Supplementary Material to:

On Computational Power and the Order-Chaos Phase Transition in Reservoir Computing

1 Computational Performance for Further Example Tasks

In this section the performance measure p_{exp} defined in eq. (1) in the main text [1] for two further example tasks is shown. In Fig. 1 $p_{\text{exp}}(C, \text{RAND}_5)$ is plotted, averaged over 20 randomly chosen 5-bit functions RAND_5, circuits C, initial conditions and input streams. A uniform distribution over all 2^{2^5} functions $\text{RAND}_5 : \{-1,+1\}^5 \to \{-1,1\}$ was applied. In Fig. 2 the quantity $p_{\text{exp}}(\text{C}, \text{AND}_5)$ for the 5-bit AND-function is plotted, i.e. the target output at time t is $\text{AND}_{5,\tau}(u,t) = \max_{i\in\{1,\ldots,5\}}\{u(t-\tau-i)\}$ for delay τ. Shown results are averages over 10 circuits C, initial conditions and input streams. The results shown in Fig. 1 and 2 are qualitatively similar to the results on the PAR-task reported in [1].

2 Definition and Calculation of p_∞

In this section we give a detailed definition of the heuristic performance measure p_∞ and outline an approach to calculate it efficiently.

2.1 Notation

Without loss of generality we may set the readout time t to 0 as the input random process $u(.)$ is assumed to be stationary. Hence the target task of the n last input bits (delay $\tau = 0$) is a function f_T of $u(-1), \ldots, u(-n)$. We introduce the following useful notation for the input:

$$\begin{aligned} u^i &:= (u^i(-1), u^i(-2), \ldots) \in \{-1,1\}^{\mathbb{N}} \\ u^{i,t} &:= (u^i(t-1), u^i(t-2), \ldots) \in \{-1,1\}^{\mathbb{N}} \end{aligned}$$

For a vector $x = (x_1, \ldots, x_N) \in \mathbb{R}^N$ we use the the p-norm $\|.\|_p$ with $p = 1$ (the Manhattan norm) for measuring distances:

$$\|x\|_1 = \sum_{i=1}^{N} |x_i|.$$

2.2 Definition of p_∞

In the following we consider two instances N1 and N2 of the same network which solely differ in the input streams $u^1(.)$ and $u^2(.)$ they receive. Let $d(k)$ be the normalized expected distance between the two network states $x^1(0)$ and $x^2(0)$ of N1 and N2 at time 0 for the case where the inputs $u^1(.)$ and $u^2(.)$ only differ at the single time step $t = -k$. Hence for given u^1, the sequence u^2 is determined by:

$$\begin{aligned} u^2(-i) &= u^1(-i) \;\; \forall i \in \mathbb{N} \wedge i \neq k \\ u^2(-k) &= -u^1(-k). \end{aligned}$$

$\forall k \in \mathbb{N}$ the inputs $u^1(-k) \in \{-1, +1\}$ are iid. with probability $p(u^1(-k) = 1) =: 0.5$. $d(k)$ is then formally defined as:

$$d(k) := \frac{1}{N} \left\langle \left\langle \|x^1(0) - x^2(0)\|_1 \right\rangle_W \right\rangle_u . \tag{1}$$

The average $\langle . \rangle_u$ is taken over all inputs $u^1(.)$, $u^2(.)$ from the ensemble defined above and all initial conditions of the network; $\langle . \rangle_W$ denotes the average over all weight matrices W. If the network is not in the fading memory regime (or equivalently: if it does not have the echo state property, for a formal definition see [2] or [3]) $d(k)$ defined in (1) might well depend on the distribution of initial conditions from which the evolution of x^1 and x^2 starts; we address this subtle point below. We define p_∞ in the following way:

$$p_\infty = \max\{ \lim_{k \to \infty} \left(d(2) - d(k) \right), 0\} = \max\{d(2) - d(\infty), 0\}.$$

