate premotor cortex during a delayed-reaching-task. As noted above, by employing a much lower-dimensional state vector $\mathbf{x}_t$, compared to the number of recorded units, one could also achieve dimensionality reduction at the same time.

Yu et al. (2005) suggest an approximate EM algorithm for model estimation. The mathematical derivations become a bit more involved at this stage, but the complications actually arise more in the M-step. Let us start with the expected log-likelihood $\mathrm{E}_{\mathbf{X}}[\log L_{\mathbf{C},\mathbf{X}}(\boldsymbol{\theta})]$ of model (9.41), which looks very similar to the one for the linear Poisson state space model treated in sect. 7.5.3 (eq. 7.83; not surprisingly, since model eq. 9.41 shares the same distributional assumptions with model eq. 7.85):

(9.42)

$$
\begin{aligned}
&E_{\{\mathbf{x}_t\}}\left[\log p(\{\mathbf{c}_t,\mathbf{x}_t\}\,|\,\boldsymbol{\theta})\right] \\
&\quad= E_{\{\mathbf{x}_t\}}\left[\log\left(\prod_t\prod_i p(c_{it}\,|\,\{\mathbf{x}_t\},\boldsymbol{\theta})\right)+\log\left(p(\mathbf{x}_1\,|\,\boldsymbol{\theta})\prod_{t>1}p(\mathbf{x}_t\,|\,\mathbf{x}_{t-1},\boldsymbol{\theta})\right)\right] \\
&\quad= \sum_{t=1}^{T}\sum_{i=1}^{N}E\left[c_{it}(\log\beta_{0i}+\boldsymbol{\beta}_{1i}\mathbf{x}_t)-\log c_{it}!-\beta_{0i}e^{\boldsymbol{\beta}_{1i}\mathbf{x}_t}\right]+ \\
&\quad\quad\sum_{t=2}^{T}E\left[-\frac{M}{2}\log 2\pi-\frac{1}{2}\log|\boldsymbol{\Sigma}|-\frac{1}{2}(\mathbf{x}_t-\mathbf{A}\mathbf{x}_{t-1}-\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_{t-1})-\mathbf{S}_t)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_t-\mathbf{A}\mathbf{x}_{t-1}-\mathbf{W}\boldsymbol{\phi}(\mathbf{x}_{t-1})-\mathbf{S}_t)\right]+ \\
&\quad\quad E\left[-\frac{M}{2}\log 2\pi-\frac{1}{2}\log|\boldsymbol{\Sigma}|-\frac{1}{2}(\mathbf{x}_1-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_1-\boldsymbol{\mu}_0)\right]
\end{aligned}
$$

Dropping constants, pulling the expectancies inside using relation $\mathbf{x}^T\mathbf{A}\mathbf{y}=tr[\mathbf{A}\mathbf{y}\mathbf{x}^T]$ as in (7.55), and using the moment generating function for the Gaussian, eq. 7.84, this becomes

(9.43)

$$
\begin{aligned}
Q(\boldsymbol{\theta}) := & \sum_{t=1}^{T}\sum_{i=1}^{N}\left(c_{it}(\log\beta_{0i} + \boldsymbol{\beta}_{1i}E[\mathbf{x}_t]) - \beta_{0i}e^{\boldsymbol{\beta}_{1i}E[\mathbf{x}_t]+\frac{1}{2}\boldsymbol{\beta}_{1i}(E[\mathbf{x}_t\mathbf{x}_t^T]-E[\mathbf{x}_t]E[\mathbf{x}_t^T])\boldsymbol{\beta}_{1i}^T} \right) \\
& -\frac{1}{2}\sum_{t=2}^{T}\Big\{ tr\left(\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_t\mathbf{x}_t^T]\right) - tr\left(\boldsymbol{\Sigma}^{-1}\mathbf{A}E[\mathbf{x}_{t-1}\mathbf{x}_t^T]\right) - tr\left(\boldsymbol{\Sigma}^{-1}\mathbf{W}E[\phi(\mathbf{x}_{t-1})\mathbf{x}_t^T]\right) - E[\mathbf{x}_t^T]\boldsymbol{\Sigma}^{-1}\mathbf{S}_t - \\
& \qquad tr\left(\mathbf{A}^T\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_t\mathbf{x}_{t-1}^T]\right) + tr\left(\mathbf{A}^T\boldsymbol{\Sigma}^{-1}\mathbf{A}E[\mathbf{x}_{t-1}\mathbf{x}_{t-1}^T]\right) + tr\left(\mathbf{A}^T\boldsymbol{\Sigma}^{-1}\mathbf{W}E[\phi(\mathbf{x}_{t-1})\mathbf{x}_{t-1}^T]\right) + E[\mathbf{x}_{t-1}^T]\mathbf{A}^T\boldsymbol{\Sigma}^{-1}\mathbf{S}_t - \\
& \qquad tr\left(\mathbf{W}^T\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_t\phi(\mathbf{x}_{t-1}^T)]\right) + tr\left(\mathbf{W}^T\boldsymbol{\Sigma}^{-1}\mathbf{A}E[\mathbf{x}_{t-1}\phi(\mathbf{x}_{t-1}^T)]\right) + tr\left(\mathbf{W}^T\boldsymbol{\Sigma}^{-1}\mathbf{W}E[\phi(\mathbf{x}_{t-1})\phi(\mathbf{x}_{t-1}^T)]\right) + \\
& \qquad E[\phi(\mathbf{x}_{t-1}^T)]\mathbf{W}^T\boldsymbol{\Sigma}^{-1}\mathbf{S}_t - \mathbf{S}_t^T\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_t] + \mathbf{S}_t^T\boldsymbol{\Sigma}^{-1}\mathbf{A}E[\mathbf{x}_{t-1}] + \mathbf{S}_t^T\boldsymbol{\Sigma}^{-1}\mathbf{W}E[\phi(\mathbf{x}_{t-1})] + \mathbf{S}_t^T\boldsymbol{\Sigma}^{-1}\mathbf{S}_t \Big\} \\
& -\frac{T}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}\left\{ tr\left(\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_1\mathbf{x}_1^T]\right) - E[\mathbf{x}_1^T]\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 - \boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}E[\mathbf{x}_1] + \boldsymbol{\mu}_0^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 \right\}
\end{aligned}
$$

This expression looks bewildering as is, but the most troubling aspect about it is that it involves expectancies of nonlinear functions of the $\mathbf{x}_t$, like $E[\phi(\mathbf{x}_{t-1})]$, $E[\mathbf{x}_t\phi(\mathbf{x}_{t-1}^T)]$, $E[\mathbf{x}_{t-1}\phi(\mathbf{x}_{t-1}^T)]$, and $E[\phi(\mathbf{x}_{t-1})\phi(\mathbf{x}_{t-1}^T)]$, in addition to the usual suspects $E[\mathbf{x}_t]$, $E[\mathbf{x}_t\mathbf{x}_t^T]$, and $E[\mathbf{x}_t\mathbf{x}_{t-1}^T]$. If we had $p(\mathbf{x}_t|\{\mathbf{c}_t\},\boldsymbol{\theta})$ and $p(\mathbf{x}_t,\mathbf{x}_{t-1}|\{\mathbf{c}_t\},\boldsymbol{\theta})$, of course, these could in principle be evaluated.

Considering the E-step first, to make use of the Kalman-filter-smoother formalism, Yu et al. (2005) get rid of the transition nonlinearity by performing a first-order Taylor expansion (linearization) of $\phi(\mathbf{x}_{t-1})$ *locally* (in time) around the previous mean estimator $\boldsymbol{\mu}_{t-1}$,

$$
\begin{aligned}
& \phi(\mathbf{x}_{t-1}) \approx \phi(\boldsymbol{\mu}_{t-1}) + \phi'(\boldsymbol{\mu}_{t-1})(\mathbf{x}_{t-1} - \boldsymbol{\mu}_{t-1}) \\
\Rightarrow \quad & p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{x}_{t-1}) \approx (2\pi)^{-M/2}|\boldsymbol{\Sigma}|^{-1/2}\, e^{-\frac{1}{2}(\mathbf{x}_t-\mathbf{K}_{t-1}\mathbf{x}_{t-1}-\mathbf{U}_{t-1})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}_t-\mathbf{K}_{t-1}\mathbf{x}_{t-1}-\mathbf{U}_{t-1})}
\end{aligned}
\tag{9.44}
$$

with

$$
\begin{aligned}
\mathbf{K}_{t-1} &:= \mathbf{A} + \mathbf{W}\phi'(\boldsymbol{\mu}_{t-1}) \\
\mathbf{U}_{t-1} &:= \mathbf{W}\phi(\boldsymbol{\mu}_{t-1}) - \mathbf{W}\phi'(\boldsymbol{\mu}_{t-1})\boldsymbol{\mu}_{t-1} + \mathbf{S}_t
\end{aligned}
$$

and Jacobian matrix $\phi'(\boldsymbol{\mu}_{t-1}) \equiv (\partial\phi(\boldsymbol{\mu}_{t-1})/\partial\boldsymbol{\mu}_{t-1})$. This is also called the *Extended Kalman Filter-Smoother*. Using this linearization in combination with the Gaussian approximation for the numerator of the posterior $p(\mathbf{x}_t|\{\mathbf{c}_{\tau\le t}\},\boldsymbol{\theta})$ in eq. 7.86, sect. 7.5.3, we can perform the Kalman filtering and smoothing operations exactly as outlined in sect. 7.5.3 for the Poisson state space model (see Yu et al. 2005). The only difference is that we replace transition matrix **A**, eq. 7.87, by $\mathbf{K}_{t-1}$, and input $\mathbf{Bs}_t$ in eq. 7.87 by $\mathbf{U}_{t-1}$, as defined above. Thus, we made two approximations in deriving $p(\mathbf{x}_t|\{\mathbf{c}_{\tau\le t}\},\boldsymbol{\theta})$ here: One to account for the nonlinearity in the transition (the Taylor-based linearization), and one to deal with the Poisson observation equation (the Gaussian approximation to the numerator of eq. 7.86). This whole procedure is implemented in **MATL9_16** and yields the means and covariance matrices for $p(\mathbf{x}_t|\{\mathbf{c}_t\},\boldsymbol{\theta})$ and $p(\mathbf{x}_t,\mathbf{x}_{t-1}|\{\mathbf{c}_t\},\boldsymbol{\theta})$. Fig. 9.21 provides an example of the estimated state path from a 'ground truth' model, i.e. an RNN simulation setup with exactly the same parameters as used for state path estimation.

