A Theoretical Framework for Understanding Computations in Cortical Microcircuits

Wolfgang Maass
Institut für Grundlagen der Informationsverarbeitung
Technische Universität Graz, Austria
Content:

1. What types of computations are carried out by the brain?

2. Cortical microcircuits are dynamical systems that consist of diverse components

3. A theoretical framework for analyzing online computing in realistic models for cortical microcircuits (or: can we replace the Turing Machine?)

4. Emergent computational properties of lamina-specific cortical microcircuit models

5. Conclusion, and thoughts about a research strategy for unraveling the structure of computations in the brain
1. What types of computations are carried out by the brain?

Models for neural circuits and neural systems are usually tested on offline computation tasks where all relevant input-components arrive simultaneously (e.g. PARITY, object recognition from static images):
In contrast to that, biological organisms need to process continuously arriving new pieces of information, and demands for outputs may arise at any time ("anytime computing", "real-time computing"): 

Hence from a mathematical point of view neural microcircuits compute filters (operators), i.e. they map input streams to output streams.

Note: it becomes then hard to distinguish computing from learning.
2. Cortical microcircuits are dynamical systems that consist of diverse components

Different types of neurons respond to the same input in different ways:
Different synapses respond to the same spike train in different ways:

one spike train, sent to two synapses

output amplitudes of synapse

output amplitudes of synapse
Thus: synapses are diverse dynamical systems that cannot be characterized by a single scalar (i.e., by the „weight“ of the synapse)

Model for synaptic dynamics from [Markram et al., PNAS 1998]:

The amplitude $A_k$ of the PSP for the $k^{th}$ spike in a spike train with interspike intervals $\Delta_1, \Delta_2, \ldots, \Delta_{k-1}$ is modeled by the equations

\[
A_k = w \cdot u_k \cdot R_k \\
u_k = U + u_{k-1} (1-U) \exp(- \Delta_{k-1}/F) \\
R_k = 1 + (R_{k-1} - u_{k-1} R_{k-1} -1) \exp(- \Delta_{k-1}/D)
\]

with hidden dynamic variables $u \in [0,1]$ and $R \in [0,1]$, whose initial values for the first spike are $u_1 = U$ and $R_1 = 1$. 
Typical values of the parameters $U$, $D$, $F$ for cortical synapses:

Empirically found distributions are reported in


Mean values of $U$, $D$, $F$:

<table>
<thead>
<tr>
<th>from</th>
<th>to</th>
<th>$E$</th>
<th>$I$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E$</td>
<td>$E$</td>
<td>0.5, 1.1, 0.05</td>
<td>0.05, 0.125, 1.2</td>
</tr>
<tr>
<td>$I$</td>
<td>$E$</td>
<td>0.25, 0.7, 0.02</td>
<td>0.32, 0.144, 0.06</td>
</tr>
</tbody>
</table>
3. A theoretical framework for analyzing online computing in realistic models for cortical microcircuits (or: can we replace the Turing Machine ?)

A circuit for online computing does not map static inputs to static outputs, but maps input streams to output streams. Hence it implements a filter (or operator).

Unfortunately the class of all filters is much too large. Can one say more about that particular subclass of filters that cortical microcircuits might be able to compute?

In engineering one prefers to consider only linear filters $F$. These are filters that can be defined (for the case of a single time-varying input $u(\cdot)$) by an integral

$$(Fu(\cdot))(t) = \int_{0}^{\infty} d\tau \ h(\tau) \ u(t - \tau)$$

with some arbitrary integrable function $h(\tau)$.

Question: Are all filters computed by cortical microcircuits linear?
Answer: certainly not.

An interesting and MUCH larger class of filters is the class of filters $F$ that can be defined by a Volterra series (written here for simplicity just for the case of a single time-varying input $u(\cdot)$):

$$(F u(\cdot))(t) = \alpha_1 \int_{0}^{\infty} d\tau_1 \ h_1(\tau_1) u(t-\tau_1)$$

$$+ \alpha_2 \int_{0}^{\infty} \int_{0}^{\infty} d\tau_1 d\tau_2 \ h_1(\tau_1,\tau_2) u(t-\tau_1) \cdot u(t-\tau_2)$$

$$+ \ldots$$
Any filter $F$ which is defined by such Volterra series has fading memory:

In order to determine the output $(Fu(\cdot))(t)$ with a given precision $\varepsilon$ it suffices to know the values of $u(t - \tau)$ up to some finite precision $\delta$ for all $\tau$ from some finite time interval $[0, T]$.