2.3 The Annealed Approximation

We want to calculate the distance $d(k)$ defined in (1) for large networks $N \to \infty$ using the annealed approximation (AA) introduced in [4]. The AA consists of approximating the original dynamics of the network by assuming that the weight matrix W is drawn iid. at every time step. First we notice that in the AA and the limit $N \to \infty$ the following holds:

$$\begin{aligned} d(k) \quad &= \quad \left\langle \left\langle \frac{1}{N} \sum_{b=1}^{N} |x_b^1(0) - x_b^2(0)| \right\rangle_W \right\rangle_u = \left\langle \left\langle |x_a^1(0) - x_a^2(0)| \right\rangle_W \right\rangle_u \quad \forall a \in \mathbb{N} \\ d(k) \quad &\xrightarrow{N \to \infty} \quad \left\langle \left| \sum_{i,j=0}^{2^m - 1} p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})(s_i - s_j) \right| \right\rangle_u \quad \forall a \in \mathbb{N}. \end{aligned} \tag{2}$$

Here $p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})$ denotes the joint probability of finding $x_a^1(t)$ in the state s_i and $x_a^2(t)$ in the state s_j given the input $u^{1,t},\ u^{2,t}$. Due to the AA this probability is independent of the node index a. Moreover $p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t})$ determines the distribution of the recurrent feedback for the next time step. Hence, in the AA and for $N \to \infty$ the state of the network is completely described by the joint distribution of a pair of coordinates $x_a^1(t)$ and $x_a^2(t)$ in the state space $\mathcal{S} \times \mathcal{S}$. We therefore define the probability q in the following way:

$$\begin{aligned} q_{ij}(t, u^{1,t}, u^{2,t}) \quad &:= \quad p(x_a^1(t) = s_i, x_a^2(t) = s_j | u^{1,t}, u^{2,t}) \\ q(t, u^{1,t}, u^{2,t}) \quad &:= \quad \left(q_{ij}(t, u^{1,t}, u^{2,t}) \right)_{i,j \in \{0, \ldots, 2^m - 1\}} . \end{aligned}$$

Thus $q(t, u^{1,t}, u^{2,t})$ is the $2^m \times 2^m$ matrix whose entry $q_{i,j}$ is the joint probability of finding $x_a^1(t)$ in the state s_i and $x_a^2(t)$ in s_j after applying the input $u^{1,t}$ and $u^{2,t}$ respectively. Using q we can rewrite (2) as:

$$\begin{aligned} d(k) \quad &= \quad \left\langle \left| \sum_{i,j=0}^{2^m - 1} q_{ij}(0, u^1, u^2)(s_i - s_j) \right| \right\rangle_u \\ &= \quad \left\langle \left| 2^{1-m} \sum_{i,j=0}^{2^m - 1} q_{ij}(0, u^1, u^2)(i - j) \right| \right\rangle_u . \end{aligned}$$

We need to calculate $q_{ij}(t, u^{1,t}, u^{2,t})$ at time $t = 0$ in order to determine $d(k)$. This can be achieved iteratively via the mapping S representing the transition from time step t to $t+1$ by applying the input pair $u^1(t)$ and $u^2(t)$:

$$q(t+1, u^{1,t+1}, u^{2,t+1}) = S\left(q(t, u^{1,t}, u^{2,t}), u^1(t), u^2(t) \right) .$$

The k-fold composition of S with the inputs u^1 and u^2 is denoted by $S^{(k)}_{u^1,u^2}$:

$$\begin{aligned} q(t,u^{1,t},u^{2,t}) &= S^{(k)}_{u^1,u^2}\left(q(t-k,u^{1,t-k},u^{2,t-k})\right) \\ q(t,u^{1,t},u^{2,t}) &= \lim_{k\to\infty}\left(S^{(k)}_{u^1,u^2}\left(q(t-k,u^{1,t-k},u^{2,t-k})\right)\right). \end{aligned}$$