Before we move on to the M-step, let us first state the Extended Kalman Filter equations more generally, as they are one of the most common tools for extending linear state space models (sect. 7.5.1) toward nonlinear/ non-Gaussian state space models. As outlined above, they (approximately) account for nonlinearities or non-Gaussian

distributions in the observation and transition equations by effectively linearizing the system about the current estimates (Fahrmeir & Tutz 2010). Consider the model

$$(9.45)\quad \begin{aligned} E_{\boldsymbol{\theta}}[\mathbf{x}_t \mid \mathbf{z}_t] &= g(\mathbf{z}_t) \\ \mathbf{z}_t &= f(\mathbf{z}_{t-1}) + \boldsymbol{\varepsilon}_t \,,\, \boldsymbol{\varepsilon}_t \sim N(\mathbf{0}, \boldsymbol{\Sigma}) \end{aligned} .$$

This encompasses non-Gaussian situations in the observations which imply nonlinear relationships between $E_{\boldsymbol{\theta}}[\mathbf{x}_t \mid \mathbf{z}_t]$ and the hidden state $\mathbf{z}_t$. The extended Kalman filter updates are then given by (Durbin & Koopman 2012)

$$(9.46)\quad \begin{aligned} \boldsymbol{\mu}_t &= f(\boldsymbol{\mu}_{t-1}) + \mathbf{K}_t[\mathbf{x}_t - g(f(\boldsymbol{\mu}_{t-1}))] \\ \mathbf{V}_t &= \mathbf{L}_{t-1} - \mathbf{K}_t \nabla_{t-1} \mathbf{L}_{t-1} \\ \mathbf{K}_t &= \mathbf{L}_{t-1} \nabla_{t-1}^T (\nabla_{t-1} \mathbf{L}_{t-1} \nabla_{t-1}^T + \boldsymbol{\Gamma})^{-1} \\ \mathbf{L}_{t-1} &= \mathbf{J}_{t-1} \mathbf{V}_{t-1} \mathbf{J}_{t-1}^T + \boldsymbol{\Sigma} \end{aligned} ,$$

where $\nabla_{(i,j),t-1} = (\partial g_i / \partial f(\boldsymbol{\mu}_{j,t-1}))$ is the Jacobian matrix of partial derivatives of the vector function *g* with respect to the one-step-ahead predictors $f(\boldsymbol{\mu}_{j,t-1})$, and $\mathbf{J}_{(i,j),t-1} = (\partial f_i / \partial \boldsymbol{\mu}_{j,t-1})$ the Jacobian of vector function *f*. Note that for the standard Gaussian linear model (7.53), since in this case the functions $f(\mathbf{z}_t) = \mathbf{A}\mathbf{z}_t$ and $g(\mathbf{z}_t) = \mathbf{B}\mathbf{z}_t$ are linear, $\nabla_{t-1} = \mathbf{B}$ and $\mathbf{J}_{t-1} = \mathbf{A}$, such that (9.46) becomes equivalent to the linear update equations (7.63). Extended Kalman filtering has been used by several authors to infer parameters of neural systems, for instance by Lankarany et al. (2013) to estimate properties of excitatory and inhibitory synaptic inputs generating observed membrane potential fluctuations.

Now let's turn to the M-step. We will not give full details of the derivations here, which are quite lengthy, but rather focus on a few key steps required in the solution, and then just state the final results for completeness. Further details can be found in the Matlab implementation **MATL9_16** (see Fig. 9.21 for an example of estimated parameters). Let's take on the expectancies involving the nonlinearity $\phi(\mathbf{x}_t)$ first. To derive these, we have to solve single and double integrals to obtain the vector and matrix elements, respectively, of $E[\phi(\mathbf{x}_{t-1})]$, $E[\mathbf{x}_t \phi(\mathbf{x}_{t-1}^T)]$, $E[\mathbf{x}_{t-1} \phi(\mathbf{x}_{t-1}^T)]$, and $E[\phi(\mathbf{x}_{t-1}) \phi(\mathbf{x}_{t-1}^T)]$. First note that $\phi(\mathbf{x}_t)$ is a *strictly monotonic, invertible* function, such that the density $p(X_{it})$ evaluated at $X_{it} = x_{it}$ returns the same value as the density $f(\Phi_{it})$ evaluated at $\Phi_{it} = \varphi_{it} := \phi(x_{it})$ (here we have used upper-case letters (X, Φ) to denote the respective random variables and lower-case letters to indicate the specific values they take on). This enables us to make use of a standard result for obtaining the distribution of (strictly monotonic) functions of random variables (see Wackerly et al. 2008), namely

$$(9.47)\quad f(\Phi_{it}) = p(\phi^{-1}(\varphi_{it})) \left| \frac{\partial \phi^{-1}(\varphi_{it})}{\partial \varphi_{it}} \right| = p(X_{it}) [\gamma(\varphi_{it} - \varphi_{it}^2)]^{-1} .$$

Taking the expectancies $E[\phi(x_{it})]$ and $E[\phi(x_{it}) \phi(x_{jt})]$ as examples, we thus get for the integrals

$$E[\phi(x_{it})] = \int_{-\infty}^{\infty} N(x_{it} \mid \tilde{\mu}_{it}, \tilde{V}_{it})\phi(x_{it})dx_{it} = \int_{0}^{1} N(\phi^{-1}(\varphi_{it}) \mid \tilde{\mu}_{it}, \tilde{V}_{it})[\gamma(1-\varphi_{it})]^{-1} d\varphi_{it}$$

(9.48)
$$E[\phi(x_{it})\phi(x_{jt})] = \int_{-\infty-\infty}^{\infty\ \infty} N([x_{it}\ x_{jt}]^T \mid \tilde{\boldsymbol{\mu}}_{ij,t}, \tilde{\mathbf{V}}_{ij,t})\phi(x_{it})\phi(x_{jt})dx_{it}dx_{jt}$$
$$= \int_{0\ 0}^{1\ 1} N([\phi^{-1}(\varphi_{it})\ \phi^{-1}(\varphi_{jt})]^T \mid \tilde{\boldsymbol{\mu}}_{ij,t}, \tilde{\mathbf{V}}_{ij,t}) \left| \frac{\partial\phi^{-1}}{\partial\varphi_{it}} \frac{\partial\phi^{-1}}{\partial\varphi_{jt}} \right| \varphi_{it}\varphi_{jt} d\varphi_{it} d\varphi_{jt}$$

For our choice of $\phi(\mathbf{x}_t)$, these integrals are unfortunately not analytically tractable, so they have to be done numerically (Yu et al., 2005, therefore use the error function instead of sigmoid $\phi(\mathbf{x}_t)$ to allow for an analytical solution of at least some of the integrals). Since for computing expectancy values we have to integrate across the whole support of involved random variables x_{it} anyway, strictly it is not necessary to determine the distributions of the $\phi(x_{it})$. In this specific case it merely eases the numerics a bit since the integration can be performed across the finite interval [0, 1] instead of going from -∞ to +∞, but it also illustrates how distributions of (monotonic) functions of random variables may be obtained more generally if needed.

Note that the Kalman filter-smoother recursions give us the fully multivariate posteriors $p(\mathbf{x}_t \mid \{\mathbf{c}_t\}, \boldsymbol{\theta})$ and $p(\mathbf{x}_t, \mathbf{x}_{t-1} \mid \{\mathbf{c}_t\}, \boldsymbol{\theta})$. In the integrals above, however, we used the *marginals* $p(x_{it} \mid \{\mathbf{c}_t\}, \boldsymbol{\theta})$, $p(x_{it}, x_{jt} \mid \{\mathbf{c}_t\}, \boldsymbol{\theta})$ and $p(x_{it}, x_{j,t-1} \mid \{\mathbf{c}_t\}, \boldsymbol{\theta})$. Hence, we have to integrate out the other variables from the multivariate Gaussian, an exercise we have already performed in sect. 7.5.1, eqn. 7.58-7.62 (see also Ch. 2.3 in Bishop, 2006). Defining the matrix partitioning

(9.49)
$$\boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_{ij} & \boldsymbol{\Lambda}_{ij,k} \\ \boldsymbol{\Lambda}_{k,ij} & \boldsymbol{\Lambda}_{k} \end{pmatrix} =: \tilde{\mathbf{V}}_t^{-1},$$

where bivariate submatrix $\boldsymbol{\Lambda}_{ij}$ collects the terms for variables of interest x_{it} and x_{jt}, the marginal means and covariance matrices as used in eq. 9.48 are given by

(9.50)
$$\tilde{\boldsymbol{\mu}}_{ij,t} = (\tilde{\mu}_{it}\ \tilde{\mu}_{jt})^T$$
$$\tilde{\mathbf{V}}_{ij,t} = (\boldsymbol{\Lambda}_{ij} - \boldsymbol{\Lambda}_{ij,k}\boldsymbol{\Lambda}_k^{-1}\boldsymbol{\Lambda}_{k,ij})^{-1}$$

Thus, as it turns out, 'computing' these marginal parameters amounts to nothing else than just picking out the respective (ij) components from the fully multivariate mean $\tilde{\boldsymbol{\mu}}_t$ and covariance $\tilde{\mathbf{V}}_t$ as returned by the Kalman filter-smoother steps (cf. Ch. 2.3 in Bishop, 2006).

Now where we have outlined how to solve the integrals, we can address how to maximize eq. 9.43 w.r.t. parameters $\boldsymbol{\theta} = \{\boldsymbol{\beta}_0, \mathbf{B}, \boldsymbol{\mu}_0, \mathbf{A}, \mathbf{W}, \boldsymbol{\Sigma}\}$. In fact, taking first derivatives of $Q(\boldsymbol{\theta})$, eq. 9.43, we will end up with sets of equations linear in all parameters except for the $\{\boldsymbol{\beta}_{1i}\}$ occurring in the Poisson term of the log-likelihood. Those we'll have to do by numerical schemes (see sect. 1.4). Otherwise, except for matrix operations standard in linear algebra, the only bits we may have to know is how to obtain derivatives of traces (not strictly necessary for maximization, however) and determinants (for these types of things, the Matrix Cookbook by Petersen & Pedersen, http://matrixcookbook.com, is a

highly recommended resource!). For instance, $\partial tr\left(\mathbf{W}^T\boldsymbol{\Sigma}^{-1}\mathbf{W}E[\mathbf{x}_{t-1}\mathbf{x}_{t-1}^T]\right)/\partial\mathbf{W} = \boldsymbol{\Sigma}^{-1}\mathbf{W}E[\mathbf{x}_{t-1}\mathbf{x}_{t-1}^T] + \boldsymbol{\Sigma}^{-1}\mathbf{W}E[\mathbf{x}_{t-1}\mathbf{x}_{t-1}^T]^T$, and $\partial\log|\boldsymbol{\Sigma}|/\partial\boldsymbol{\Sigma} = |\boldsymbol{\Sigma}|^{-1}(|\boldsymbol{\Sigma}|\boldsymbol{\Sigma}^{-1}) = \boldsymbol{\Sigma}^{-1}$ using the chain rule (recall that $\boldsymbol{\Sigma}$ is symmetric, in fact diagonal in model eq. 9.41). To state the final results compactly, we define the following matrices:

