On large-scale neural simulations and applications in neuroinformatics

SIMON BENJAMINSSON

Akademisk avhandling som med tillstånd av Kungliga Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen måndagen den 3 juni 2013 kl 13.00 i sal F3, Lindstedtsvägen 26 Kungliga Tekniska högskolan, Stockholm.

TRITA-CSC-A 2013:06
ISSN-1653-5723
ISRN-KTH/CSC/A--13/06-SE
© Simon Benjaminsson, maj 2013
Abstract

This thesis consists of three parts related to the in silico study of the brain: technologies for large-scale neural simulations, neural algorithms and models and applications in large-scale data analysis in neuroinformatics. All parts rely on the use of supercomputers.

A large-scale neural simulator is developed where techniques are explored for the simulation, analysis and visualization of neural systems on a high biological abstraction level. The performance of the simulator is investigated on some of the largest supercomputers available.

Neural algorithms and models on a high biological abstraction level are presented and simulated. Firstly, an algorithm for structural plasticity is suggested which can set up connectivity and response properties of neural units from the statistics of the incoming sensory data. This can be used to construct biologically inspired hierarchical sensory pathways. Secondly, a model of the mammalian olfactory system is presented where we suggest a mechanism for mixture segmentation based on adaptation in the olfactory cortex. Thirdly, a hierarchical model is presented which uses top-down activity to shape sensory representations and which can encode temporal history in the spatial representations of populations.

Brain-inspired algorithms and methods are applied to two neuroinformatics applications involving large-scale data analysis. In the first application, we present a way to extract resting-state networks from functional magnetic resonance imaging (fMRI) resting-state data where the final extraction step is computationally inexpensive, allowing for rapid exploration of the statistics in large datasets and their visualization on different spatial scales. In the second application, a method to estimate the radioactivity level in arterial plasma from segmented blood vessels from positron emission tomography (PET) images is presented. The method outperforms previously reported methods to a degree where it can partly remove the need for invasive arterial cannulation and continuous sampling of arterial blood during PET imaging.

In conclusion, this thesis provides insights into technologies for the simulation of large-scale neural models on supercomputers, their use to study mechanisms for the formation of neural representations and functions in hierarchical sensory pathways using models on a high biological abstraction level and the use of large-scale, fine-grained data analysis in neuroinformatics applications.
Acknowledgement

First and foremost I would like to thank my supervisor Anders Lansner for the opportunity to pursue this field of research and for initiating the collaborations resulting in the papers of this thesis. I also want to thank my co-supervisor Örjan Ekeberg and all other senior people and administrators at CB for the support and good work you do by building and maintaining an organization which affects so many people.

I have collaborated with several people on the papers which make up this thesis and who I would like to thank. At KTH these include in particular Pawel Herman and Mikael Lundqvist. At KI, with the support from Stockholm Brain Institute, these include in particular Peter Fransson and Martin Schain. I also collaborated with many other people on the different projects, I am very grateful to all of you.

I give A+ to all the past and present members of CB including Pierre, Bernhard, Pradeep, David, Cristina, Johannes, Joel, Marcus, Jenny, Nalin, Henrik, Mikael, Erik, Malin, Phil and everyone else. All of the friendly and talented people making up the Stockholm Brain Institute junior researchers’ club, the best brain junior researchers’ club which has its own ski conference! deserves a special mention.

Also a note to all friends and family in the real world; I once again promise you that we are not trying to build Skynet. I hope that the fruits of this research field’s collective labor will benefit you all sooner rather than later.
# Contents

1. Introduction .................................................................................................................. 1  
   1.1 List of papers included in the thesis ................................................................. 3  
   1.2 Software developed and used in thesis work ....................................................... 3  
   1.3 List of papers not included in the thesis ............................................................. 4  
   1.4 Contributions per paper ...................................................................................... 4  
2. Large-scale neural simulations .................................................................................... 7  
   2.1 Levels of model complexity .................................................................................. 8  
   2.2 Paper I: The Nexa simulator .............................................................................. 10  
      2.2.1 Motivation and design ................................................................................ 10  
      2.2.2 Results ........................................................................................................ 12  
   2.3 Discussion ............................................................................................................ 17  
3. Algorithms and models ............................................................................................... 19  
   3.1 Paper II: Activity-dependent connectivity ........................................................... 20  
      3.1.1 Motivation .................................................................................................. 20  
      3.1.2 Model ......................................................................................................... 20  
      3.1.3 Results ....................................................................................................... 22  
      3.1.4 Discussion .................................................................................................. 22  
   3.2 Paper III: Large-scale modeling of the olfactory system ..................................... 24  
      3.2.1 Biological background: The vertebrate olfactory system ........................... 24  
      3.2.2 Motivation .................................................................................................. 26  
      3.2.3 Model ......................................................................................................... 26  
      3.2.4 Results ....................................................................................................... 28  
      3.2.5 Application: Machine olfaction (Paper X) ............................................... 31  
      3.2.6 Discussion .................................................................................................. 33  
   3.3 Paper IV: A model of categorization, learning of invariant representations and  
       sequence prediction utilizing top-down activity .................................................. 34  
      3.3.1 Motivation .................................................................................................. 34  
      3.3.2 Model ......................................................................................................... 35  
      3.3.3 Results ....................................................................................................... 35
3.3.4. Discussion.............................................................................................................. 39

4. Large-scale data analysis of neuroimaging data......................................................... 41
   4.1. fMRI.................................................................................................................... 42
   4.2. PET..................................................................................................................... 43
   4.3. Graph methods.................................................................................................... 43
   4.4. Paper V: fMRI resting-state networks ............................................................... 44
       4.4.1. Results ........................................................................................................ 45
   4.5. Paper VI: Arterial input function from PET image voxels ................................. 48
       4.5.1. Results ...................................................................................................... 51
   4.6. On the use of in-situ visualization in large-scale data analysis ........................... 53
   4.7. Discussion .......................................................................................................... 55

5. Conclusions and outlook............................................................................................ 57
Not only does the study of the human brain give us a chance to catch sight of ourselves, it is also an endeavor with promises of major positive societal impacts. If we consider brain-related diseases, dementia alone is estimated to currently cost society $604 billion a year, which is more than 1% of the global GDP (Batsch and Mittelman, 2012). Moreover, these costs are expected to rise as elderly populations increase. Improvements in the diagnosis, therapies or solutions to brain-related diseases would lower suffering, healthcare costs and increase the lifespan and value of life for a large proportion of the world’s population. The brain is also the only truly intelligent systems we can study and try to replicate in order to understand the computational principles of intelligence and human-like reasoning. If this would be successful and lead to intelligent robots, it would lead to major societal and ethical consequences and, as predicted by economists, lead to an arbitrary GDP increase in the world as the cost of work would be reduced to zero (Krugman, 2012). Computers currently perform human-type tasks such as language translation by calculating and comparing the statistics in large data sets. While successful in specific tasks, they are still far from having the flexibility and generalizability of human-level intelligence. For instance, they lack humans’ capability to use scarce and unstructured data sources to incrementally build knowledge as seen during e.g. language acquisition (Tenenbaum et al., 2011). On the hardware side, new computing architectures could become far more resilient to noise and faults than anything currently available by replicating the solutions of the brain (International Technology Roadmap for Semiconductors: www.itrs.net, BrainScales: www.brainscales.eu).

This thesis is an in silico study of the brain, i.e. where computers and computer simulations are used. The brain can be viewed as a massively parallel compute structure with computation co-localized with memory. Structurally, this is similar to the most powerful computers of today making these supercomputers well suited for neural simulations of also very large sizes. With the current trend of increasingly larger datasets in most fields of science and technology, novel methods designed for neural simulations
and brain-inspired algorithms may also be suitable in the handling and analysis of massive datasets.

The thesis explores methods for large-scale neural simulations, large-scale models and use the developed methods and models in applications. These applications are predominantly in the field of neuroinformatics (a field at the intersection of neuroscience and IT), where we develop methods for fine-grained analysis of large-scale neuroimaging data. With regards to the included papers, what will be presented can be summarized into three areas of study: platforms, algorithms and models and applications.