For the sake of fast numerical evaluation, we calculate $q(0,u^1,u^2)$ by starting at $t=-n$ with the initial condition q^*:

$$q(0,u^1,u^2) = S^{(n)}_{u^1,u^2}\left(q^*\right).$$

The initial condition $q^* := 2^{-m}\mathrm{id}_{2^m\times 2^m}$ (where $id_{n\times m}$ is the $n\times m$ identity matrix) was chosen because of the following relation:

$$q^* = \left\langle q(t,u^1,u^1)\right\rangle_{u^1}.$$

2.4 Separation approximation

Unfortunately the complexity to calculate q scales like $\mathcal{O}(2^{2m})$ (capital $\mathcal{O}$ notation) with the number of bits m. This basically renders the approach described above useless for $m>5$. However, this can be circumvented by generalizing the AA in the following way. Since the state $x^i_a(t)$ of neuron a is quantized with m bits, we can write it in the following way:

$$\begin{aligned} x^i_a(t) &= \sum_{l=0}^{m-1} 2^{-l}(b^i_l(t)-1/2), \quad b^i_l(t)\in\{0,1\} \qquad (3)\\ B_l(x^i_a(t)) &:= b^i_l(t). \end{aligned}$$

According to (3) there is a unique binary representation of $x^i_a(t)$ given by $(b^i_0(t),\ldots,b^i_{m-1}(t))$; the mapping $B_l(.)$ maps a state to its l^{th} bit. Equation (3) can be interpreted as effectively replacing unit a with m binary units of states $b^i_l(t)$ whose outputs are reduced by $1/2$ and multiplied by 2^{-l} and finally summed up; this is still exact. Now we assume that each of these m units receives input drawn independently from the input distribution and has different weights drawn independently from $N(0,\sigma^2)$ every time step; hence this approach generalizes the AA. Therefore, for given presynaptic input the $b^i_l(t)$ are independent for different i and l under this approximation. Thus:

$$\begin{aligned} q_{ij}(t,u^{1,t},u^{2,t}) &\approx \prod_{l=0}^{m-1} q^l_{B_l(s_i),B_l(s_j)}(t,u^{1,t},u^{2,t}) \\ q^l(t,u^{1,t},u^{2,t}) &= (q^l_{b^1_l,b^2_l}(t,u^{1,t},u^{2,t}))_{b^1_l,b^2_l\in\{0,1\}}. \end{aligned}$$

$q^l(t,u^{1,t},u^{2,t})$ is the 2×2 matrix, whose entry $q^l_{b^1_l,b^2_l}(t,u^{1,t},u^{2,t})$ is the joint probability of finding the bit number l of unit $x^1_a(t)$ in state $b^1_l\in\{0,1\}$ and of unit $x^2_a(t)$ in state $b^2_l\in\{0,1\}$. Under this approximation we only have to calculate $4m$ matrix entries instead of the 2^{2m} entries, which is a considerable reduction of complexity.
We denote the update mapping for q^l by S^l:

$$q^l(t+1,u^{1,t+1},u^{2,t+1}) = S^l(q(t,u^{1,t},u^{2,t}),u^1(t),u^2(t)).$$

An explicit form for S^l can be derived in the following way. First we condition the probability $q^l(t+1,u^{1,t+1},u^{2,t+1})$ on the presynaptic input $h^1=(h^1_1,\ldots,h^1_K)$ for network N1 and $h^2=(h^2_1,\ldots,h^2_K)$ for network N2 and on the weight matrix W for this time step (which is the same for N1 and N2). The pairs $(h^1_i,h^2_i),\ i\in 1,\ldots,K$ are iid. according to $q(t,u^{1,t},u^{2,t})$. This yields:

$$\begin{aligned} q^l(t+1,u^{1,t+1},u^{2,t+1}) &= \left\langle\left\langle q^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,w)\right\rangle_W\right\rangle_{h^1,h^2} \\ &= \sum_{h^1,h^2} p(h^1,h^2)\int dW\ p(W)q^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,W). \qquad (4) \end{aligned}$$