(9.51)

$$\mathbf{E}_1 = \sum_{t=2}^{T} E[\phi(\mathbf{x}_{t-1})\phi(\mathbf{x}_{t-1}^T)] \quad , \quad \mathbf{E}_2 = \sum_{t=2}^{T} E[\phi(\mathbf{x}_{t-1})\mathbf{x}_t^T] , \quad \mathbf{E}_3 = \sum_{t=2}^{T} E[\mathbf{x}_{t-1}\phi(\mathbf{x}_{t-1}^T)] ,$$

$$\mathbf{E}_4 = \sum_{t=2}^{T} E[\mathbf{x}_{t-1}\mathbf{x}_{t-1}^T] \quad , \quad \mathbf{E}_5 = \sum_{t=2}^{T} E[\mathbf{x}_t\mathbf{x}_{t-1}^T] \quad , \quad \mathbf{E}_6 = \sum_{t=2}^{T} E[\mathbf{x}_t\mathbf{x}_t^T] ,$$

$$\mathbf{F}_1 = \sum_{t=2}^{T} \mathbf{S}_t E[\phi(\mathbf{x}_{t-1}^T)] \quad , \quad \mathbf{F}_2 = \sum_{t=2}^{T} \mathbf{S}_t E[\mathbf{x}_{t-1}^T] \quad , \quad \mathbf{F}_3 = \sum_{t=2}^{T} E[\mathbf{x}_t]\mathbf{S}_t^T \quad , \quad \mathbf{F}_4 = \sum_{t=2}^{T} \mathbf{S}_t\mathbf{S}_t^T .$$

Suppose we have already solved for $\mathbf{B}^{N\times M} = (\boldsymbol{\beta}_{11},...,\boldsymbol{\beta}_{1N})$ by numerical means (see **MATL9_16** for details), then maximization of eq. 9.43 w.r.t. all other parameters yields

(9.52)

$$\boldsymbol{\beta}_0 = \left(\sum_{t=1}^{T}\mathbf{c}_t\right)\circ\left(\sum_{t=1}^{T} e^{\mathbf{B}E[\mathbf{x}_t]+\frac{1}{2}(\mathbf{B}\tilde{\mathbf{V}}_t\mathbf{B}^T)\circ\mathbf{I}}\right)^{-1}$$

$$\boldsymbol{\mu}_0 = E[\mathbf{x}_1]$$

$$\mathbf{A} = [(\mathbf{E}_5 - \mathbf{E}_2^T\mathbf{E}_1^{-1}\mathbf{E}_3^T + \mathbf{F}_1\mathbf{E}_1^{-1}\mathbf{E}_3^T - \mathbf{F}_2)\circ\mathbf{I}][(\mathbf{E}_4 - \mathbf{E}_3\mathbf{E}_1^{-1}\mathbf{E}_3^T)\circ\mathbf{I}]^{-1}$$

$$\mathbf{W} = (\mathbf{E}_2^T - \mathbf{A}\mathbf{E}_3 - \mathbf{F}_1)\mathbf{E}_1^{-1}$$

$$\boldsymbol{\Sigma} = \frac{1}{T}\Big[\mathrm{var}(\mathbf{x}_1) + \mathbf{E}_6^T - \mathbf{F}_3 - \mathbf{F}_3^T + \mathbf{F}_4 + (\mathbf{F}_2 - \mathbf{E}_5)\mathbf{A}^T + \mathbf{A}(\mathbf{F}_2 - \mathbf{E}_5)^T + \mathbf{A}\mathbf{E}_4^T\mathbf{A}^T + \mathbf{A}\mathbf{E}_3\mathbf{W}^T + \mathbf{W}\mathbf{E}_3^T\mathbf{A}^T$$
$$+ (\mathbf{F}_1 - \mathbf{E}_2^T)\mathbf{W}^T + \mathbf{W}(\mathbf{F}_1^T - \mathbf{E}_2) + \mathbf{W}\mathbf{E}_1^T\mathbf{W}^T\Big]\circ\mathbf{I}$$

where '$\circ$' denotes the element-wise product (recall that **A** and $\boldsymbol{\Sigma}$ are diagonal in the model as defined above), and $\mathbf{I}$ is the identity matrix. This completes the derivations for the nonlinear, non-Gaussian state space model (9.41). One big advantage here is that such a model, once it has been estimated from experimental data, could be used to gain further insight into the dynamics that putatively generated the observations by investigating its fixed points or other dynamical objects, and their stability and bifurcations.

Unfortunately, convergence of the approximate EM algorithm (9.42)-(9.52) is not guaranteed for nonlinear model (9.41), unlike the exact linear case discussed in sect. 7.5.1 (c.f. Wu 1983), but Yu et al. (2005) assure us that practically convergence was mostly given for the cases they had examined. Another note of caution to be made is that model eq. 9.41, as formulated, is over-parameterized (thus not uniquely identifiable); one may want to set $diag(\mathbf{W}) = \mathbf{0}$ to avoid redundancies with the parameters in **A**, and may have to impose further constraints on $\mathbf{B}$ (e.g. by assuming all interactions are captured by **W**), given that linear changes in **B** could be compensated for by rescaling the states accordingly. One may also fix $\boldsymbol{\Sigma} = \mathbf{I}$ (cf. Park et al. 2016), as in factor analysis (sect. 6.4), as variation in the output may not uniquely attributable to $\boldsymbol{\Sigma}$, $\boldsymbol{\Gamma}$, $\mathbf{B}$, or the states. In general, identifiability and uniqueness of solutions remain a problem that plagues both linear and nonlinear state space models (see e.g. Wu, 1983; Roweis & Ghahramani, 2001; Mader et

al. 2014; Auger-Méthé et al., 2016, for further discussion). Regularization techniques as introduced in sect. 2.4 and Ch. 4 may also be helpful in this context, and have been developed for state space models (Buesing et al. 2012).

In closing this section on approximate schemes, one alternative important class of methods for approximating the state path integrals should be mentioned, besides the Laplace approximation introduced in sect. 7.5.3 (eq. 7.89) and the Extended Kalman Filter introduced above. These are methods based on the calculus of variations. In the context of state space models, variational inference methods attempt to approximate the state posterior $p(\mathbf{Z}|\mathbf{X},\theta)$ by minimizing the Kullback-Leibler distance (cf. sect. 6.6, 9.5) between $p(\mathbf{Z}|\mathbf{X},\theta)$ and a parameterized target distribution $q(\mathbf{Z})$ (see Ostwald et al., 2014, for an excellent introduction to variational inference related to state space models; see also Macke et al. 2015).

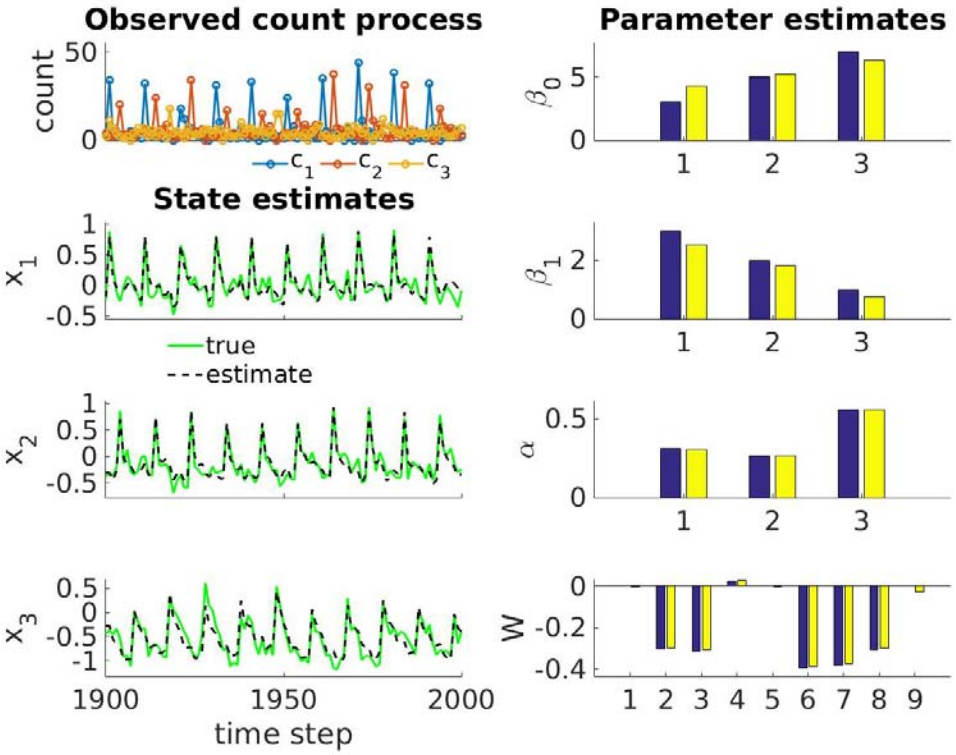

Fig. 9.21. State and parameter estimation in non-linear/ non-Gaussian RNN model eq. 9.41. A 3-unit RNN model was first trained by simple gradient descent (see eqn. 9.13-9.14) to produce a stable nonlinear oscillation across 10 time steps (see **MATL9_16** for details). Left, top: Example of 'spike' count observations produced by Poisson process eq. 9.41 for the three output units. Left, 2nd – 4th row: Illustration of the true RNN trajectories (green) for all 3 units, and the states estimated by the Extended Kalman-Filter-Smoother recursions eq. 9.44-9.46 (dashed black) when correct model parameters θ were provided. Right, from top to bottom: True (blue bars) and estimated (yellow bars) parameters $\boldsymbol{\beta}_0$, $\mathbf{B}$ (diagonal entries), $\mathbf{A}$, and $\mathbf{W}$ when empirical state estimates from long model simulations were provided.