**Platforms**

We develop a large-scale simulator and investigate techniques for the simulation of neural systems on a high level of biological abstraction and of sizes on the order of small mammalian brains when run on massively parallel computers (Paper I and software package VII). A software library for use in the data analysis of large-scale neuroimaging data (package VIII) is also developed.

**Algorithms and models**

We simulate a full-scale model of the mammalian early olfactory system using rate-based neural units and investigate its properties in classification and segmentation tasks (Paper III). An algorithm for building activity-dependent connectivity in hierarchical models (Paper II) is proposed and we suggest how top-down activity could aid the learning of invariant representations in sensory hierarchies and aid in sequence prediction (Paper IV).

**Applications**

We use methods inspired by and extended from the developed models to discover structure in data and their parallelized implementations, and use them in large-scale analysis of brain imaging data (package VIII). Their applicability are confirmed using fMRI resting-state data as it discovers resting-state networks thought to reflect the underlying connectivity of the human brain (Paper V). We benefit from the parallelization and use of supercomputers in a clinical application using PET data, where we show that our proposed analysis method can be used to partly remove the need of invasive arterial blood sampling for patients during PET sessions to a higher extent than in any previously proposed method (Paper VI).

This thesis is organized as follows: Chapter 2 describes large-scale neural simulation and the technologies developed in the thesis. Chapter 3 presents the modeling papers and related work and Chapter 4 describes the applications in neuroinformatics. Chapter 5 concludes the thesis work and presents possible future directions.
1.1 List of papers included in the thesis


* Denotes co-first author

1.2 Software developed and used in thesis work

(VII) Nexa: Large-scale neural simulation software. Described in (I). Available from repository at github.com/simonbe/nexa, LGPL license. Used in (I), (III), (IV), (IX), (X) and (XII)-(XIV).

(VIII) Large-scale imaging data analysis library. Available upon request. Used in (V) and (VI).
1.3 List of papers not included in the thesis

The thesis is partly based on the following publications which are not included:

Book chapters


Conference proceedings


Technical Reports


1.4 Contributions per paper

Paper I: I conceived the simulator architecture, developed all core code, did all experiments and wrote the paper. Bindings to VisIt were developed together with Vladimir Slavnić and Marko Spasojević (paper XIII) and bindings to MUSIC together with Erik Rehn (paper XIV).

Paper II: I wrote most of the Methods part of the paper, co-developed code and made figure 2.3.
Paper III: I modeled the cortical parts of the full model, co-designed the experiments, generated all results in the Results section and co-wrote the paper.

Paper IV: I co-conceived the ideas and design of experiments, did all experiments and co-wrote the paper.

Paper V: I co-conceived the design of the analysis experiments, did all analysis experiments and co-wrote the paper.

Paper VI: Together with Martin Schain, I co-conceived our method in the paper (the PWC method), evaluated performance and testing of all early and the final version of the PWC method. I did all coding with regard to the PWC method, wrote the PWC parts in the Methods section, parts of the Discussion and generated Table 2.
Large-scale neural simulations

Computational studies of the brain use simulations as the principal tool, where the biological system is simulated on a specific scale. For example, a low-level scale in the neural context can be ion channels and propagating electrical pulses. On a higher level, populations of neurons and static synaptic connections can be simulated. In order to perform a simulation, a model needs to be developed, describing the neural system at the scale in question. If such a description results in many state variables, it is referred to as a large-scale simulation. One example is the simulation of networks with large quantities of neurons and synapses.

Here we aim at efficient large-scale simulation of brain-like network architectures of size that approaches that of real brains, i.e. many millions of units sparsely connected via billions of connections. To be able to simulate such models on current compute architectures, one need to use distributed computers with enough compute cores to do it efficiently and enough memory to hold the network parameters.

Why are large-scale simulations important? If we want to compare our simulated dynamics with the dynamics of real biological systems, large-scale studies are required. If we want to use our models for applications we would also often need large-scale models to be able to store enough information to achieve sufficient performance. For example, in Section 3.2 (Paper III) we see how the performance of a mixture segmentation task will saturate if the model is not large enough. Also in machine learning the importance of large models has been pointed out: For instance, in unsupervised learning, factors such as the number of hidden nodes in a model often turn out to be more important than the learning algorithm used to achieve high performance (Coates et al., 2011; Coates, 2012).

However, actual simulation is only one part of the simulation workflow. Handling input stimuli, simulation outputs, parameters, analysis and visualization of simulations are important to achieve high user productivity. Complexity in data handling and interpretation of simulation results increases for large-scale simulations as the amounts of
output data increases. One way to handle this increase of complexity is to make it easy for the modeler to add analysis code into the simulator. Alternatively, one could integrate existing tools for visualization and analysis of large data-sets on parallel machines (e.g. VisIt: visit.llnl.gov or ParaView: paraview.org). The simulator presented in this chapter (Section 2.2 and Paper I), Nexa, demonstrates examples of both strategies.

2.1. Levels of model complexity

One way to classify modeling work in neuroscience is the complexity of the model (Gerstner et al., 2012): Biologically detailed modeling work typically uses complex models with many parameters while more abstract modeling uses less. While analytical solutions of biological systems provided by theory can provide a full understanding of a system they are restricted to simple models. Simulations can be used to study a much wider class of models. However, it may be difficult to fully explore the parameter space of a model and the biological parameters may only be partially known, restricting the conclusions one can draw from the results of a simulation.

Ranging from biologically detailed to more abstract, models are often divided into the Hodgkin-Huxley, integrate-and-fire, rate-based and mean-field regimes:

In the **Hodgkin-Huxley** (1952) type of models, the mechanism for generating and propagating an action potential is modeled. Often individual neurons are modeled by several compartments. Such detailed biophysical models can be compared to intracellular electrophysiological recordings.

In **integrate-and-fire** models, the spiking behavior and not the spiking generating mechanism is modeled. This results in simpler equations which are faster to simulate than Hodgkin-Huxley type of models but still reproduce firing patterns from experiments (Brette and Gerstner, 2005). Individual neurons are typically modeled as point neurons.

In **rate-based** models the spiking is not considered to be of importance, but rather the firing rate of a neuron or a population of neurons. A unit representing a population of neurons may in a simulation be communicating a real-value representing the firing rate, instead of a binary spike. The simulator presented in this chapter is mainly focused on neural simulations at this level.

In **mean-field** models, methods from physics such as the Fokker-Planck equation are used to describe the macroscopic activity arising from the interaction of populations. This can then be used to e.g. analytically study parameter spaces (Deco et al., 2008).
Parallelization

Since the end of frequency scaling in CPUs, where speedup in computation was achieved through increase in the processor’s frequency, processor performance gains have been achieved through parallel computing, i.e. by increasing the number of compute cores. The fastest available computers are called supercomputers and are constructed by a massive number of multi-core processors connected through an interconnect. Some supercomputers of today combine the processors with General-purpose graphics processing units (GPGPUs) forming a heterogeneous compute architecture.

To use parallel computers, it is necessary to write parallel computer programs utilizing the parallel architecture. These are often more complex to write than sequential programs as dependencies can arise between instructions carried out by different processes. The biggest obstacles to good parallel performance are often synchronization (a “handshake” between processes, where processes wait until all have reached a certain point in the program) and communication between processes.

For parallel implementations of neural simulations, synchronization is handled by either synchronous or asynchronous updating. For communication on supercomputers, the standard model is the Message Passing interface (MPI) (Gropp et al., 1999). Parallel performance is often evaluated in terms of weak and strong scaling. These concepts are explained below.