**Question**: Might cortical microcircuits be able to approximate any filter $F$ that can be defined by a Volterra series?
Answer: This is not as impossible as it first appears.

Consider very simple computational models ("liquid state machines") that only consist of a filterbank and a readout unit.

\[ y(t) = f(x(t)) \]

filter output ("liquid state")

memoryless readout

filterbank, consisting of basis filters \(B_1, \ldots B_k\)
**Theorem:** (based on [Boyd and Chua, 1985])

Any filter $F$ which is defined by a Volterra series **can be approximated** with any desired degree of precision by such very simple computational model

- **if** there is a rich enough pool $B$ of basis filters (time invariant, with fading memory) from which the basis filters $B_1, \ldots, B_k$ in the filterbank can be chosen ($B$ needs to have the pointwise separation property)
  
  and

- **if** there is a rich enough pool $R$ from which the readout functions $f$ can be chosen ($R$ needs to have the universal approximation property).

**Def:** A class $B$ of basis filters has the pointwise separation property if there exists for any two input functions $u(\cdot)$, $v(\cdot)$ with $u(s) \neq v(s)$ for some $s \leq t$ a basis filter $B \in B$ with $(Bu)(t) \neq (Bv)(t)$.

Examples for classes that have this property:

- $B = \{\text{delay lines}\}$
- $B = \{\text{linear filters}\}$
- $B = \{\text{models for biologically realistic dynamic synapses}\}$

**Question:** can we get more computational power if we allow feedback from readouts?
Relating this abstract theorem to neural microcircuits:

For a realistic physical system with noise, the abstract pointwise separation property is not quite adequate. A noise-robust variation would demand that \((Bu)(t)\) is significantly different from \((Bv)(t)\) if the input streams \(u(\cdot), v(\cdot)\) were significantly different in the past.

Computer models of cortical microcircuits tend to have this practically more relevant version of the pointwise separation property:
Do neural readouts have the universal approximation property?

The pool $\mathbf{R}$ of possible readout functions $f$ has the universal approximation property if any continuous function $h$ can be approximated (on any closed and bounded domain) with any desired degree of precision by some $f \in \mathbf{R}$.

If one believes that projection neurons (e.g. on layer 2/3 or layer 5/6) of a cortical microcircuit compute the readout function, then the resulting pool $\mathbf{R}$ does not have this property, since such neural readouts have trouble to approximate nonlinear functions.

Note: Linear functions do not have the universal approximation property.
But adaptive linear readouts suffice if they are supported by a suitable fixed nonlinear preprocessing (a "kernel" in the terminology of machine learning).

**Example:** If a circuit precomputes all products $x_i \cdot x_j$ of some input variables $x_1,...,x_n$, then a subsequent linear readout has the same expressive power as quadratic readouts in terms of the original input variables $x_1,...,x_n$.

More abstractly: A microcircuit has the functionality of an ideal kernel if it maps different input streams $u_1(\cdot),...,u_k(\cdot)$ onto circuit states $x_{u_1}(t),...,x_{u_k}(t)$ at a subsequent time point $t$ that are not only pairwise different, but in addition linearly independent.

**Remark:** A unique advantage of linear readouts: their learning cannot get stuck in local minima of the error function.
Hence the following functionalities of cortical microcircuits would suffice in order to approximate any nonlinear filter $F$ that can be defined by Volterra series:

$$u(s) \text{ for } s \leq t$$

$$B_1 \quad \cdot \quad \cdot \quad \cdot$$

$$B_k$$

nonlinear kernel

memoryless linear readout

$$y(t) = f(x(t))$$
These theoretical considerations suggest the following model for the computational function of a cortical microcircuit:

- **Input stream**
  - Optimized by unsupervised learning
  - Analog fading memory
    - Nonlinear projection into high-dimensional state space ("kernel")

- **Sufficient** to adjust these synapses during supervised learning

- **Diverse readouts**, trained for specific tasks, provide outputs at any time

- Feedbacks from readouts into the circuit can create partial attractors
How realistic is this theory?