As already raised in the Introduction, we can always discretize a continuous time dynamical system and thus make it amenable to techniques as those described above (e.g. Huys & Paninski 2009, Paninski et al. 2012, Lankarany et al. 2013; Ostwald et al. 2014). But there are also more direct methods, developed in physics in particular, that retain the continuous time description and augment ODE systems with noise processes. The resulting stochastic differential equations are known in statistical physics as Langevin

equations (Risken 1996). Let us illustrate ML estimation within such systems with a simple spiking neuron model, the leaky-integrate-&-fire (LIF) model (credited to Lapicque; see historical notes in Brunel & van Rossum 2007). The LIF model consists of a single linear differential equation for the membrane voltage, representing an electrical circuit which consists of a capacitance in parallel with a 'leak' conductance in series with a battery. The battery drives the membrane voltage toward the cell's (passive) reversal potential, the leak conductance stands for the non-gated ('passive'), open (leak) ion channels, and the capacitance reflects the electrical properties of the bilipid-layer membrane. The model generates spikes through a sharp voltage threshold (and that's the model's nonlinear aspect): Once it's exceeded, a spike is recorded and the membrane potential is reset (that is, spikes are not modeled explicitly as in (9.32)). Adding a noise term (Stein 1965), this becomes

$$\text{(9.53)}\quad \begin{aligned} &\frac{dV}{dt} = \frac{g_L}{C}(E_L - V) + \frac{I}{C} + \xi(t)\,, \quad \xi(t) \sim N(0, \sigma_\xi^2\, \mathrm{I}\{t' = t\}) \\ &if\ V \geq V_{tn} : V \rightarrow V_{reset} \quad , \end{aligned}$$

with ξ(t) a Gaussian white noise process in units of Vs^{-1} with co-variance σ_ξ^2 for $t'=t$, and 0 everywhere else. The term ξ(t) may derive, e.g., from fluctuating synaptic inputs into a neuron (e.g. Brunel & Wang 2001): The total synaptic input into a neuron at any time is usually a sum of hundreds to thousands of small amplitude postsynaptic currents (Markram et al. 1997, London et al. 2010), with synaptic transmission itself being a highly probabilistic process (Jahr & Stevens 1990). Thus, thanks to the central limit theorem, Gaussian assumptions may be well justified (although there is some filtering by the synaptic time constants). By relatively standard methods from statistical physics (Risken 1996), Langevin equations can be translated into a partial differential equation for the (joint) probability distribution of the stochastic variables, P(V) for model (9.53) above, leading into the Fokker-Planck equation (e.g. Brunel & Hakim 1999; Brunel 2000; Brunel & Wang 2001; Hertäg et al. 2014). For model (9.53) it is given by

$$\text{(9.54)}\quad \frac{\partial P(V,t)}{\partial t} = -\frac{\partial}{\partial V}\left[\left(\frac{g_L}{C}(E_L - V) + \frac{I}{C}\right)P(V,t)\right] + \frac{\sigma_\xi^2}{2}\frac{\partial^2 P(V,t)}{\partial V^2}\,.$$

The first term on the right hand side of this equation describes the (systematic) drift in the probability process (coming from the systematic part in eq. 9.53), and the second its diffusion. The Fokker-Planck equation may be thought of as a kind of continuous-time analogon to the Kalman filter equations (7.57).

To solve this PDE, we need to define initial and/or boundary conditions. Since in terms of the interspike-interval process eq. 9.53 describes a 'memoryless' renewal process, i.e. the system is reset to the very same state (V_{reset}) upon each spike, one natural initial condition in this case is that the probability density $P(V,0)$ becomes a delta-impulse at start (previous spike) time t=0. Another boundary condition can be derived from the fact that V cannot lie above the spiking threshold V_{th}, since it will be immediately reset to V_{reset} as soon as it reaches V_{th}, by definition. These boundary conditions can be summarized as (e.g. Paninski et al. 2004; Dong et al. 2011)

$$\text{(9.55)}\quad P(V,0) = \delta(V - V_{reset}) \quad , \qquad P(V_{th},t) = 0\,.$$

In the absence of a spiking threshold, the linear dynamics implied for the membrane voltage by eq. 9.53 would simply give rise to a Gaussian density that drifts and expands in

time. At each instance t in time, $P(V,t)$ would integrate up to 1 to yield a proper density function. In the presence of a threshold, however, $P(V,t)$ is sharply cut off at V_{th}. The cut-off probability 'mass' belongs to the spike event which must sum up with $P(V,t)$ to 1 (Ostojic 2011), i.e.

$$(9.56)\quad \int_{-\infty}^{V_{th}} P(V,t)dV + \int_{0}^{t} f_{spike}(\tau \mid 0)d\tau = 1 \quad \Rightarrow \quad f_{spike}(t \mid 0) = -\frac{\partial}{\partial t}\int_{-\infty}^{V_{th}} P(V,t)dV \,,$$

where $f_{spike}(t \mid 0)$ denotes the conditional spike density at time t, given the last spike occurred at time 0. In other words, the cumulative distribution $\int_{-\infty}^{V_{th}} P(V,t)dV$ of the subthreshold voltage gives the probability that there was no spike yet up to time t, having started at time 0 from V_{reset}, and thus, vice versa, $1-\int_{-\infty}^{V_{th}} P(V,t)dV$ is the cumulative distribution for having a spike. The temporal derivative of this expression therefore yields the conditional (on having the last event at t=0) spike density function in time which can be used to construct the likelihood function for observing a particular sequence of spike times $\{t_1,\ldots,t_M\}$ given the model parameters (Paninski et al. 2004; Dong et al. 2011):

$$(9.57)\quad l_{\{t_m\}}(\{C,g_L,E_L,V_{th},V_{reset},I,\sigma_\xi^2\}) = \prod_{m=2}^{M} f(t_m \mid t_{m-1}) \,,$$

that is, we simply need to evaluate the conditional spike density $f(t \mid t_{last})$ at all empirical spike times t_m because of the renewal properties of the spiking process in the standard LIF model eq. 9.53 (Paniniski et al. 2004).

If there is history-dependence, i.e. carry-over effects from previous interspike-intervals, e.g. due to spike rate adaptation and additional dynamical variables, things can become quite involved and tedious. Such an approach based on conditional densities $f(t_m \mid t_1..t_{m-1})$ has been developed by Dong et al. (2011) for a two-dimensional spiking neuron model with adaptive threshold. One key take-home at this point is simply that stochastic differential equations with Gaussian noise can be transformed by straightforward rules (note how the terms from eq. 9.53 reappear in eq. 9.54) into differential equations for the probability distribution of the dynamic variables. Once we have that, we can – in principle – solve for the 'hidden' state path of the dynamic variables (numerically) and relate it in an ML approach to an observed series of spike times (see also Toth et al. 2011; Kostuk et al. 2012).

All techniques described so far for estimating nonlinear dynamical latent variable models rest on Gaussian approximations to the 'state path integral' and/ or local linearizations, yielding closed-form sets of equations, although parts of it may still have to be done numerically. A different approach which is becoming more and more fashionable and, in theory, enables parameter and state estimation in arbitrarily complex models, is Monte Carlo methods and numerical sampling. Rather than seeking analytical approximations to $p(\mathbf{Z}|\mathbf{X},\theta)$ or $p(\mathbf{X}|\theta)$, where $\mathbf{X}$ are the observed time series data and $\mathbf{Z}$ the state path from the underlying latent process, one attempts to estimate these distributions by sampling from them. We will briefly outline 'particle filters' here, a sequential Monte Carlo method which is kind of a sampling analogue to the Kalman filter recursions, relying on the same temporal dissection as given by eq. 7.57 (Bishop 2006; Durbin & Koopman

2012; Turner & Van Zandt 2012), which we reiterate here for convenience:

$$p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_t) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t)\, p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})}{p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})}$$

$$= \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t) \int_{\mathbf{z}_{t-1}} p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{z}_{t-1})\, p_{\boldsymbol{\theta}}(\mathbf{z}_{t-1} \mid \mathbf{x}_1,...,\mathbf{x}_{t-1}) d\mathbf{z}_{t-1}}{p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})} \tag{9.58}$$

At each time step, the distribution $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_t)$ is represented by a set of 'particles' (samples) $\{\mathbf{z}_t^{(1)},...,\mathbf{z}_t^{(K)}\}$, drawn from $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})$, together with a set of weights $\{w_t^{(1)},...,w_t^{(K)}\}$ which quantify the relative contribution of the particles to the likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t)$ (Bishop 2006; Durbin & Koopman 2012),

$$w_t^{(r)} = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t^{(r)})}{\sum_{k=1}^{K} p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t^{(k)})}. \tag{9.59}$$

In other words, these weights quantify the relative 'consistency' of the samples drawn from $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})$ with the current observation $\mathbf{x}_t$. Note that the samples $\{\mathbf{z}_t^{(k)}\}$ from $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})$ represent the integral in eq. 9.58, while the weights $\{w_t^{(k)}\}$ approximate the term $p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{z}_t) / p_{\boldsymbol{\theta}}(\mathbf{x}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})$. Based on these, we can evaluate the moments of any function of the states $\mathbf{z}_t$, for instance $E[\phi(\mathbf{z}_t)] = \sum w_t^{(k)} \phi(\mathbf{z}_t^{(k)})$ as required for model (9.41) above. Finally, note from eq. 9.58 that we can push our set of particles one step forward in time to yield $p_{\boldsymbol{\theta}}(\mathbf{z}_{t+1} \mid \mathbf{x}_1,...,\mathbf{x}_t)$ (the integral from eq. 9.58, now over $\mathbf{z}_t$) by using our previously established representation of $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_t)$: We first draw samples from the previous particle set $\{\mathbf{z}_t^{(k)}\}$ (that is, from $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{x}_1,...,\mathbf{x}_{t-1})$) with replacement according to the weights $\{w_t^{(k)}\}$, and then generate from these a new set of particles $\{\mathbf{z}_{t+1}^{(k)}\}$ using the transition probability $p_{\boldsymbol{\theta}}(\mathbf{z}_{t+1} \mid \mathbf{z}_t)$, thus yielding $p_{\boldsymbol{\theta}}(\mathbf{z}_{t+1} \mid \mathbf{x}_1,...,\mathbf{x}_t)$. However, it is quite relevant to compute the desired quantities like $E[\phi(\mathbf{z}_t)]$ first, using the weights $\{w_t^{(k)}\}$ and particles $\{\mathbf{z}_t^{(k)}\}$, that is *before* re-sampling from the $\{\mathbf{z}_t^{(k)}\}$ according to the $\{w_t^{(k)}\}$. This is because the re-sampling step will introduce additional variation (see Durbin & Koopman, 2012, for more details on this and other issues related to this approach). This specific implementation of the particle filter, using $p_{\boldsymbol{\theta}}(\mathbf{z}_t \mid \mathbf{z}_{t-1})$ directly to generate new samples, is also called a *bootstrap* particle filter (Durbin & Koopman 2012).

Particle filters have been applied, for instance, by Huys & Paninski (2009) to infer biophysically more detailed models, comprising Hodgkin-Huxley-type equations for a cell's ionic conductances (Koch 1999), or by Paninski et al. (2012) to retrieve synaptic inputs causing observed membrane potential fluctuations.

In closing, we would like to at least briefly mention the *unscented* Kalman filter as a further alternative technique (Durbin & Koopman 2012). It is in some sense between the analytical EKF approximation and the sampling based particle filter. It uses a carefully chosen set of sample points, called *sigma points* in this context, that preserve the first and second moments of the underlying distribution, together with a set of (sigma) weights. The transition or output nonlinearity, respectively, is then applied to these sigma points, based on which the means and covariances are computed using the respective sigma weights

(see Durbin & Koopman, 2012, for more details).