We used the following abbreviations :

$$\begin{aligned}
\left\langle X(h^1,h^2)\right\rangle_{h^1,h^2} &:= \sum_{h^1,h^2} p(h^1,h^2)X(h^1,h^2) \\
&= \sum_{h_1^1,h_1^2}\cdots\sum_{h_K^1,h_K^2}\left(\prod_{i=1}^{K} q_{h_i^1,h_i^2}(t,u^{1,t},u^{2,t})\right)X(h^1,h^2) \qquad (5)\\
\langle X(W)\rangle_W &:= \int dW\ p(W)X(W) = \int_{\mathbb{R}}\cdots\int_{\mathbb{R}} X(W)p(w_1,\ldots,w_K)\ dw_1\cdots dw_K.
\end{aligned}$$

Here $w_1,\ldots,w_K$ denote the K presynaptic weights. Conditioned on the input and the weights, the network realizations N1 and N2 are independent:

$$\begin{aligned}
q_{ij}^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,w) &= p(b_l^1(t+1)=i|h^1,w)p(b_l^2(t+1)=j|h^2,w)\\
&= \delta(B_l(f_m(w^Th^1+u^1(t))),i)\delta(B_l(f_m(w^Th^2+u^2(t))),j)\\
w^Th^i &:= \sum_{\alpha=1}^{K} w_\alpha h_\alpha^i.
\end{aligned}$$

$\delta(.,.)$ denotes the Kronecker-Delta of its two arguments. The K-fold integral over the weights appearing in (4) can be reduced to a single integral:

$$\int dW\ p(W)q_{ij}^l(t+1,u^{1,t+1},u^{2,t+1}|h^1,h^2,W) = F_{ij}^l(h^1,h^2,u^1(t),u^2(t))$$

$$\begin{aligned}
F_{ij}^l(h^1,h^2,u^1(t),u^2(t)) = \frac{1}{(8\pi\det(C)\Sigma_{22})^{1/2}}\sum_{\alpha=0}^{2^l-1}\int_{I_{\alpha,i}-u^1(t)} dv\ \exp\left(-\frac{v^2}{2}\left(\Sigma_{11}-\frac{\Sigma_{12}^2}{\Sigma_{22}}\right)\right)\times\\
\sum_{\beta=0}^{2^l-1}\left[\mathrm{erf}\left(\frac{(I_{\beta,j}^+-u^2(t))\Sigma_{22}+\Sigma_{21}v}{(2\Sigma_{22})^{1/2}}\right)-\ \mathrm{erf}\left(\frac{(I_{\beta,j}^--u^2(t))\Sigma_{22}+\Sigma_{21}v}{(2\Sigma_{22})^{1/2}}\right)\right]
\end{aligned}$$

Here we use the following definitions:

$$\begin{aligned}
C &= \sigma^2\begin{pmatrix}(h^1)^Th^1 & (h^1)^Th^2\\ (h^2)^Th^1 & (h^2)^Th^2\end{pmatrix}\\
\Sigma_{ij} &= \left(C^{-1}\right)_{ij}\\
\bigcup_{\alpha=0}^{2^l-1} I_{\alpha,i} &= \mathrm{supp}(\delta(B_l(f_m(.)),i))\\
I_{\alpha,i} &= [I_{\alpha,i}^-,I_{\alpha,i}^+]\\
I_{\alpha,i}^- &:= \tanh^{-1}\left(2^{-l+1}\alpha-1\right)\\
I_{\alpha,i}^+ &:= \tanh^{-1}\left(2^{-l+1}(\alpha+1)-1\right).
\end{aligned}$$

The support of the function $\delta(B_l(f_m(.)),i)$ is the union of 2^l disjoint intervals $I_{\alpha,i}$ with lower bound $I_{\alpha,i}^-$ and upper bound $I_{\alpha,i}^+$.
Using the expression F_{ij}^l the update can finally be written as:

$$q_{ij}^l(t+1,u^{1,t+1},u^{2,t+1}) = \left\langle F_{ij}^l(h^1,h^2,u^1(t),u^2(t))\right\rangle_{h^1,h^2}.$$

For the sake of computational tractability for larger m and K, we do not evaluate the $2K$ sums explicitly involved in the average over the presynaptic input $\langle.\rangle_{h^1,h^2}$. Instead we determine this expectation value by a finite number a samples from the joint distribution $p(h^1,h^2)$; this sampling is easily realizable since $p(h^1,h^2)$ is of the product form given in (5); sample size for all experiments was chosen to be 150.