Here are some results of computer simulations of generic neural microcircuits (without laminar structure)

neurons: leaky integrate-and-fire neurons, 20% of them inhibitory, neuron $a$ is synaptically connected to neuron $b$ with probability $C \cdot \exp(-D^2(a,b)/\lambda^2)$

synapses: dynamic synapses with fixed parameters $w$, $U$, $D$, $F$ chosen from distributions based on empirical data

input spike trains injected into 30% randomly chosen neurons, with fixed randomly chosen amplitudes
What exactly is the input for linear readouts in the case of a neural microcircuit model?

We assume that a linear readout receives as input a time-varying vector $\mathbf{x}(t)$, whose $i^{\text{th}}$ component $x_i(t)$ is a low-pass filtered version of the spike train from the $i^{\text{th}}$ presynaptic neuron. It outputs a time-varying value $\mathbf{w} \cdot \mathbf{x}(t)$.

**Note:** Wessberg et al. have shown in Nature 2000 that hand trajectories of primates can be predicted by such linear readouts from many neurons in several cortical areas.
What can neural microcircuits compute in this way?

Circuit input: 4 Poisson spike trains with firing rates \( f_1(t) \) for spike trains 1 and 2 and firing rates \( f_2(t) \) for spike trains 3 and 4, drawn independently every 30 ms from the interval [0, 80] Hz.

\[
\begin{align*}
f_1(t) & : \text{sum of rates of inputs 1&2 in the interval } [t-30\text{ ms}, t] \\
f_2(t) & : \text{sum of rates of inputs 3&4 in the interval } [t-30\text{ ms}, t] \\
f_3(t) & : \text{sum of rates of inputs 1-4 in the interval } [t-60\text{ ms}, t-30\text{ ms}] \\
f_4(t) & : \text{sum of rates of inputs 1-4 in the interval } [t-150\text{ ms}, t] \\
f_5(t) & : \text{spike coincidences of inputs 1&3 in the interval } [t-20\text{ ms}, t] \\
f_6(t) & : \text{nonlinear combination } f_6(t) = f_1(t) \cdot f_2(t) \\
f_7(t) & : \text{nonlinear combination } f_7(t) = 2f_1(t) - 4f_1^2(t) + \frac{3}{2}(f_2(t) - 0.3)^2
\end{align*}
\]
Testing our approach on a popular benchmark task: speech recognition

We consider the task considered by Hopfield and Brody in PNAS 2000 and 2001 (with the same transcription of speech into spike trains):

recognition of spoken words "zero", "one", ... "nine", each spoken 10 times by 5 different speakers, each spoken word encoded into 40 spike trains by Hopfield and Brody
Comparing the performance of generic diverse circuits and constructed circuits for this task:

- Linear readouts from a generic neural microcircuit model (consisting of 135 neurons) recognize after training spoken test-words as well as the ingenious circuit consisting of $> 6000$ I&F neurons constructed especially for this task by Hopfield and Brody.

- The generic neural microcircuit model can handle linear time warps in the input at least as well as the circuit constructed to achieve that (and it can also handle nonlinear time warps).

- The generic neural microcircuit model classifies the spoken word instantly when the word ends (i.e., in real-time), rather than 300 – 500 ms later.
linear readouts from the generic microcircuit model can even be trained to classify a word before it ends ("anytime computing"):

e.g. anytime recognition of "one"
Consider a readout neuron that is trained to fire whenever „one“ is currently spoken.

Thus:

linear readouts can form complex equivalence classes of circuit states \( x(t) \)

resulting values of \( w x(t) \)

Thus:

linear readouts can form complex equivalence classes of circuit states \( x(t) \)

\[
\begin{array}{cccc}
\text{class „one“} & \text{class „other“} \\
0.43 & 0.098 & 0.68 & 0.3 & 0.11 & -0.6 & -0.37 & -0.01 & -0.16 & -0.21
\end{array}
\]

Question: What is the „neural code“ of these circuits?
Recognition of spatio-temporal input patterns is also possible in the presence of additional periodic inputs:

It is shown in (Kaske, Maass, 2005) that neural readouts can still extract in this case information about preceding spike inputs.
Is a cortical microcircuit model the only dynamical system that can do speech recognition?