9.3.2 Dynamic Causal Modeling

Dynamic Causal Modeling (DCM) is a statistical-computational framework introduced by Friston et al. (2003) to estimate from neuroimaging data the functional (effective) connectivity among brain areas and its modulation by perturbations and task parameters. For instance, one hypothesis in schizophrenia research is that the cognitive symptoms observed in these patients, like deficits in working memory, can be related to altered functional brain connectivity (Meyer-Lindenberg et al. 2001). To test such ideas one needs a statistical tool to extract from, e.g., the multivariate BOLD measurements this functional connectivity among relevant areas, and how it adapts to the task demands. DCM provides such a tool. It is essentially a state space model (defined in continuous time) with a hidden (unobserved) neural dynamic generating the observed fMRI BOLD signal (but other measurement modalities, like EEG, could be easily accommodated within this framework as well).

The hidden neural state $\mathbf{z}(t)$ is modeled by the set of ordinary differential equations (Friston et al. 2003)

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \sum_{j=1}^{N} u_j \mathbf{B}_j \mathbf{z} + \mathbf{C}\mathbf{u} \,, \tag{9.60}$$

where $\mathbf{A}$ is the matrix of intrinsic (i.e., independent from external inputs) inter-areal connectivities, the $\{u_j\}$ are a series of external perturbations or inputs (specific to the experimental conditions) which convey their impact on the functional connectivity through associated matrices $\mathbf{B}_j$, and directly on the neural dynamics through the weight matrix $\mathbf{C}$. Hence $\theta_N=\{\mathbf{A}, \{\mathbf{B}_j\}, \mathbf{C}\}$ is the set of neural model parameters to be estimated from the observed multivariate time series. Transition model (9.60) is considered a nonlinear model (a so-called *bilinear* form) since it contains product terms of internal states $\mathbf{z}$ and external inputs u. Note that in the basic DCM formulation the transition model is deterministic, since the inputs u are assumed to be known (fixed by the experimental manipulations), but more recent extensions to stochastic transition equations exist as well (Daunizeau et al. 2012).

What exactly is observed depends on the measurement modality and is defined through a set of observation equations (sometimes called a 'forward model' in this context) which links the neural dynamics to the observed quantities. In the case of BOLD signals, Friston et al. (2003) introduced the 'Balloon-Windkessel model' as a simple model of the hemodynamic response to changes in neural activity. It is specified by a set of nonlinear differential equations that describe the various biophysical quantities involved:

$$\begin{aligned}
\dot{s}_i &= z_i - \kappa_i s_i - \gamma_i (f_i - 1) \\
\dot{f}_i &= s_i \\
\tau_i \dot{v}_i &= f_i - v_i^{1/\alpha} \\
\tau_i \dot{q}_i &= f_i E(f_i, \rho_i)/\rho_i - v_i^{1/\alpha} q_i / v_i \\
y_i &= g(v_i, q_i) + \varepsilon_i \,,\ \varepsilon_i \sim WN(0, \sigma^2)
\end{aligned} \tag{9.61}$$

$\dot{s}_i$ is the change in the vasodilatory signal for voxel i, $\dot{v}_i$ the corresponding change in blood vessel volume, q_i the deoxyhemoglobin concentration, and f_i blood inflow. The BOLD signal y_i in voxel i, finally, is a function of volume v_i and deoxyhemoglobin q_i, and some measurement noise ε_i (for more details, see Friston et al. 2003). Thus, the observation equations contain another set of parameters θ_H subject to optimization/ estimation. Note

that there is no feedback from eq. 9.61 to the neural dynamics eq. 9.60 (hence the term 'forward model').

Parameter $\{\theta_N, \theta_H\}$ estimation in this model proceeds through the EM-algorithm within a Bayesian framework (Friston et al. 2003). In fact, at least for the hemodynamic model, reasonable priors can be derived from the literature on this biophysical system. Constraining the model through specification of priors may also be necessary to cope with the potentially large number of parameters in relation to the often rather short fMRI time series.

9.3.3 Special issues in nonlinear (chaotic) latent variable models

Parameter estimation in nonlinear dynamical system models comes with specific problems that we have concealed so far, beyond the complexities involved already in solving nonlinear equations and high-dimensional integrals. Even numerical methods may face insurmountable hurdles if the system exhibits chaotic dynamics in some parameter regimes and not all of the system's variables were observed (as common in neural modeling): In this case, likelihood or LSE functions are usually highly rugged and fractal (Fig. 1.4, Fig. 9.22; Judd 2007; Wood 2010, Abarbanel 2013; Perretti et al. 2013), and numerical solvers are bound to be stuck in very suboptimal local minima or will erratically jump around on the optimization surface. This is because the chaotic or near-chaotic behavior of the unobserved variables on which the observed state probabilities depend causes erratic behavior in the likelihood or LSE function as well. At first sight, these problems seem to open a huge gap between the fields of nonlinear dynamics and statistics.

This section will discuss two potential approaches for dealing with such issues: 1) One may *force* the model system onto the observed trajectory and thereby constrain and regularize the LSE or likelihood function (Abarbanel 2013); 2) One may define the LL function in terms of sensible summary statistics which essentially average out chaotic fluctuations, instead of defining it directly on the original time series (Wood 2010; Hartig et al. 2011; Hartig & Dormann 2013).

Through the first approach, *forcing* the system onto the desired trajectory by specifically designed external inputs during training, the search space is narrowed down and optimization landscapes are smoothened out (a method that falls into an area known as *chaos control*; Ott 2009; Kantz & Schreiber 2004; Abarbanel 2013). We have already briefly visited this kind of idea in the discussion of RNNs in sect. 9.1.2, where during training these were always forced back onto the correct trajectory after observing an output at each time step. Say we have observed a scalar time series $\{U_t\}$, e.g. the membrane potential trace $U(t)$ of a real neuron sampled at discrete times t, which was generated by an underlying higher-dimensional dynamical system for which we have an ODE model (9.32). The central idea is to add a term proportional to the error ($U(t)$ - $V(t)$) to the differential equation for the system variable $V(t)$ which is supposed to mimic $U(t)$ (Abarbanel 2013):

$$
\begin{aligned}
&(a) \quad -C_m\dot{V} = I_L + I_{Na} + I_K + g_M h(V-E_K) + g_{NMDA}\sigma(V)(V-E_{NMDA}) + \kappa(U(t)-V(t)) \\
&(b) \quad \dot{n} = \frac{n_\infty(V)-n}{\tau_n}, \quad n_\infty(V) = \left[1+\exp\left((V_{hK}-V)/k_K\right)\right]^{-1} \\
&(c) \quad \dot{h} = \frac{h_\infty(V)-h}{\tau_h}, \quad h_\infty(V) = \left[1+\exp\left((V_{hM}-V)/k_M\right)\right]^{-1} \quad ,
\end{aligned}
\tag{9.62}
$$

where the other terms are defined as in eq. 9.32. For Fig. 9.22 (**MATL9_17**), we have generated from (9.32) for fixed model parameters in the chaotic regime the voltage trajectory $U(t)$. Starting from arbitrary (since in practice unknown) initial conditions

$\{V_0,n_0,h_0\}$, Fig. 9.22 gives the squared error $\sum(U_t - V_t)^2$ as a function of model parameter g_{NMDA}. Note that it is almost impossible to pick out the true value g_{NMDA}=11.4 that was used in simulating the series $\{U_t\}$ from this very rugged error landscape. Even numerical methods as described in sect. 1.4.3, like genetic algorithms or grid search, may offer little hope when numerous minima and maxima get (infinitesimally) close to each other. However, as we increase the coupling κ, forcing system (9.62) toward the training data $\{U_t\}$, the error landscape becomes increasingly smoother, and more and more clearly reveals the value for g_{NMDA} actually used in generating the trajectory. We refer the reader to the excellent monograph by Henry Abarbanel (2013) for in depth treatment of this approach.

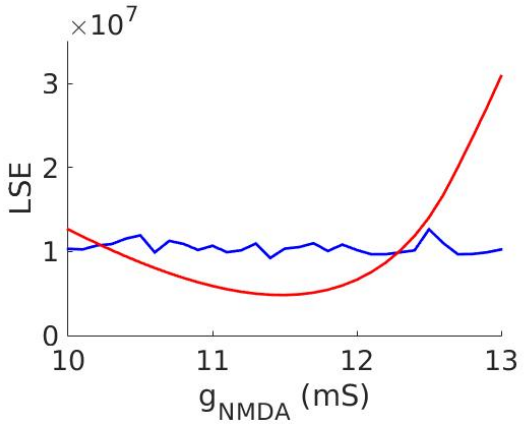

Fig. 9.22. LSE function (defined on the *V* variable only) of model (9.62) with white Gaussian noise in the chaotic regime (g_{NMDA} = 11.4) with κ = 0 (blue curve, no forcing) and κ = 1 (red curve: 100 x LSE, forcing by observed data). Without forcing, the LSE landscape is rugged with only a little dip at the correct value of g_{NMDA}, while forcing smoothens the LSE landscape about the correct value, allowing for parameter estimation by conventional gradient descent. **MATL9_17**.

Another avenue to the problem is to extract suitable summary statistics which capture the essence of the dynamics and are amenable to conventional likelihood/ LSE approaches, instead of formulating the likelihood problem directly in terms of the original trajectories (i.e., time series; Wood 2010; see also Hartig & Dormann 2013). In fact, for a chaotic system, we may not be that much interested in the exact reproduction of originally observed trajectories as these will depend so much on little (unknown) perturbations and differences in initial conditions anyway. This is exactly what makes the optimization functions so nasty for these systems (Judd 2007). Alternatively one may define a set of *summary statistics* **s** that captures the essence of the system dynamics (Wood 2010). These may, for instance, be the coefficients from a polynomial (spline) regression on the system's mutual information or auto-correlation function. If the summary statistics **s** take the form of regression coefficients, normal distribution assumptions $\mathbf{s} \sim N(\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)$ may be invoked, with mean $\boldsymbol{\mu}_\theta$ and covariance $\boldsymbol{\Sigma}_\theta$ functions of the model parameters $\boldsymbol{\theta}$. Based on this, a 'synthetic' log-likelihood, as suggested in the seminal paper by Wood (2010), may be defined as

$$(9.63)\quad l_s(\boldsymbol{\theta}) := -\frac{1}{2}\log\left|\hat{\boldsymbol{\Sigma}}_\theta\right| - \frac{1}{2}(\mathbf{s}-\hat{\boldsymbol{\mu}}_\theta)^T \hat{\boldsymbol{\Sigma}}_\theta (\mathbf{s}-\hat{\boldsymbol{\mu}}_\theta).$$

This synthetic likelihood is commonly a much smoother function of the system's

parameters $\boldsymbol{\theta}$. However, distribution parameters $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$ usually cannot be obtained explicitly as they depend in a complex manner on parameters $\boldsymbol{\theta}$ of the underlying nonlinear dynamical system, and so parametric bootstrapping may have to be used to estimate them: For fixed $\boldsymbol{\theta}$, one generates N_{bs} realizations (samples) of the dynamic process $\dot{\mathbf{x}} = f_\theta(\mathbf{x}, \boldsymbol{\varepsilon}_t)$ (for different noise realizations and initial conditions), computes the defined set of summary statistics $\mathbf{s}^*$ from each of them, and plugs those into the mean and covariance estimates.