References

[1] Anonymous Author(s). On Computational Power and the Order-Chaos Phase Transition in Reservoir Computing, 2008. submitted for publication.

[2] H. Jaeger. The "echo state" approach to analyzing and training recurrent neural networks. GMD Report 148, German National Research Center for Information Technology, 2001.

[3] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[4] B. Derrida and Pomeau Y. Random networks of automata: A simple annealed approximation. *Europhysics Letters*, 1(2):45–49, 1986.

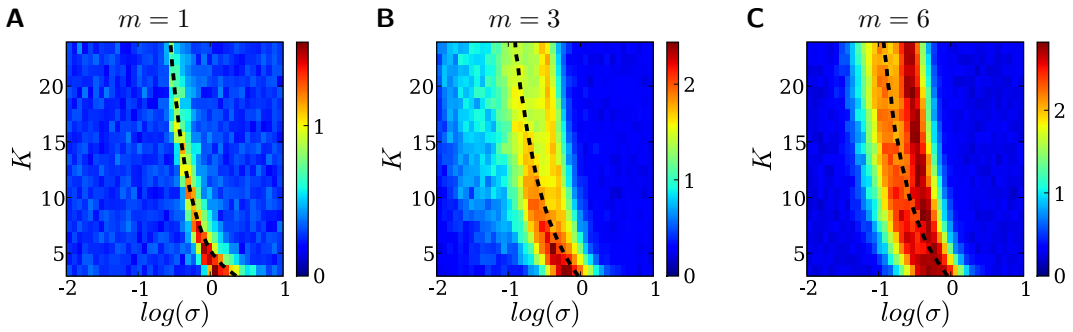

Figure 1: The performance $p_{\text{exp}}(C, \text{RAND}_5)$ for three different quantization levels $m = 1, 3, 6$, averaged over 20 randomly drawn functions of 5 bits, circuits C, initial conditions and input streams. $p_{\text{exp}}(C, \text{RAND}_5)$ is plotted as a function of the network in-degree K and the weight STD σ. The networks size is $N = 150$. The input time series have length 10000. The solid line represents the numerically found critical line.

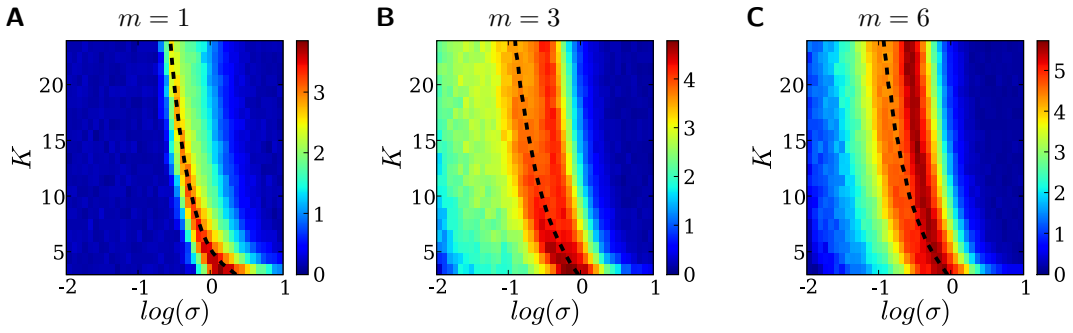

Figure 2: Same figure as Fig. 1 showing the performance $p_{\text{exp}}(C, \text{AND}_5)$ for the 5-bit AND-task averaged over 10 circuits C, initial conditions and input streams.