Synchronous and asynchronous updating

Synchronous updating means that updates are taking place every time step in a synchronous, or clock-driven, fashion. That is, a simulation progresses with the same time-step on each process and the states of all network objects such as neural units and connections are updated once. The processes are synchronized each time step. Alternatively, a simulation can use asynchronous, or event-driven, updating. In that case neurons are updated as they receive an incoming event. In practice, asynchronous and synchronous updating will lead to different communication schemes. Asynchronous updating has the potential of being faster and scale better, as units are only updated as information is received. It is however harder to implement (both communication schemes and such things as I/O and visualization) on current parallel machines and the update barriers necessary for synchronous update does not necessarily need to be a bottleneck (Morrison et al., 2007). Most widely used large-scale simulators use synchronous updating (e.g. NEST: Gewaltig and Diesmann 2007, PCSim: Pecevski et al., 2009, NEURON: Carnevale and Hines, 2006, C2: Ananthanarayanan et al. 2009) with some exceptions (e.g. SPLIT: Hammarlund and Ekeberg, 1998).
Message passing interface

The dominant model for parallel scientific programs on current supercomputers is by utilizing the message-passing system MPI. This communication protocol supports both point-to-point (between individual processes) and collective (across all or a subset of processes) communication. The way MPI allows for collective communication for a subset of processes is by the use of *Communicators*. An algorithm using collective communication can then run on an entire machine or a subset of processes by switching the communicator it uses when calling communication routines. In the Nexa simulator (Paper I), we allow different parts of a network to have their own communicators. An algorithm using collective communication, e.g. for analyzing activity, can then be added (e.g. in the model description) to a network subpart.

Scaling performance

Performance measurements of parallel code are typically calculated using weak and strong scaling tests. In a weak scaling test, the size of the problem (such as the size of the simulated network) is incremented by the same factor, e.g. doubled when the number of processes is doubled. Here, one wants to see a constant run time for linear scalability. In a strong scaling test, the same problem size (keeping the size of the simulate network the same) is run for different number of processes. Here, a speedup with linear characteristics as the number of processes is increased is desirable, and would verify that the parallelized algorithm is efficient.

2.2. Paper I: The Nexa simulator

Nexa is an experimental scalable neural simulator for parallel simulation of large-scale neural network models at a high level of biological abstraction. It also contains methods for online analysis, real-time visualization and real-world sensor input.

2.2.1. Motivation and design

The motivation for designing and implementing Nexa was to

1. Construct a large-scale simulator aimed at the construction of networks handling (predominantly) rate-based units and allowing for easy integration of parallel algorithms e.g. from machine learning. Examples include algorithms for structural plasticity, i.e. the construction of network connectivity based on activity, without explicitly modeling how this takes place in a neurally detailed way. This high level of biological abstraction has not been the focus of other large-scale simulators.
2. Explore and test methods for combining neural simulation software with large-scale visualization and integration with real-world sensors.
3. Create a suitable design for the simulation of abstract large-scale neural network models, e.g. of the olfactory system (Chapter 3.2 and Paper III) and machine olfaction applications (Chapter 3.2.5 and Paper X).

The basic construction is an object-oriented and domain-targeted design: E.g. basic objects exist for a Unit and Connection. These are then extended to specific neuron models (such as a RateUnit) and a synapse model, by default containing a weight and optionally a delay. Various Modifiers can be added on top of populations of units or projections to change the state variables in a model during a simulated time step. E.g. synaptic plasticity can be used by adding a BCPNN modifier to a projection and will then be called and able to change the synaptic weights once each time step. The neural units and their corresponding connections in a network are by default distributed evenly across the processes of a machine.

However, Nexa also contains more abstract objects such as Population and a corresponding PopulationModifier. These are defined over a subset of a network, or a population of units, and can access and update state variables for all the units belonging to the subset. In this way, an operation such as winner-take-all (WTA) can be used for the activity values of a population of units without explicitly constructing the neural circuit which performs the operation.

If an operation such as WTA is performed on a particular population of a network and the neural units in the population are residing on different processes we run into a problem – a defined function that is supposed to calculate the winning unit cannot do that since it only has access to information about the neural units on its own process. We solve this by allowing subsets of the network to have their own MPI Communicators and communicating state information between the processes belonging to a Communicator group on an as-needed basis. For a user implementing a WTA function, it will look like there is access to all the activity values for the neural units of a population while the communication of these values is performed if needed.

A strategy for using Communicators and division of a network into subsets which can perform collective communication between the processes of the subset independent of other parts can also be used by other parts of a simulation workflow. Plug-ins such as VisIt (visit.llnl.gov) can interact with only part of the machine without interfering with other parts and does not need to be optimized or designed for the sizes of the full machine but only the size of the subpart.

Our experiments and reasoning followed the scenario that for upcoming generations of supercomputers it can be assumed that “FLOPs are cheap and moving data is expensive” (Dongarra et al. 2011). Handling of the simulation output by analysis and visualization should then take place where and when the simulation is running.
2.2.2. Results

To show the performance of the developed code, scaling experiments were performed on a variety of supercomputers. The use of in-situ visualization and online analysis was demonstrated and its suggested usefulness was described. Also using input from the outside world was demonstrated (in separate Paper XIV) leading to a capability of real-time input, simulation, analysis and output visualization.

2.2.2.1. Scaling performance

Strong scaling measurements were done on a Blue Gene/P system (JUGENE at the Jülich Supercomputing Centre) with 4-core 850 Mhz PPC processors sharing 2 GByte of memory. The system consisted of multi-core CPUs only. The model used for the measurements, illustrated in Figure 1, consisted of a randomly sparsely connected recurrent network divided into columns of 150 rate-based units with a winner-take-all mechanism in each column forcing a sparse representation (0.7% network activity). A Hebbian-based plasticity model BCPNN (Lansner and Holst 1996) was active during a training phase comprising a few random patterns. During a test phase, parts of the training patterns were stimulated and an analysis class was used during runtime to check in parallel if the retrieved patterns were the same as the ones in the training set. This analysis calculated the Hamming distances between global network activity and the trained patterns. The largest network consisted of $6.6 \cdot 10^7$ units and $3.9 \cdot 10^{11}$ connections.

The left panel of Figure 2 shows the strong scaling results (size of the network kept the same while varying the number of processes) for 65,536 processes up to 262,144 processes. A close to linear scaling was seen, with ~10% performance drop for a 4x increase of the number of processes.

The right panel of Figure 2 shows a breakdown of the total simulation time into percentage of time spent for the simulation of neural units, the MPI communication, the network initialization (buildup of connections), the plasticity model, the winner-take-all population operation in each column and the analysis of network activity. The slight increase in the fraction of the simulation spent performing MPI communication was responsible for the deviation from linear scaling. A slightly higher decrease in time spent in the plasticity than one would expect was also found. This was due to positive cache-effects when scaling up large-scale simulations, as has been noted elsewhere (Morrison et al. 2005), as a substantial time was spent accessing the synaptic weights from memory. In another model which was performance tested in Paper I, ~60% of the neural units simulation time was spent accessing synaptic weights for a single-process run.

Good scaling of the analysis at these large network sizes is important as stored activity data may get so large that post-processing on a local machine would be difficult in practice and the storing itself might become impractical.
Scaling experiments for two other models were also run, see Paper I for details. The major factor deciding the parallel performance was for all models the amount of communication performed.

**Figure 1.** Model used for strong scaling experiments on a Blue Gene/P.

**Figure 2.** Strong scaling with breakdown on a Blue Gene/P (JUGENE) up to 262144 processes (64 out of the total 72 racks of the machine). The model consisted of $6.6 \cdot 10^7$ neural units setup in a columnar structure and $3.9 \cdot 10^{11}$ recurrent synaptic connections. In a training phase 10 distributed patterns were stored. These were then retrieved during a testing phase by exposing the network to parts of the patterns, i.e. by pattern completion. The model relied on online analysis of the global network activity for interpretation of the dynamics as the storing and post-processing of network activity was practically difficult for these large network sizes. Most time was spent changing the weights of the synaptic connections during the training phase. MPI communication was responsible for a slightly larger fraction of the time spent as the number of processes was increased.
2.2.2.2. Online analysis and in-situ visualization

The typical workflow for a user is to do analysis and visualization off-site, most conveniently on a local machine (Figure 3a). This is often cumbersome or impossible for large-scale simulations as the output gets too large to analyze locally or to transfer across a network regularly. One alternative would be to have analysis and visualization tools independently running in the same network or on the same supercomputer (Figure 3b). Libraries adapted for neural simulations where such integration is simplified have recently been developed: MUSIC (MUSIC: Djurfeldt et al. 2010) is a protocol designed to handle communication between different simulators, making it possible to combine different models. We also use this in the Nexa simulator (see next Section and Paper XIV). However, to avoid moving data, we want to do as much analysis and visualization as possible where the data is located, i.e. in-situ, which is the type of analysis and visualization suggested in Paper I (Figure 3c).