The question remains of how to maximize the log-likelihood. An extensive search through parameter space $\boldsymbol{\theta}$ will be computationally prohibitive in most cases, and so Wood (2010) suggests the technique of *Markov Chain Monte Carlo (MCMC)* sampling as a way out. MCMC is a family of very general probabilistic numerical devices employed in difficult terrain, applicable when samples have to be drawn from complicated, high-dimensional, analytically intractable probability distributions (cf. sect. 1.4.3). In the current setting, one starts with an initial guess $\boldsymbol{\theta}_0$ and then performs a kind of 'random walk' (guided by the underlying density) in parameter space according to the following update rules (Wood 2010):

$$(9.64)\quad \boldsymbol{\theta}_n = \begin{cases} \boldsymbol{\theta}_{n-1} + \boldsymbol{\eta}_n\,,\ \boldsymbol{\eta}_n \sim N(\mathbf{0}, \boldsymbol{\Psi})\,, \textit{ with } pr\left(\min[1, e^{l_s(\boldsymbol{\theta}_{n-1}+\boldsymbol{\eta}_n) - l_s(\boldsymbol{\theta}_{n-1})}]\right) \\ \boldsymbol{\theta}_{n-1} \quad \textit{otherwise} \end{cases}.$$

For $n \to \infty$, the empirical distribution for the set of samples $\boldsymbol{\theta}_n$ converges to the true underlying distribution $p(\boldsymbol{\theta})$ (Bishop 2006).

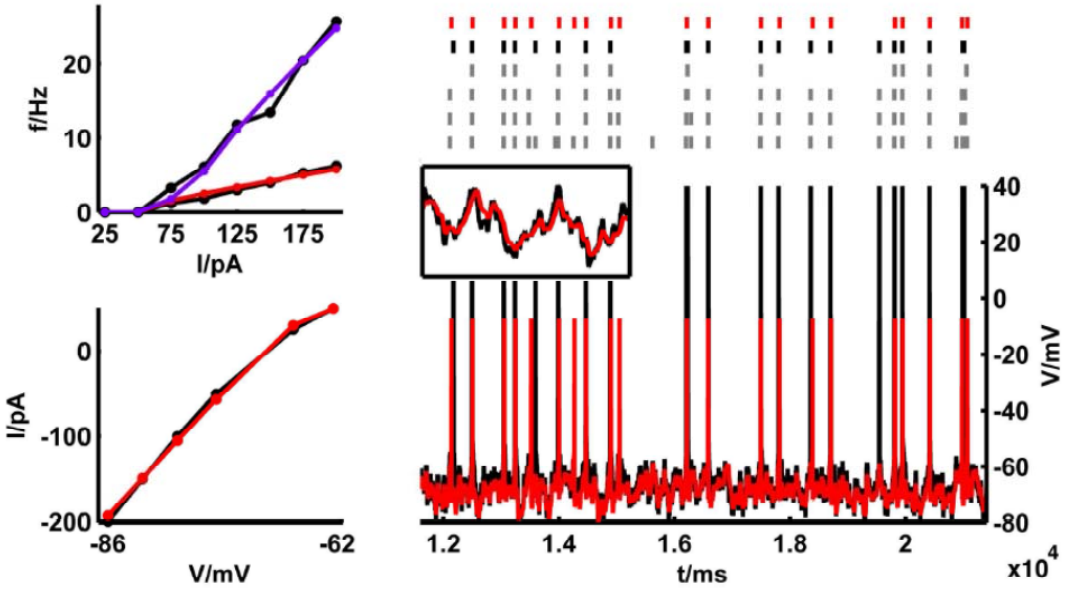

Fig. 9.23. Parameter estimation in a variant of model (9.65) using empirical training data consisting of initial (defined as the reciprocal of the first inter-spike-interval) and steady-state (after the neuron has been settling into a stable spiking mode) f/I curves (top left), and sub-rheobase I/V curves (bottom left). These were combined into a single LSE criterion. Training data were generated by standard DC current step protocols. Experimentally recorded data are indicated in black in all panels, while model fits on the training data are indicated in blue and red on the left. On the right are experimental voltage traces (bottom, black curve) and spike trains from repetitions with identical stimulus

conditions (black and gray), as well as model predictions shown in red. For this prediction set, neurons were stimulated with fluctuating input currents not part of the training set (the training data were obtained with DC injections only). Reproduced from Hertäg et al. (2012).

We close this section with a neuroscientific example that highlights some of the issues important, in the author's view, in estimating neuro-computational models from experimental data. For parameter estimation from voltage recordings in a nonlinear single-neuron model, Hertäg et al. (2012) derived closed-form expressions for the instantaneous and steady-state f/I (spike rate over current) curves. The details of the model are not so important here (in fact, it was not originally formulated as a statistical latent variable model; see Pozzorini et al., 2015, for a state space approach to single neuron estimation). But to give the reader at least some idea, the model was modified from the "adaptive exponential leaky-integrate-&-fire" (AdEx) model introduced by Brette & Gerstner (2005), defined by

$$
\begin{aligned}
& C\frac{dV}{dt} = g_L(E_L - V) + g_L\Delta_T e^{(V-V_T)/\Delta_T} + sI - w \\
& \tau_w \frac{dw}{dt} = a(V - E_L) - w \\
& if\ V \geq V_{th} : V \to V_{reset}\ ,\ w \to w + b \quad ,
\end{aligned}
\tag{9.65}
$$

where C is the membrane capacitance, g_L a leak conductance with reversal (resting) potential E_L, I some externally applied current, and w an adaptation variable with time constant τ_w. The second, exponential term in the voltage equation is supposed to capture the exponential upswing at the onset of a spike. A simplification of the model allowing for closed-form f/I expressions was achieved by setting a=0 and assuming $\tau_w >> \tau_m = C/g_L$ (*separation of time scales*; cf. Strogatz 1994). Instead of attempting to evaluate a likelihood directly on the experimentally observed series of spike times or voltage traces, the initial and steady-state f/I curves which *summarize aspects of the system dynamics* were used in a cost function (Fig. 9.23, left). It turned out that despite this apparent discard of information, the estimated neuron model was often almost as good in predicting spike times in fluctuating voltage traces not used for model fitting as were the real neuron's own responses on different trials under the very same stimulus conditions (Fig. 9.23, right). That is, the average discrepancy between model predictions and real neuron responses was on about the same order as the discrepancy between two response traces from the same neuron with identical stimulus injections. Trying to 'fine-tune' the model predictions through an input scaling parameter (parameter s in eq. 9.65) actually resulted in over-fitting the estimated spike rates compared to the real physiological rate variation.

There are three take-homes from this: 1) In keeping with Wood (2010), often it might scientifically make more sense to perform model estimation on summary statistics that capture those aspects of the data deemed most important for characterizing the underlying dynamical system. 2) A good biological model should make predictions about data domains *not at all* visited during training, i.e. with validation data actually drawn from statistical distributions which *differ* from those employed in model estimation (this goes beyond conventional *out-of-sample prediction* as discussed in Ch. 4). In the case of the Hertäg et al. study, the test samples actually had input statistics and dynamical properties quite different from those used for model training. 3) It may also be illuminating to compare model performance, where possible, to a nonparametric nonlinear predictor like (8.1-8.3) formed directly from the data (see also Perretti et al. 2013), on top of potentially pitching different models against each other on the same data set.

9.4 Reconstructing State Spaces from Experimental Data

We had already introduced in sect. 8.1 the technique of temporal delay embedding, where from a scalar (or multivariate) time series $\{x_t\}$ we form delay embedding vectors (Abarbanel 1996; Kantz & Schreiber 2004; Sauer 2006)

$$\mathbf{x}_t = (x_t, x_{t-\Delta t}, x_{t-2\Delta t}, ..., x_{t-(m-1)\Delta t}) \quad (9.66)$$

An important and powerful result in nonlinear dynamics is the *delay embedding theorem* (Takens 1981) and its extensions by Sauer et al. (1991), according to which one can reconstruct (in the sense of dynamical and topological equivalence) the original attractor of a multivariate, higher-dimensional system from just univariate (scalar) measurements of one of the system's variables (or some function of the system's variables which satisfies certain conditions), yielding a 1:1 mapping between trajectories $\mathbf{y}_t$ in the original space and $\mathbf{x}_t$ in the delay-embedding space. This sounds a bit like magic, and in fact it is only true under certain conditions, e.g., strictly, only in *noise-free* systems in which all degrees of freedom are coupled (i.e., in which all variables directly or indirectly influence each other), and provided that the embedding dimension m is chosen high enough (at least > $2\times$ the so-called box counting dimension of the underlying attractor; see Kantz & Schreiber 2004; Sauer et al. 1991). Intuitively, we may understand this result by interpreting the univariate measurements $\{x_t\}$ from one variable as a probe into the system: There are many ways in which a single measurement x_t could have come about, but as we take more and more time points $x_{t-\Delta t} \ldots x_{t-(m-1)\Delta t}$ into account, we impose more and more constraints on the nature of the underlying system. Ultimately, with a long enough time series, and as long as it is not corrupted by any noise, any dynamical system $\dot{\mathbf{x}} = f(\mathbf{x})$ will leave a *unique signature* on each probe that could have been produced only by one specific type of attractor dynamic. Another way to see this is that we replace missing information about other system variables by time-lagged versions from those variables we did observe: If all degrees of freedom are coupled, then the unobserved variables will leave a footprint in the time series of those observed. Of course, one caveat here is that empirically the noise may be too large to recover those footprints.

Practically speaking, we need to determine a proper delay Δt and embedding dimension m (Abarbanel 1996; Kantz & Schreiber 2004). Although the choice of Δt, in a noise-free system, is theoretically irrelevant, i.e. does not affect our ability to reconstruct the attractor, practically it will, depending on the number of data points we have, the amount of noise, and so forth (Fig. 9.24). If Δt is chosen too small, consecutive delay vectors will be highly correlated, will tend to cluster along the main diagonal, and structure may be difficult to discern from noise (Fig. 9.24, center-left). If, on the other hand, Δt is too large, consecutive vectors may be completely unrelated, such that the system may appear to erratically jump between points in state space (Fig. 9.24, bottom-right). Hence the ideal choice for Δt, at which trajectories become nicely unfolded yet retain structure distinct from a random whirl, will lie somewhere in between. As a rule of thumb, one may choose the value at which auto-correlations in the time series dropped to about e^{-1} (Takens 1981; Cao et al. 1998).