[Fernando and Sojakka, ECAL 2003] applied our approach to a bucket of water (which is also a dynamical system):

“We made a bucket of water, vibrated it with lego motors, filmed the waves with a webcam do speech recognition.”

states $\mathbf{x}(t)$
Question: What makes cortical microcircuits better suitable for online computing than a bucket of water?
Actually: Not much is known so far about that type of dynamical systems to which cortical microcircuits (and gene regulation networks) belong

<table>
<thead>
<tr>
<th></th>
<th>cellular automata</th>
<th>iterative maps</th>
<th>differential equations</th>
<th>threshold circuits</th>
<th>cortical microcircuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>analog?</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>continuous time?</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>high-dimensional?</td>
<td>yes</td>
<td>no</td>
<td>usually no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>with noise?</td>
<td>no</td>
<td>no</td>
<td>usually no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>with online input?</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>usually no (exception: recent paper)</td>
<td>yes</td>
</tr>
</tbody>
</table>
Two new theoretical results about online computing for that category of dynamical systems to which neural microcircuits belong:

1. The computational power of such dynamical systems can be predicted by estimating directly their kernel-capability and their generalization capability (VC-dimension). Both can be done by computing the rank of the ensemble of circuit states for suitable ensembles of inputs. The „edge-of-chaos“ is not a good predictor.

(Maass, Legenstein, Bertschinger, NIPS 2004): Methods for estimating the computational power and generalization capability of neural microcircuits)

2. The computational power of dynamical systems with fading memory increases drastically if one allows feedback from trained readouts

(Maass, Prashant, in preparation): Principles of real-time computing with partial attractors in generic microcircuit models
How much computational power does one gain by adding feedback?

Note: Getting feedback from a trained readout is equivalent to training a neuron within the circuit.

A simple mathematical result shows that by adding feedback even to the simplest type of fading memory circuits, one can simulate any dynamical system (defined by any high order nonlinear DE).
For neural microcircuit models such feedback adds the power of a (non-fading) finite state machine, even in the presence of noise:

It can create **partial attractors** that constrain some coordinates of the high-dimensional dynamical system, while other coordinates of the circuit state can execute non-trivial online computations, such as computing the product of 2 input firing rates. Partial attractors can add context-info to online computations.
4. Emergent computational properties of lamina-specific cortical microcircuit models

Figure 10.12
The cytoarchitecture of the striate cortex. The tissue has been Nissl stained to show cell bodies, which appear as dots. (Source: Adapted from Hubel, 1988; p. 97.)
Cortical microcircuits are highly structured dynamical systems
(不像那些由理论家研究的动态系统)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L2/3 – E</td>
<td></td>
<td>0.26</td>
<td>0.21</td>
<td></td>
<td>0.08</td>
<td></td>
<td>0.55</td>
</tr>
<tr>
<td>L2/3 – I</td>
<td></td>
<td>0.16</td>
<td>0.25</td>
<td></td>
<td></td>
<td></td>
<td>0.2 ?</td>
</tr>
<tr>
<td>L4 – E</td>
<td></td>
<td>0.28</td>
<td>0.10</td>
<td>0.17</td>
<td>0.19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4 – I</td>
<td></td>
<td>0.50</td>
<td>0.2 ?</td>
<td>0.10</td>
<td>0.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5 – E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.09</td>
<td>0.10</td>
</tr>
<tr>
<td>L5 – I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.12</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Table 1: Connection probabilities within a column of about 100 μm diameter.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L2/3 – E</td>
<td>0.26</td>
<td>0.21</td>
<td>0.08</td>
<td>0.55</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2/3 – I</td>
<td>0.16</td>
<td>0.25</td>
<td></td>
<td>0.2</td>
<td>?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4 – E</td>
<td>0.28</td>
<td>0.10</td>
<td>0.17</td>
<td>0.19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L4 – I</td>
<td>0.50</td>
<td>0.2</td>
<td>0.10</td>
<td>0.50</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5 – E</td>
<td></td>
<td></td>
<td>0.09</td>
<td>0.10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L5 – I</td>
<td></td>
<td></td>
<td>0.12</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Connection strengths (mean amplitude of PSP measured at soma, in mV) between 6 populations of cortical neurons. Source of data same as for Table 1.
There are also specific lamina where inputs arrive, and from where information is projected to other circuits:

- **Input stream 2** → input stream 1 (e.g. from thalamus) → layer 2/3 → layer 4 → layer 5 → to lower brain areas

- To lateral and higher cortical areas
We have made our computer model also in other aspects more realistic:

• We use Hodgkin-Huxley neurons instead of I&F neurons.

• We provide to each neuron noisy background currents to match data from intracellular recordings by Destexhe and others.

• We take into account that a neural readout (e.g. on layer 5) does not get input from all neurons in the circuit. Rather we use for each readout a much smaller data-based set of presynaptic neurons.

• We take into account that a neural readout does not have the same flexibility as a linear readout, since the sign of each „weight“ is already determined by the type of the presynaptic neuron.
Results:

Data-based layered circuits have significantly better computational properties than randomly connected control circuits with the same number of neurons and synapses.

\[ \text{tcl} = \text{classification of spatio-temporal patterns of length 30 ms, index 1 (2) indicates input primarily into layer 4 (2/3), -delta-t refers to preceding (overwritten) input of length 30 ms} \]

\[ \text{XOR} = \text{XOR of indices of patterns in both input streams 1 and 2} \]
Why has the data-based circuit with laminar structure more computational power?

Our hypothesis: The data-based laminar circuit is less chaotic (but has both high sensitivity to the current input and temporal integration capability).
5. Conclusions,

and thoughts about a research strategy for unraveling the structure of computations in the brain
My hypothesis:
Cortical microcircuits are designed by evolution to have particularly good temporal integration- and kernel-capability. In addition they are designed for presenting information to neural readouts in such a way that readouts can learn from few examples.
Arguments that support this hypothesis:

1. We have on this basis been able to carry out demanding computations (for computational tasks that are likely to occur in nature) on data-based cortical microcircuit models.

2. It turns out that for speech recognition such data-based neural microcircuit models have better computational properties than artificial circuits that were designed for this task.

3. One can demonstrate on this basis that the laminar structure of cortical microcircuits entails specific computational advantages.

4. Is there any alternative to this approach? (provided one takes into account that biologically realistic computations map input- to output streams)
Weak points of this hypothesis:

Can neural readouts also be trained with much fewer training examples?

Where does the supervision come from that is needed to train neural readouts?

Possible solutions:

1. If one learns predictions, then nature itself provides the supervision (just a little bit later)

2. Learning of associations does not require supervision (just learning-control)

3. Genetically encoded developmental learning phases focus on specific adaptive control problems (e.g. hand-eye coordination) where a suitable supervision can be derived from external feedback signals. Extensive prior developmental learning may provide the basis for subsequent specific learning from few training examples.

4. Reinforcement learning of a higher level system produces learning targets for lower level supervised learning (e.g., for learning the value function or the Q-function).
... and resulting thoughts about a research strategy for unraveling the structure of computations in the brain

• We are lacking insight into the initiation and control of multiple parallel computational processes in the brain.

• We are lacking insight into the organization of learning processes in the brain (i.e., the organization of plasticity that occurs simultaneously on many spatial and temporal scales).

• The structure and organization of these multi-level learning processes is the only place where we have a chance to find unifying principles (rather than just more and more diversity and exceptions to rules).

A simple thought-experiment: Assume that the brain would be just a feedforward neural net that is trained by backprop.
(Note: this is a silly assumption !!!)
Until we discover that this network is structured via backprop, we would just find lots of more-or-less accidental data about „neural codes“ of neurons on the first or the last hidden layers. But this type of data will never allow us to understand the true structure of the system.
Acknowledgements

I have reported results of recent joint work with Alexander Kaske,

Nils Bertschinger  
Prashant Joshi  
Stefan Häusler  
Robert Legenstein

It was based on previous joint work with Henry Markram and Thomas Natschläger.