Furthermore, the use of multiple communicators helps in the integration of the simulation. A developed analysis method can be assigned to a network subpart. That is, the communicator and surrounding methods take care of communication of network information while a user can develop an analysis method without considering how and where a network will be simulated. The analysis method could then be used by adding it to the subpart of the network of interest in the network model description. Figure 4a shows an example of an implemented online analysis performed during a simulation and visualized afterwards.

For real-time visualization, bindings for the parallel library libsim for the visualization software VisIt were developed (Paper XIII). This allows a simulation to be connected and providing input to a locally running instance of the VisIt software from which a simulation can be remotely controlled and certain simulation outputs can be visualized in real-time. This also runs in tandem, or in-situ, with the simulator on the same processes which performs the simulation. With the libsim library containing rendering capabilities, the visualization itself can be parallelized. This is one way to remove the need to store simulation data for later inspection. It could potentially also be used as a way to systematically synthesize macroscopic measures such as voltage sensitive dye imaging signals from a simulation, which could be compared to biologically measured values (Paper XIII). Figure 4b shows an example of a real-time visualization run during a simulation.
Figure 3. **a)** Standard user pattern where the analysis and visualization is separated from the simulation on a supercomputer. **b)** Integration of tools running separately on the same hardware or in the same network. **c)** The integration investigated here. The tools run on the same nodes as the simulations in order to minimize communication.

Figure 4. **a)** Recurrence plot visualizing the dynamics of the entire network in an instance of a model tested in Paper I (model 1) with 4.6 million neural units upon input of a stimulus. Full network activity is compared by a Euclidean measure to all previous simulation time steps. The plot was generated in parallel during simulation runtime, which removes the need to save network activity to disk in order to generate the plot. **b)** Snapshot of the parallel visualization environment VisIt (visit.llnl.gov) during the simulation of a small instance of the model in Figure 1. Here the user has selected to visualize firing rate activities (binary in this instance) with a contour plot and to show the activity values.
2.2.2.3. Closing the loop: Real-time input from the outside world

With online analysis and real-time visualization, a large-scale simulation could be monitored and steered by a user. As suggested in Paper XIII, e.g. artificial imaging signals could be generated directly from a simulation as we peek into different parts of a large-scale simulation.

One could imagine a simulation seeing the same type of input as an animal during an experimental study and compare their response as it happens. Such instant feedback could both help to tune models and explore novel experimental setups. Also for robotic applications with a perception-action loop, such real-world input would be a critical component.

To be able to handle these types of setups, we combined Nexa with the MUSIC framework (Djurfeldt et al., 2010; Paper XIV). Instead of connecting Nexa to another simulator, we connected it to a silicon retina image sensor (Lichtsteiner et al., 2007) which then sent real-time spikes to an input layer of a model. The silicon retina sensor is capable of event-based communication, i.e. similar to the Ganglion cells in the visual system each pixel transmits spiking (binary) information as a contrast-change takes place.

Figure 5 shows a simulation of a network consisting of a population with recurrent connectivity which has been trained on various geometrical figures. The silicon retina was connected to a local machine and its output (event-spikes) from an input of drawn patterns on a paper was sent by a lightweight MUSIC server over a network connection to an instance of Nexa running the trained model on a supercomputer (Cray XE6 Lindgren at KTH). The model performed a pattern completion, where the noisy input was completed into one of the trained geometrical figures. Any output or activity from this network could have been sent back to the MUSIC server to complete a perception-action loop or be visualized or analyzed in real-time.
Figure 5. Demonstration of pattern completion with input from an event-based silicon retina (Lichtsteiner et al., 2007), sent through a MUSIC server (Djurfeldt et al., 2010) to the Nexa simulator (Paper I). Y-axis shows the number of active units for every time step during the test phase for the input layer and the autoassociative layer. The activities in both input layer and receiving layer are shown as 2d-images for a selection of the time steps. The receiving layer performed pattern completion by recurrent connections trained by BCPNN (Lansner and Holst, 1996). Additional info in Paper XIV (Rehn et al., 2012).

2.3. Discussion

In Paper I we introduced an explorative parallel simulator, Nexa, mainly focused on large-scale simulations of neural networks on a high level of biological abstraction. These included models comprising rate-based units, non-neural implementations of known functions such as winner-take-all and integration with parallel implementations of
methods for e.g. self-organization and structural plasticity. We solved this by letting parts of a network model which utilized an abstract function to have their own communicators and thereby be able to perform their own collective communications over their network subparts. This way the communication and location of neural units on different processes can largely be hidden from a user implementing such functions. We also demonstrated by using the same method the use of online analysis. This can be crucial to be able to interpret a large-scale simulation since the simulation output can get too large to handle locally.

We studied the performance of Nexa by scaling runs on some of the largest machines available. Besides suggesting solutions to the data handling problems of large-scale simulations by the use of online analysis, we also suggested and implemented real-time visualization run on the same processes as the simulation (Paper XIII). This minimizes the need to transfer information which today is a bottleneck and is expected to become an even bigger problem for the coming generations of supercomputer architectures (Dongarra et al., 2011). By integrating Nexa with the MUSIC framework (Djurfeldt et al., 2010), we used an event-based camera sensor as input to a large-scale simulation running on a supercomputer in realtime (Paper XIV). This could close the perception-action loop and e.g. make large-scale models useful in robotics applications. The simulator and its solutions are freely available to use or incorporate in other simulators (LGPL license) and it is available at github.com/simonbe/nexa.
Algorithms and models

The modeling of the brain is often considered from three different levels as initially introduced by David Marr (Marr, 1982): (1) The ‘hardware’ level consisting of the neural substrate, (2) the ‘algorithmic’ level describing the processes the neural substrate executes and (3) the ‘computational’ level describing the problems the brain solves in in high-level functional terms disregarding the neural algorithm and implementation.

In a bottom-up research approach, detailed biophysical models of the ‘hardware’ level are used, often using a Hodgkin and Huxley formalism (Hodgkin and Huxley, 1952), to describe each neuron with its cell membrane and ion channels in detail. If all parameters in such models were known, whole brains could be simulated on a detailed biophysical level (Markram, 2006; Gerstner et al., 2012). However, there are a massive number of human brain parameters arising from the ~10^{11} neurons each connected to thousands of other neurons resulting in a total of ~10^{14} synapses. To constrain such a parameter space, and to extract computational principles from it, may be an impossible task if based only on experimental findings at this ‘hardware’ level.

In a top-down research approach, one tries to find out the principles at Marr’s ‘computational’ or ‘algorithmic’ level and at a later stage implement them as detailed neural, or mechanistic, processes. If these principles are good estimations of the underlying processes, they can be used to constrain and control parameters and connectivity of more complex simulations. One example is the Hebbian cell assembly theory and its closely related Hebbian learning rule: Originally hypothesized by Donald Hebb (Hebb, 1949), neurons which are active together should develop strong connections between each other (the so-called Hebbian learning rule). This will result in network of neurons with strongly connected subgroups which are activated repeatedly during certain mental processes (so-called Hebbian cell assemblies). The resulting neural network may serve as a content addressable associative memory, where full memory activation can happen from the stimulation of only parts of the neurons in the cell assembly. Perceptual rivalry processes can arise from lateral inhibition in such a network.
The models in this chapter in major parts reside on this higher level. These abstract concepts of memory have also inspired and been used to constrain more biologically detailed models of memory (Lansner, 2009; Gerstner et al., 2012). Also the models presented here (Paper II and Paper III) have parts (a modular model of cortical layer 2/3) which have been investigated in more biophysically detailed models (Fransén and Lansner, 1998; Lundqvist et al., 2006; Lundqvist et al., 2010).