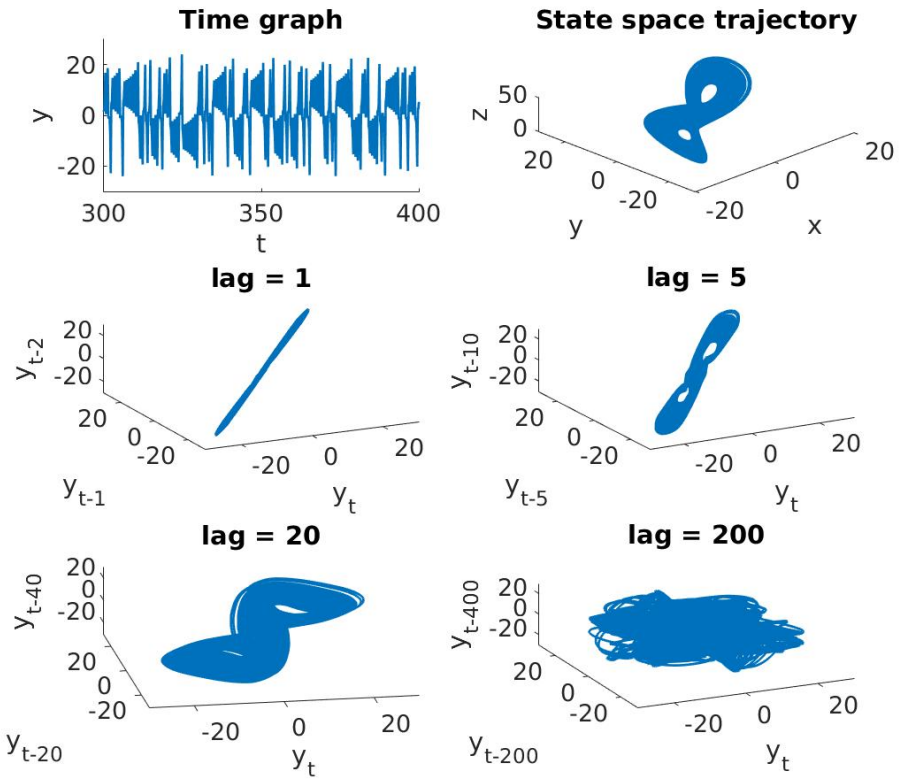

Fig. 9.24. Time graph (top left) and state space (top right) of the Lorenz equations within the chaotic regime (see **MATL9_18**). Center and bottom row show m=3 dimensional embeddings of the time series of system variable y for different lags Δt (an embedding dimension of 3 is, strictly, not sufficient for reconstructing the Lorenz attractor, but will do for the purpose of illustration). Nice unfolding of the attractor is achieved for intermediate lags, while for too small lags (1) data points cluster close to a one-dimensional subspace and for too large lags (200) data points tend to erratically hop around in space, obscuring the attractor structure. **MATL9_18**.

As emphasized above, the choice for Δt is theoretically irrelevant and “only” of practical importance. This is different, of course, for the choice of delay embedding dimension m which has to be larger than twice the (usually unknown) ‘box-counting’ attractor dimension (loosely, the box-counting dimension assesses how the number of non-empty boxes from a regular grid covering a set scales with box size ε; Kantz & Schreiber 2004). Most commonly a suitable m is estimated by determining the number of 'false neighbors' (Kennel et al. 1992). These are points which fall close together in a lower-dimensional state space projection, but are really far from each other in the true (full) state space. If, for instance, two orbits live in a truly 3-dimensional space, of which only two coordinates could be accessed empirically, then points which are separated along the third, non-accessed dimension but are close on the other two, observed dimensions, may be difficult to discern or even fall on top of each other (Fig. 9.25; see also Fig. 6.3). Such points may reveal themselves if the distance between them does not grow continuously with (embedding) dimensionality but suddenly jumps from quite low in m dimensions to much higher in m+1 dimensions. Hence we may set up the following criterion (Kennel et al. 1992; Kantz & Schreiber 2004):

(9.67) $$n_{FN} = \frac{1}{T-(m+1)\Delta t} \sum_{t=1}^{T-(m+1)\Delta t} \mathrm{I}\left\{ \frac{\left\| \mathbf{x}_t^{(m+1)} - \mathbf{x}_\tau^{(m+1)} \right\|}{\left\| \mathbf{x}_t^{(m)} - \mathbf{x}_\tau^{(m)} \right\|} > \theta \right\} \quad with\ \tau = \arg\min_k \left\| \mathbf{x}_t^{(m)} - \mathbf{x}_k^{(m)} \right\| ,$$

which simply counts the relative number of points for which such a jump, defined by some threshold θ, occurs across the series of length T when moving from m to m+1 dimensions. For fairness, one may exclude from this count all pairs which already have a distance in the m-dimensional space on the order of the data standard deviation, since they have reached the bounds and cannot extend much further. One may then choose m, for instance, such that n_{FN}<1%.

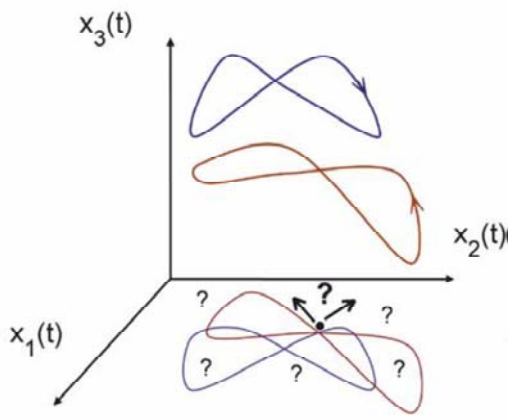

Fig. 9.25. Two trajectories (blue and brown) well separated in the (x_1,x_2,x_3)-space fall on top of each other in the (x_1,x_2)-projection, yielding many intersections and 'false neighbors' around which the flow is no longer uniquely defined. Reproduced from Balaguer-Ballester et al. (2011).

Other criteria for choosing m that have been proposed, and have been generalized to multivariate time series, are based on time series prediction errors (Abarbanel 1996; Cao et al. 1998; Vlachos & Kugiumtzis 2008). The idea is that the true system underlying the observed series $\{\mathbf{x}_t\}$ follows a continuous, smooth map (since we are sampling at discrete time points t), $\mathbf{z}_{t+1} = G(\mathbf{z}_t)$. Since G is continuous and sufficiently smooth, points $\{\mathbf{z}_t\}$ with a similar history should have a similar future $\{\mathbf{z}_{t+1}\}$, i.e. $\mathbf{z}_t$ close-by in state space should yield one-step-ahead predictions that do not stray away too much from each other, and hence the same should be true for the reconstructed map $\mathbf{x}_{t+1} = F(\mathbf{x}_t)$, if the attractor was indeed reconstructed correctly (Cao et al. 1998; Vlachos & Kugiumtzis 2008). Thus, within the optimal reconstruction, $\mathbf{x}_{t+1}$ should be optimally predictable from spatial neighbors of $\mathbf{x}_t$. Assume we have observed vectors $\mathbf{x}_t = (x_{t1}, x_{t2}, ..., x_{tp})$ of p variables. We seek an embedding $\tilde{\mathbf{x}}_t = (x_{t1}, x_{t-\tau_1,1}, x_{t-2\tau_1,1}, ..., x_{t-(m_1-1)\tau_1,1}, x_{t2}, ..., x_{t-(m_2-1)\tau_2,2}, ..., x_{tp}, ..., x_{t-(m_p-1)\tau_p,p})$ with parameters $\boldsymbol{\theta} = \{m_1, \tau_1, m_2, \tau_2, ..., m_p, \tau_p\}$ such that a prediction error criterion like

(9.68) $$Err(\boldsymbol{\theta}) = \frac{1}{T-(m_{\max}-1)\tau_{\max}} \sum_{t=(m_{\max}-1)\tau_{\max}+1}^{T} (\mathbf{x}_t - \hat{\mathbf{x}}_t)\boldsymbol{\Sigma}_x^{-1}(\mathbf{x}_t - \hat{\mathbf{x}}_t)^T$$

is minimized. Note that although original time series $\{\mathbf{x}_t\}$ will be embedded according to set $\boldsymbol{\theta}$, the prediction error is only obtained on those vector components defined in the original series, thus making this measure strictly comparable across different dimensionalities of

the embedding space. Typical predictors could be chosen based on kNN (sects. 2.7, 8.1) or LLR (sect. 2.5): In the case of kNN (that is, the *zeroth-order* or *locally constant* predictor; see sect 2.7), we define a local neighborhood $H_\varepsilon(\tilde{\mathbf{x}}_{t-1}) = \{\tilde{\mathbf{x}}_{s-1} \mid d(\tilde{\mathbf{x}}_{t-1}, \tilde{\mathbf{x}}_{s-1}) \le \varepsilon, \mid t-s \mid > (m_{\max}-1)\tau_{\max}+1\}$ for each $\tilde{\mathbf{x}}_t$ in the embedded series in turn, and make the (one-step-ahead) prediction

$$\text{(9.69)}\quad \hat{\mathbf{x}}_t = \frac{1}{|H_\varepsilon(\tilde{\mathbf{x}}_{t-1})|} \sum_{\tilde{\mathbf{x}}_{s-1} \in H_\varepsilon(\tilde{\mathbf{x}}_{t-1})} \mathbf{x}_s \ .$$

It is important that we exclude from the neighborhoods $H_\varepsilon(\tilde{\mathbf{x}}_{t-1})$ *temporal neighbors* up to some horizon $\mid t-s \mid > (m_{\max}-1)\tau_{\max}+1$, at the very least exclude all embedding vectors which share components with the prediction target $\mathbf{x}_t$, since we are interested in reconstructing the *spatial*, *topological aspects* of the attractor and not just temporal auto-correlations. In the case of LLR, we would use the vectors in $H_\varepsilon(\tilde{\mathbf{x}}_{t-1})$ to fit a locally linear model (Abarbanel 1996)

$$\text{(9.70)}\quad \mathbf{X}_{H(t)} = \mathbf{b}_{t0} + \tilde{\mathbf{X}}_{H(t-1)}\mathbf{B}_t \ ,$$

where matrix $\tilde{\mathbf{X}}_{H(t-1)}$ collects all the full embedding vectors in $H_\varepsilon(\tilde{\mathbf{x}}_{t-1})$, and $\mathbf{X}_{H(t)}$ all the corresponding one-step-ahead values (i.e., in the original, non-embedded space). This is indeed nothing else than fitting an AR(m) model (with additional time lags Δt among predictor variables) *locally* on all data points in $H_\varepsilon(\tilde{\mathbf{x}}_{t-1})$ except target point $\mathbf{x}_t$ to their respective 1-step-ahead values, and then using the estimated coefficient matrix $\mathbf{B}_t$ to predict $\mathbf{x}_t$:

$$\text{(9.71)}\quad \hat{\mathbf{x}}_t = \mathbf{b}_{t0} + \tilde{\mathbf{x}}_{t-1}\mathbf{B}_t \ .$$

Note that in eq. 9.71 we used time-lagged versions of $\mathbf{x}_{t-1}$ (that is, the full delay-vector) to make the prediction, while in (9.69) the delay embedding is only used to define the local neighborhood. Especially in higher dimensions, however, local neighborhoods might be very sparse, such that estimating a full coefficient matrix $\mathbf{B}_t$ locally for each t may not be sensible. Neither may increasing neighborhood radius ε too much (thus approaching a globally linear model) be a solution for highly nonlinear dynamics. One could try to get around this and reduce the degrees of freedom by parametrizing coefficient matrices $\mathbf{B}_t$ in some way, or by a factorization like $b_{ij} = a_i \cdot c_j$. As a note of caution, with such model fitting we are potentially trespassing into statistical terrain, in the sense that the optimal embedding may reflect more the optimal bias-variance-tradeoff given a finite noisy sample (see Ch. 4) rather than the dimensionality of the underlying dynamical object.

For implementation of the above concepts and other tools based on state space reconstructions the reader is referred to the excellent TISEAN (time series analysis) toolbox by Hegger et al. (1999; see also Kantz & Schreiber 2004).

Because of the huge role that attractor concepts play in theoretical and computational neuroscience, there has been a longer-standing interest in identifying such dynamical objects from experimental recordings. In the nervous system, this turns out to be a quite difficult endeavor, because of the huge complexity of the underlying system, the many degrees of freedom of which only a few can be accessed experimentally at any one time, and the usually high noise levels and many uncontrollable sources of variation (like

inputs from other brain areas and the environment). So the empirical demonstration of attractor-like behavior so far has been mainly confined to comparatively simple or reduced systems under well-controlled conditions, like primary sensory areas in invertebrates (Mazor & Laurent 2005; Niessing & Friedrich 2010), and with averaging across a large number of trials. Or it has relied on quite indirect evidence, like the tendency of recorded neural networks to converge to one of several discrete states with gradual changes in the input (Niessing & Friedrich 2010; Wills et al. 2005). In the former set of studies, however, mainly input (stimulus)-driven convergence to one stable state was shown, while the latter results may potentially also be explained through the presence of a strong input nonlinearity which separates initial states.

A more direct demonstration of attractor-like behavior would be given if systems were shown to return to previous states after temporary perturbations (Aksay et al. 2001), or if convergence to stable states could be demonstrated. As Fig. 9.25 makes clear, convergence – even if present – may be very difficult to reveal because experimentally one usually has access to only a tiny subset of the dimensions (neurons) that describe the system. Hence, in the experimentally assessed subspaces neural trajectories are likely to be heavily entangled, folded, and projected on top of each other. Balaguer-Ballester et al. (2011; Lapish et al. 2015) tried to resolve this issue by combining basis expansions (see sect. 2.6) with temporal delay embedding as described above. In these so delay-embedded and expanded spaces, neural trajectories indeed started to disentangle, and a convergent flow to specific task-relevant states became apparent. This convergence was furthermore sensitive to pharmacological manipulations (Lapish et al. 2015).

A possible effect obtained from combining delay embedding with dimensionality reduction techniques as discussed in Ch. 6, sometimes desirable for the purpose of visualization, is a temporal smoothing of trajectories as time-lagged variables get combined on dimensions in the reduced space.

9.5 Detecting causality in nonlinear dynamical systems

The implementation of Granger causality through AR models is problematic for various reasons (Liu et al. 2012; Sugihara et al. 2012). First and obviously, the predictive relationship may be nonlinear, which is likely to be the case in many if not most biological systems. Second, this will be a problem in particular when the coupling is quite weak, as will be true for instance for synaptic connections between neurons. Third, Granger causality cannot differentiate between the (quite common) scenario where some variables **X** and **Y** are driven by a common source vs. the case of uni- or bidirectional causal coupling among them.

At least the first point is accounted for by an information-theoretic measure dubbed transfer entropy (Schreiber 2000). It comes back to Granger's probabilistic definition (eq. 7.43) and directly quantifies how much more we know about the conditional distribution of $\mathbf{x}_t$ if we take into account not only $\mathbf{x}_t$'s own past, but in addition another variable $\mathbf{y}_t$ from which we suspect a causal influence on $\mathbf{x}_t$ (Schreiber 2000):

$$(9.72)\quad T_{y\to x}(\tau) = \sum_{\{\mathbf{x}_t,\ldots\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},\ldots\mathbf{y}_{t-\tau}\}} p(\mathbf{x}_t,\mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},\ldots\mathbf{y}_{t-\tau}) \log \frac{p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},\ldots\mathbf{y}_{t-\tau})}{p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau})} .$$

This is a type of *Kullback-Leibler divergence* (see also sect. 6.6) between the conditional distributions $p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},\ldots\mathbf{y}_{t-\tau})$ and $p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau})$: If **Y** does not help us in predicting **X**, $p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},\ldots\mathbf{y}_{t-\tau}) \approx p(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ldots\mathbf{x}_{t-\tau})$, so eq. 9.72 quantifies the *excess information* about $\mathbf{x}_t$ that we obtain by taking $\mathbf{y}_{t\text{-past}}$ into account (Schreiber 2000; Kantz & Schreiber 2004). The measure is also asymmetric (directional), so that, in general, it will

not give the same result if we swap the roles of **X** and **Y** in eq. 9.72. In practice, the $(\mathbf{x}_t,...\mathbf{x}_{t-\tau},\mathbf{y}_{t-1},...\mathbf{y}_{t-\tau})$ space will have to be binned such as to yield sufficiently high cell counts to evaluate the probabilities in eq. 9.72 (either way, with several variables and time lags involved, an impressive amount of observations may be required).

Another recent approach (introduced by Sugihara et al. 2012) that accounts for all of the three issues noted with AR-based Granger causality above is *Convergent Cross-Mapping* (CCM). CCM builds very much on Taken's theorem (see sect. 9.4) and the reconstruction of attractor manifolds through delay embedding. Assume that variables *X* and *Y* form a dynamical system where *X* exerts a strong influence on *Y* but not vice versa, i.e., *Y* has no or only a weak influence on *X*. Yet, assume *Y* is not fully synchronized (that is, completely determined) by *X* either, but still obeys its own dynamics. This implies by Taken's delay embedding theorem that it should be possible to reconstruct the full (*X,Y*)-system dynamics from *Y*, since it records aspects of *X*'s dynamics, i.e. *X* would leave a signature in the time series of *Y* (Sugihara et al. 2012). The reverse would not be true (at least not to the same degree), however, that is we could not reconstruct the full system dynamics from *X* only, since *X* is *not influenced* by *Y* and hence *oblivious to the dynamics of Y* (as long as *X* doesn't drive *Y* to full synchronization). Thus, for probing a causal relationship $X \rightarrow Y$, we would try to recover *X* from *Y*'s history, which seems to put the usual Granger causality logic upside down. More specifically, one starts by reconstructing attractor manifolds $\mathbf{M}_X$ and $\mathbf{M}_Y$ of variables *X* and *Y* through suitable delay-embeddings $\mathbf{x}_t = (x_t, x_{t-\Delta t}, x_{t-2\Delta t},..., x_{t-(m-1)\Delta t})$ and $\mathbf{y}_t = (y_t, y_{t-\Delta t}, y_{t-2\Delta t},..., y_{t-(n-1)\Delta t})$ (Sugihara et al. 2012). To make a prediction $\mathbf{y}_t \rightarrow \hat{\mathbf{x}}_t$, one collects the *k* nearest spatial neighbors $\mathbf{y}_s \in H_k(\mathbf{y}_t)$, similar as in nonlinear prediction, sect. 8.1 (Sugihara et al. choose $k=n+1$, the minimum number of points required to form a bounding simplex in an *n*-dimensional space). From these one obtains the predictor

$$\text{(9.73)}\quad \hat{\mathbf{x}}_t = \frac{\sum\{w_{s,t}\mathbf{x}_s \mid \mathbf{y}_s \in H_k(\mathbf{y}_t)\}}{\sum w_{s,t}}.$$

Sugihara et al. (2012) suggest an exponential weighting of neighbors $\mathbf{x}_s$, with the weights $w_{s,t} = \exp(-\|\mathbf{y}_s - \mathbf{y}_t\| / \|\mathbf{y}_{s^*} - \mathbf{y}_t\|)$ an exponential function of the Euclidean distance between the associated sample and target points on $\mathbf{M}_Y$ and $\mathbf{y}_{s^*}$, the point from H_k closest to $\mathbf{y}_t$. Based on this prediction, the strength of interaction may then be quantified through the correlation between predictors $\hat{\mathbf{x}}_t$ and targets $\mathbf{x}_t$, or some form of prediction error like eq. 9.68 (sect. 9.4). Hence, if *X* drives *Y* but not vice versa, then $\mathbf{M}_Y$ would contain dynamical information about $\mathbf{M}_X$ but not necessarily vice versa, and points close-by on attractor manifold $\mathbf{M}_Y$ should correspond to points close-by on $\mathbf{M}_X$ (but not necessarily vice versa, if *X* itself evolves independently of *Y*).

There is one more important ingredient (Sugihara et al. 2012): If there are these hypothesized structural relationships between reconstructed attractor manifolds $\mathbf{M}_Y$ and $\mathbf{M}_X$, then our prediction should improve as trajectories on $\mathbf{M}_Y$ become denser and hence the quality of the neighborhoods H_k gets better. In other words, the correlation between predicted $\hat{\mathbf{x}}_t$ and actual $\mathbf{x}_t$ outcomes should increase with time series length *T*, and should *converge* to some upper bound imposed by other factors like coupling strength and noise in the system. This convergence with time series length is an important property to distinguish true causal interactions from other factors like contemporary correlations between *X* and *Y*. If the neighborhood H_k is just filled up by (uncorrelated) random fluctuations in *X* and *Y*, there would be no convergence.

A drawback of the methods discussed here is that they usually require quite a large

amount of data, and tend to work best in comparatively small systems (few measured variables) with relatively little noise. If our data come from complex high-dimensional systems with plenty of noise, even if they are nonlinear, AR-based Granger causality may still be the best option left.

Transfer entropy measures have been used mainly in the context of MEG or EEG recordings to reveal causal or directed interactions and information flow among brain areas (Lindner et al. 2011; Wibral et al. 2011), but modified versions to tackle spike train interactions also exist (Li & Li 2013). As of the time of writing, they remain, however, much less popular than the methods based on linear models (sect. 7.4), partly presumably because of their higher requirements in terms of data “quality” and quantity.