From an even higher top-down perspective, cognitive data can provide clues on the ‘computational’ level. It has been argued that the sparse, noisy, and ambiguous input data humans are exposed to in many ways is far too limited to support the rapidly learnt and used generalizations and other inferences we make and that the use of abstract knowledge could help explain this discrepancy (Tenenbaum et al., 2011). Modeling of cognitive result often disregards the neural substrate components and instead models using e.g. Bayesian methods (Goodman et al., 2011). While it has been hypothesized how the inference in such models could be reduced to the neural substrate (by e.g. stochastic sampling (Pecevski et al., 2011)), both model selection and to include how the computations are carried out have been argued to be of high importance (Tenenbaum et al., 2011). In Paper IV, we maintain a biological base by starting from a hierarchical minimal model with explicit neural substrate components and hypothesize how top-down activity in this model could shape memory representations during learning, lead to rapid generalizations and the encoding of temporal information.

3.1. Paper II: Activity-dependent connectivity

3.1.1. Motivation

Several attractor network models of neocortex layer 2/3 have been formulated focusing on the associative memory functions of neocortex both on an abstract rate-based level (Lansner and Holst, 1996; Sandberg et al., 2002; Johansson and Lansner, 2006) and on a detailed spiking level (Lundqvist et al., 2006). Associative memories work best when the representations are sparse and decorrelated. Sparsification and decorrelation is a process which in this model, is assigned to neocortex layer 4. Here we continued previous work (Johansson and Lansner, 2006) on a model which self-organizes a modular (hypercolumnar - population of minicolumns) structure and also decorrelates the input forming specific receptive fields and response properties of units (minicolumns – vertical cortical columns comprised of 80-120 neurons, here coded as single rate values) in this layer. This abstract layer 4 feeds to the layer 2/3 model.

3.1.2. Model

The proposed algorithm for one module worked in several stages (Figure 2.1 in Paper II): First a sensor clustering followed by a vector quantization step partitioned the input space. This lead to the input sensors being grouped based on the statistical correlations in
the input data. The responses in each group were decorrelated and sparsified in a feature extraction step, again using vector quantization. Finally the data was fed into an associative memory which was used in a feed-forward classification setting.

Partitioning of the input space was performed by first calculating the statistical dependence, as determined from mutual information, between input sensors. This resulted in one value between each pairs of input sensors which represented their dependence. If \( N \) input sensors were to be used, there would be \( N \cdot N \) dependencies. The sensors would be grouped so that sensors showing strong dependencies would end up in the same groups. If we view dependencies as an \( N \)-dimensional graph with the sensors as the vertices, one way to think of this informally is as follows: An inverse of the dependence in \([0,1]\) was taken so that a dependence close to one would result in a low value and a low dependence in a high value. We would then treat this value as the distance between the sensors in another \( M \)-dimensional space. An algorithm that can perform this operation is multi-dimensional scaling (MDS) (Young, 1985). Once our sensors had been positioned in the \( M \)-dimensional space, we performed a clustering to find the clusters. These clusters would then comprise the sensors which showed a strong dependence to each other. Section 2.2.1 in paper II details this when mutual information was used as the measure of statistical dependence.

After a partitioning has been performed, a layer with a number of hypercolumns corresponding to the number of partitions was created. Each hypercolumn was dedicated to a group of input sensors, e.g. for an image a number of pixels. Such a partitioning resulting from the MNIST dataset in Figure 6a can be seen in Figure 6b. As the statistics in image data typically is of a local nature, different hypercolumns will be dedicated to pixels which are spatially close to each other.

---

Figure 6. a) MNIST dataset. b) Grouping of input sensors (pixels) when fed with the MNIST dataset, leading to local patches. c) Example of a specific receptive field for a unit (minicolumn) in one of the patches.
The response properties of the minicolumns were set up in order to be able to represent the training data suitably. This was achieved by clustering the training data: Each of the spatial subparts of the training data, forming the receptive fields of the hypercolumns, were clustered independently of the others. A winner-take-all operation was performed in each hypercolumn which resulted in a sparse and distributed representation. Such representations are suitable for associative memories as it will lead to a high storage capacity and noise-resistance. It is also suitable for pattern completion and classification in cortical layer 2/3 models. Each minicolumn of a hypercolumn was in this way tuned to a ‘prototype’ of the incoming data and this would be its receptive field. For instance, if a hypercolumn would have all pixels from the MNIST dataset as input, the minicolumns’ receptive fields would be prototypes of the entire dataset, which would correspond to ‘1’s, ‘2’s, ‘3’s etc. With several hypercolumns, the prototypes would be subparts of the dataset such as small lines (Figure 6c).

Outgoing connections to a classification layer were trained with the Hebbian learning rule BCPNN (Lansner and Holst, 1996). The representations in the classification layer were set to the different classes of the training data, which resulted in a network which during operation performed classification of incoming test data.

3.1.3. Results

Two ways of using the network was explored.

Firstly it was applied in a classification setting, where Hebbian learning was used to bind representations to an output layer. This was tested on the MNIST dataset (LeCun et al., 1998) with results similar to standard machine learning methods, ~95%. Slightly better results have been achieved later (~97%, unpublished).

Secondly it was used to show benefits of an unsupervised algorithm constantly running and gradually changing how the incoming data is represented. This was demonstrated on sensor data from machine olfaction. Such sensors may change their response properties over time, so called sensor drift. A traditional way of training on some data and then keeping the system constant during operation may result in bad performance as the training data over time may not be representative of the data being tested on. A solution to this problem is to gradually change the system and correct for the drift of the sensors during operation (Figure 2.4 in Paper II and further investigated in Paper XI).

3.1.4. Discussion

Paper II presented a way to build connectivity between two modular layers based on the statistics in the lower layer, a form of structural plasticity. Also, the response properties of
individual units in the upper layer were set by the incoming data. Together, this resulted in a data-driven setup of connectivity and responses.

This can be continued higher up in the sensory hierarchy, creating higher-level representations driven from the statistics of the incoming sensory data, as illustrated in Figure 7. For visual data, this would implement Hubel and Wiesel’s (1962) proposal that neighboring receptive fields in cat primary visual cortex feed into the same complex cells. The patches resulting from the simulated network display this kind of neighboring structure as a result of the statistics in the 2D visual input data. A hierarchy with several stages would lead to units in patches having gradually larger receptive fields, tuned to object features of gradually higher complexity. For other type of modalities, such as high-dimensional olfaction data (Paper III), the statistics may exhibit a non-neighboring topography, whereby the method would allow for the construction of a corresponding high-dimensional topographic connectivity mapping. The fact that the method finds the strong dependencies in a dataset irrespective of its dimensionality makes it applicable to generically cluster strong statistical dependencies. We use a similar setup for data analysis applied to fMRI data in Paper V.

Gradually larger receptive fields and units with more complex response properties are similar to other hierarchical models of the visual processing starting with the NeoCognitron (Fukushima, 1980). Using a winner-take-all operation in each module in the receiving layer corresponds to a MAX operation used also in other cortex-inspired models for object recognition (Riesenhuber and Poggio, 1999; Serre et al., 2007). Here however, instead of using pre-setup gabor-like filters and receptive fields sizes, the receptive fields are setup in a data-driven fashion. Also, instead of constructing complex cells by combining neighboring units, the combinations are also a result of the statistics in the incoming data. For visual data, this results in 2D local combinations due to the locally high correlations.
Figure 7. The method developed in Paper II can be used to construct data-driven connectivity and response properties of units recursively to form sensory hierarchies. Higher-order patches (layer 2) can be constructed from the combinations of lower-order patches (layer 1), resulting in units within the patches with continually larger receptive fields.

3.2. Paper III: Large-scale modeling of the olfactory system

3.2.1. Biological background: The vertebrate olfactory system

Figure 8 illustrates the vertebrate olfactory system with its three major parts: the olfactory epithelium, the olfactory bulb and the olfactory cortex.

The olfactory epithelium within the nasal cavity contains a large number of the olfactory receptor neurons (ORNs) which can sense molecules from single chemicals or complex mixtures. For instance, in the hamster the epithelium harbors approximately 20 million receptor neurons (Hallem and Carlson, 2006). The ORNs project axons to the olfactory bulb located in the brain. This six-layered structure receives converging axons from the ORNs into structures called glomeruli, where the ORNs expressing the same olfactory receptors target the same glomeruli. These are collections of neurons which are spatially grouped together, similarly to how the barrels are structured in the barrel cortex in rodents or columns in human visual cortex. The output is carried from the olfactory bulb to the olfactory cortex and other areas (amygdala, entorhinal cortex, olfactory tubercle
and the anterior olfactory nucleus) by the mitral and tufted cells (MCs and TCs). The bulb also receives feedback input from the pyramidal cells of the olfactory cortex (Shepherd, 1990).

Similarly to other neural sensory systems, the response of the olfactory receptors is combinatorial. That is, a receptor is not specialized towards a specific odour; instead it exhibits a response to several. For each odour stimulus, this will lead to a response pattern across the entire epithelium. This low specificity also allows novel odours and odour mixtures never before encountered to be represented. However, it may make the separation of odour mixtures more difficult, especially if the components are similar (Sandström, 2010).

Contrary to stimuli from other senses, odour sensing is slow and with no spatial low dimensionality (e.g. not two-dimensional like visual input) and mixtures are without any spatial ordering.

Among the tasks the olfactory system performs are the detection and classification of odours, localization of an odour source, background-foreground segmentation (detecting a specific odour in a background of other odours) and mixture segmentation (detection, separation and classification of several components in a mixture odour).

![Figure 8](image)

**Figure 8.** Illustration of the three parts of the vertebrate early olfactory system. The olfactory epithelium with the olfactory sensors the receptor neurons. The olfactory bulb performs basic processing and filtering. More complex processing and storage of odour
memory is performed in the olfactory cortex. This processing also includes, we hypothesize, mixture segmentation.

3.2.2. Motivation

With Paper III we constructed a large-scale model of the mammalian olfactory system. It included an olfactory epithelium, a bulb model using a columnar representation where the columns corresponded to glomeruli and an olfactory cortex where the connectivity between the bulb and cortex was setup in an activity-dependent manner. The olfactory cortex was modeled with a columnar architecture with recurrent connectivity.

Our motivation was threefold, to

1. Explore the characteristics of the model and the coding of single odours and odour mixtures in all three olfactory stages.
2. Suggest and explore the possibility that mixture segmentation is performed by adaptation in the olfactory cortex.
3. Use the model as a platform for data processing in machine olfaction, see Section 3.2.5 (and presented in paper IX and X).

3.2.3. Model

The three-stage model of the mammalian olfactory system, implementing the early olfactory system is shown in a schematic drawing in Figure 9. It comprises the olfactory epithelium (OE), olfactory bulb (OB) and a population implementing holistic processing capabilities that corresponds to the olfactory cortex (OC). Input data into the model was synthesized with the intention to resemble the distribution of various features of ORN response patterns to naturalistic odour stimuli. The OB implemented an interval concentration coding (Sandström et al., 2009), where the receptive fields of different units formed overlapping intervals with preferred concentrations, where the unit's response was maximal, and with soft boundaries in the stimulus intensity domain. Connectivity between the OB and OC was setup in an activity-dependent manner during a training phase where the network was exposed to single odours (varying number of concentrations for each odour depending on the experiment). This used the same self-organizing algorithm as in Paper II (also see Section 3.1). The odour representations in the OC were sparse and distributed with recurrent connections.

The weights in the recurrent connections were set by the Hebbian-based BCPNN plasticity learning rule (Lansner and Holst, 1996; Sandberg et al., 2000). That is, during the training phase, plasticity was turned on for the recurrent connections in the cortical population. During a test phase for each experiment (e.g. when the network was exposed to odour mixtures to test its mixture segmentation capability) plasticity was turned off. The trained recurrent connections implemented a pattern completion capability, where also only partial or noisy stimulus was enough to make the cortical activity state to move
into one of the learnt odours. One key response characteristic was an increasingly decorrelated representation of an odour from the first stage at the OE to the cortical response. This, together with the cortical pattern completion, could lead to drastically different cortical response to different odours and different concentration of an odour even if they had a similar olfactory bulb response. The model olfactory cortex also gave the same response across concentrations of an odour, which is crucial to be able to have stability in olfactory perception and be able to generalize (Wilson and Sullivan, 2011) and a known function of the olfactory cortex (Barnes et al., 2008).

Neural adaptation implemented the mixture segmentation capability where the response for active cortical units was gradually decreased over time (Section 2.3.2 in Paper III). This could also have been implemented by synaptic depression. As the response of units corresponding to one learnt odour went down in the cortical population, units for another learnt odour were allowed to respond. The pattern completion capability enhanced this functionality, as only a partial stimulus was enough to make the cortical activity state move into one of the learnt odours.

The model was constructed and simulated in the Nexa simulator (Section 2.2/Paper I) and all experiments were run on the Lindgren (CRAY XE6 parallel computer) at the PDC center for high performance computing at KTH.

**Figure 9.** The three-stage olfactory model: Synthesized input data resembling ORN response patterns to naturalistic odour stimuli is used with an OB model implementing an interval concentration coding (Sandström et al., 2009). Connectivity between OB and OC is setup in an activity-dependent manner (Section 3.1/Paper II) and the OC implements
odour discrimination and odour mixture segmentation by recurrent connectivity and neural adaptation.

3.2.4. Results

The model was evaluated on classification performance, binary mixtures segmentation performance and multiple mixtures segmentation performance with up to five odour components. Also, a task-dependent training was demonstrated, where segmentation performance was evaluated when the model was exposed more to certain single odours during training.

In the binary mixtures segmentation tasks, the OB layer had 7200 units and the OC layer had 1500 units fully recurrently connected to implement a pattern completion capability. Unit adaptation was active on each unit in the OC layer to facilitate the segmentation. 20 different binary mixtures of varying concentration combinations for a total of 1944 items were tested and the training set consisted of 50 odours of varying concentrations for a total of 2044 items.

The left panel in Figure 10 shows the results for binary mixture segmentation where the two odours were mixed by an equal amount (constant concentration ratio mixing). Correct segmentation was seen for medium to high concentrations. The use of a full model, containing both OB and OC allowed us to trace back to compare the performance with the OB pattern representations. The right panel in Figure 10 shows that successful mixture segmentation corresponded well to the low Euclidean distances between the mixture and its components relative to other odours for the OB input patterns.
Figure 10. Binary mixture segmentation as performed by the olfactory cortex. **Left panel:** Bars display the performance for fully correct segmentation of both components in a mixture (0% if one but not the other odour is detected). Dotted lines display the percentage of correct detection (50% if one but not the other odour is detected). Constant concentration ratio mixing displayed correct segmentation for medium to high concentrations. Some components were still detected at low concentration as seen from the dotted line. **Right panel:** Euclidean distances between all tested binary mixtures with its components and a component not part of the mixture at various concentrations for constant concentration mixing. Successful mixture segmentation in (a) corresponded well to the low Euclidean distances between the mixture and its components relative to other odours for the olfactory bulb input patterns.

The segmentation performance was affected by the number of learnt cortical representations for single odours. In Figure 11 the performance is shown mixtures were segmented for varying cortex size and for training sets of different sizes (1, 2 and ~40 samples of each of 50 odours respectively). For the training sets with few samples, the performance saturated at a small cortex size. As we increased the training set sizes, a cortex with more units was needed to see the same saturation of performance. In other words, when the model was exposed to more single odours, a larger cortex was needed to reach maximum mixture segmentation performance.
Figure 11. Relation between cortex size, single-odour training samples and performance. Trained on 50, 100 and 2044 samples of 50 odours (1, 2 and ~40 concentrations of each odour respectively) and tested on 2044 samples of 20 different mixtures. For the larger training set with 2044 samples, the performance was increased for the lower cortex sizes and it did not saturate as early. As the model was exposed to more single odours during training, a larger cortex was needed to reach maximum mixture segmentation performance.

Behavioural studies have explored the performance of humans for classification of single odours and mixture segmentation of complex odours with multiple components (Laing and Francis, 1989). Figure 12 shows these data points reproduced and compared to the performance of the model. Similarly to the human behavioural data, the model displayed a performance drop as the number of odour components were increased. The components detected by the model in the mixtures with multiple components were part of the mixture but in the failed cases not all of the components were found.
Figure 12. Classification of single odours and segmentation of mixtures composed of 2-5 components. The model was trained on two middle-concentrations of each odour and tested on multiple concentrations. The increased difficulty of the task as the number of mixture components was increased is similar to human behavioural data (Laing and Francis, 1989).

3.2.5. Application: Machine olfaction (Paper X)

The early olfactory system model was further evaluated using real machine olfaction input data.

We replaced the synthesized olfaction data as input data with data coming from a modular polymeric chemical sensor array with 16384 sensors constructed from 29 types of organic conducting polymers (Beccherelli and Giacomo, 2006; Beccherelli et al., 2009; Beccherelli et al., 2010). Left panel in Figure 13 shows an illustration of the polymer sensor array and the right panel shows example responses of sensor elements over time.

The range of selectivities and variability of sensitivities of the sensors in such a chemical sensor array is similar to the diversity of the olfactory receptors. With similarity in the input, one could assume that later stages of the olfactory system, and models of these, could be suitable for the processing of such data.
3.2.5.1. Results

Data from measurements using the large sensor array on three pure analytes, butanone, ethanol, and acetic acid at different concentrations were processed by the OC model.

Three classification tasks and one segmentation task were devised:

**Classification set 1:** For each analyte tested, trials were divided into three categories based on stimulus concentration; the training set was composed of 50% of trials from each category with the remaining samples used as the test set.

**Classification set 2:** The training set was formed out of all trials from the lowest and medium concentration range for each ligand. The test set was composed of the remaining trials representing the highest concentration stimulus. This strategy allowed for validation of the generalization capacity of the model over concentrations.

**Classification set 3:** The training and test set consisted of trials representing alternate concentration categories, i.e. for 6 concentration groups, those numbered 1, 3 and 5 formed the training set and the remaining groups 2, 4 and 6 – the test set.
**Segmentation task:** The training set was formed out of all trials. The testing set was created using 25 trials of a mixture of ethanol and butanone at three concentration levels.

The results obtained with the reduced olfactory cortex model were compared to the outcome of Support Vector Machine (SVM) classification. These are summarized in Table 1. Classification into one of the three classes of single analytes demonstrated a high level of data separability. They were comparable to the classification accuracy rates obtained with a conventional multi-class SVM where one SVM classifier for each class is trained and the classifier(s) with the highest confidence are taken as the result during testing. Although with the multi-class SVM an additional assumption about the number of components present in a mixture is required which is not needed in the biological model. Additional details can be found in paper IX and paper X.

**Table 1.** Classification and segmentation performance on three dataset generated from a polymer sensor array. Comparable results were found for the olfactory model and a linear SVM. Reproduced from Paper X.

<table>
<thead>
<tr>
<th>CLASSIFICATION TASK</th>
<th>DIMENSIONALITY</th>
<th>CLASSIFICATION ACCURACY [%]</th>
<th>SEGMENTATION ACCURACY [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>OC model</strong></td>
<td><strong>SVM$_{lin}$</strong></td>
</tr>
<tr>
<td>Set 1</td>
<td>192</td>
<td>86</td>
<td>84</td>
</tr>
<tr>
<td>Set 2</td>
<td>192</td>
<td>89</td>
<td>89</td>
</tr>
<tr>
<td>Set 3</td>
<td>192</td>
<td>88</td>
<td>90</td>
</tr>
</tbody>
</table>

3.2.6. **Discussion**

In paper III we built an olfactory model and used it on artificially generated odour data for classification and mixture segmentation. We argued that mixture segmentation is accomplished adaptation in combination with recurrent connections in the olfactory
cortex. The performance of the model in the mixture segmentation task was found to saturate with relation to the number of units in the cortical layer, where a larger number of cortical units leads to higher segmentation accuracy. The model exhibited task dependence, i.e. where larger training exposure to certain odours lead to enhanced segmentation performance for the mixtures they contributed to. The dependence of the model’s segmentation accuracy on the number of odour components displayed similar characteristics to psychophysical data reported in experimental studies with human subjects.

We also used parts of this model in a machine olfaction application in a similar real-world scenario. We compared its performance to an SVM in three classification tasks and a multi-class SVM in a mixture segmentation task, and found similar performance: ~90% correct for classification and 60% for mixture segmentation.

Why do we want to pursue the use of olfactory models instead of traditional machine learning methods for the processing in machine olfaction? We see several reasons for this:

(a) With sensors being inspired from biology and sharing several properties of the olfactory receptors, we expect the models to be well suited for processing of the incoming high-dimensional sensor data.
(b) Large-scale models run on parallelized simulators can naturally handle large numbers of input sensors, which is important to be able to handle data from the large sensor arrays.
(c) The models could be used for a number of functional tasks involving chemosensor data on the same platform, including odour recognition, concentration estimation, source tracking, figure-background segmentation and mixture segmentation. Here we performed odour recognition and mixture segmentation in the same model. In a longer perspective, since the “cortical processing” principles are modality independent, such a platform has the potential to be further integrated with other models providing multimodal integration combining e.g. olfactory and visual processing.

3.3. Paper IV: A model of categorization, learning of invariant representations and sequence prediction utilizing top-down activity

3.3.1. Motivation

It has proven troublesome to reproduce biologically realistic invariant representations in computational studies (DiCarlo and Cox, 2007). For instance, realistic invariant object representations should still be sensitive to external information such as location and rotation of the given object instance (DiCarlo and Maunsell, 2003; DiCarlo and Cox, 2007). Also, models solely constructing the response properties for neurons based on
conjunctions of lower-layer neurons and layer-by-layer learning (Fukushima, 1980; Hinton et al., 2006) result in representations which are not consistent with what is known about higher-order visual areas (inferior temporal (IT) cortex): IT representations also carry category-specific information e.g. for human and non-human faces (Kanwisher et al., 1997; Kiani et al., 2007; Kriegeskorte et al, 2008).

Since prefrontal cortex (PFC) generally shows a dense activity pattern during categorization tasks (Freedman et al., 2001) we investigated if such activity patterns could be used to group or tag representations further down in the sensory stream and as a result be shaping invariant memory representations in e.g. area IT.

We were also interested in further exploring if such top-down activity could be a general strategy beyond sensory coding. In particular, we were considering the internally generated cell assembly sequences reported in the hippocampus (Pastalkova et al., 2008), and we hypothesized that top-down activity from such internal circuits could be utilized to control and predict lower-level sequences.

3.3.2. Model

The core model seen in Figure 14a consisted of a population providing sensory input, two stacked populations of threshold-linear units and a top population providing downstream activity to the population below. Each population had symmetric and recurrent inhibitory connectivity resulting in a pattern decorrelation (Wiechert et al., 2010). The symmetric connections ensured the convergence to stable attractor states in each population for a fixed input. The feed-forward and feedback connections between two populations in the hierarchy were set up with random weights at the onset of each simulation. Hebbian plasticity was used in the feed-forward connections between populations and served to associate the stable response of a lower population to a stable response of a higher population. One could interpret such a structure as a mapping of activity in one population with pre-set response properties to activity patterns in other populations and implementing a so-called LEGO memory (Perin et al., 2011; Markram and Perin, 2011).

Model input was varied by both providing static input patterns and rapid sequences of input patterns.

3.3.3. Results

Formation of invariant representations and sequence prediction were investigated to demonstrate the computational mechanisms of the model:

**Rapid one-shot learning of representations gradually shaped by top-down activity.**

The basic behavior in the model, without any top-down feedback, was to gradually create more decorrelated representations of the incoming sensory stimuli the higher up the hierarchy the sensory input progressed (Figure 1 in Paper IV). To this setup we added
backprojections from the top layers. A random, arbitrarily chosen activity in the top layer spread downwards and affected the information content of the representations in lower populations. Learning took place in the feed-forward connections, connecting local assemblies with each other, forming global assemblies that responded to specific inputs. Figure 14 shows the shaping of representations in the model for a random and gradually more correlated dataset by a top-down category signal: The activity in the lower population reflected mostly the statistics of the sensory input, whereas that of the higher population reflected a combination of the decorrelated sensory input and the top-down activity. With feedforward one-shot learning, these category invariant representations were rapidly imprinted into the network. Following this learning the representations remained invariant also in the absence of the original top-down activity present during learning.

For a dataset consisting of rotations of an image (Figure 4 in Paper IV) coupled with top-down activity signaling category during a training phase, we saw that the feedback provided during training shaped the response properties of the network and forced the transfer of category separability for previously unseen rotations in a test set. That is, along the hierarchy gradually more invariant responses to the sensory input could be created by a category signal originating in the top layer. This only required one-shot exposure to training examples and a simple Hebbian learning rule. The biological interpretation of the category signal could for instance have been an assumption about object identity. We studied the response properties of units after learning (Figure 5 and 6 in Paper IV). For an interval of top-down strengths during learning, some units in the layer below the top layer, in the model viewed as IT cortex, became tolerant to the rotations while others became highly specific. This is consistent with experimental findings (DiCarlo and Maunsell, 2003; DiCarlo and Cox, 2007; Zoccolan et al., 2007). The relative amount of the two types of information present in the population response depended on the balance between input strength and top-down projection strength.

**Encoding of temporal relations into spatial representations.** If subthreshold activity in a population did not have time to decay to zero between input pattern presentations, this activity affected the convergence to the stable states. In effect, this subthreshold activity could be seen as part of the input. In consequence, the network still reached stable states, but they were dependent upon the history of stimuli. An example can be seen in Figure 15, where the same sequence of stimuli is presented to a population but with different inter-stimulus times. If a long inter-stimulus time was used, the population encoded the stimuli independent of previous stimuli. If a short inter-stimulus time was used, the population ended up in different stable states dependent on what previous stimuli were presented to the population. The combination of decorrelation properties and lingering sub-threshold activity thereby provided a way to create representations dependent on the temporal history.
We used this transfer of temporal information into spatial representations in combination with top-down activity and suggested how this could be used in tasks such as sequence prediction. We used a fixed sequence of top-down activity starting at the onset of each input sequence. This could be interpreted as the type of internally generated cell assembly sequences reported in the hippocampus (Pastalkova et al., 2008). When combined with external input sequences, we showed that in the middle of an input sequence the network could, deprived of further input, successfully predict the next state using only the internal, prototypic sequence (Figure 7 in Paper IV).
Figure 14. Model with top-down activity and representations shaped by top-down feedback activity. a) Network setup. b) Left panels shows the network activity and right panels show the correlations between stable states in each population. The representations with same sensory input as in Figure 1b in Paper IV is used but shaped by top-down activity. The activity in the lower population reflects mostly the statistics of the sensory input, whereas that of the higher population reflects a combination of the decorrelated sensory input and the top-down activity.
Figure 15. Encoding of temporal relations into different spatial representations. Here illustrated by showing a sequence of 7 stimuli with three different in-between time steps between two stimuli and taking the PCA trajectories of the resulting stable population responses. With a long inter-stimulus time (∞), the population response is independent of the previous stimuli. With a short inter-stimulus time (50), the population response becomes dependent of the history of stimuli and thereby ending up in different stable states which, if read out, can provide information about the temporal history.

3.3.4. Discussion

In the object categorization task, when a dataset consisting of rotations of an image was used as sensory input, both invariant representations within each chosen category as well as sharp population activity transitions across categories were formed (Figure 4 in Paper IV). Units with response properties both tolerant to the rotations and others highly specific to particular rotations were found after learning (Figure 5, 6 in Paper IV), similarly to what has been observed in vivo (Freedman et al., 2003). Units contained information about both category and the specific object instance, thereby representing invariance in the population response rather than specific invariant units, as also seen in experiments (DiCarlo and Maunsell, 2003; DiCarlo and Cox, 2007; Zoccolan et al., 2007).

To handle sequences, we used a mechanism to encode temporal information in the spatial neural representation. This is consistent with findings of monkey IT neurons reflecting the association between a temporally related set of stimuli (Miyashita, 1988). Contrary to the way reservoir computing networks (Jaeger, 2001; Maass et al., 2002) responds to input sequences, the model always ended up in stable states in each population instead of having transient states, allowing for stable binding of temporal information between populations and having a basin of attraction for each state.
Large-scale data analysis of neuroimaging data

Neuroimaging is the use of techniques for imaging of structure or function of the brain. Structural imaging deals with the mapping of structures, such as detection and diagnosis of injuries. Functional imaging can be used to measure changes that relates to the function of the brain, e.g. to detect which parts of the brain that show increased processing during a specific task.

Common functional neuroimaging methods include the techniques electroencephalography (EEG) and magneto-encephalography (MEG) which can map brain activity by recording electric and magnetic fields respectively. Here we will only consider the techniques of functional magnetic resonance imaging (fMRI) and positron emission tomography (PET), though the methods developed could be applied to other neuroimaging techniques as well. fMRI detects changes in blood flow, which is known to be closely related to neural activity (Huettel et al., 2009), by the use of strong magnetic fields. PET detects gamma rays emitted by a radioactive tracer injected to the body.

The neuroimaging methods allow us to get a 3-dimensional image of the brain. The electrophysiological methods like EEG and MEG can detect fast changes but with low resolution, whereas PET and fMRI can detect slow changes but with a high spatial resolution. The resolution can be measured by the number of voxels (volume elements) the 3-dimensional image is divided into. Full width at half maximum (fwhm) can also be used to describe the resolution, which measures at what width half of the signal remains (the smaller the higher resolution). While a high resolution will generate more information by having a higher resolution, the noise level may also increase thus reducing the certainty of this information.

Maximum resolution for fMRI and PET data has been steadily growing for the past decades. For example, the High Resolution Research Tomograph (HRRT, Siemens) which generated the data in our PET study (Paper VI) has an estimated resolution of 1.5 mm fwhm in the centre of the field of view. This high resolution lead to improved
Large-scale data analysis of neuroimaging data

quantification in small brain structures (Schain et al., 2013b). In addition, the imaged region is increased, where the HRRT includes large parts of the head and neck regions. In terms of data points for the PET study, this leads to signals from millions of voxels generated for each time step sampled. The resolution of fMRI hardware is determined largely by the size of the magnet and expressed by the magnetic field strength. The machine used in the fMRI study (Paper V) had a 1.5 Tesla magnet. In terms of data points for the fMRI study, signals from hundreds of thousands of voxels were generated for each time step sampled.

The time resolution of PET data is determined by the reconstruction algorithm. There is a tradeoff between the time resolution and noise, where a higher time resolution can lead to higher noise. As the signal is coming from emissions from an injected tracer, PET measurements have a starting point and an ending point where the tracer is too weak to be used for measurements. This is typically on the order of minutes, where a data point for each voxel is generated every couple of seconds.

For fMRI, the time resolution is typically one measurement for each voxel every ~1-2 seconds. There is no time limit on how long data can be collected. In the fMRI study here (Paper V), every subject was laying in the scanner for a couple of minutes, leading to data for 300 time points for every voxel per subject (10 subjects in total).

4.1. fMRI

Conventional methodology of fMRI analysis has traditionally so-called model-based approaches where the fMRI signal is correlated with a estimated functional response model or where a statistical comparison is performed between control states and activated states (Chuang, 1999). The prime example is the General Linear Model used in the popular software packages SPM (Friston et al., 1995) and FSL (Smith et al., 2004). It can be written in matrix form as

\[ Y = XB + U \]

Where \( U \) contains errors or noise, \( B \) the parameters that are to be estimated, \( X \) contains the experimental design and \( Y \) contains the measured data. In most fMRI experiments conducted, there is a clear way of describing the experimental design, e.g. one knows the exact time-frames when a subject sees a stimulus. This makes it easy to compare the time frames when the stimulus is on and off and the parameters in \( B \) can be estimated by e.g. linear regression.

However, experimental settings exist where there is no easy way of describing the experimental design. One example of this is resting-state data, where people are told to lay in the scanner and rest and not perform an explicity task. Thus, in such data there is no stimulus to correlate against. Then a model-free approach is used which will try to find the existing structures in the data.
Independent Component Analysis (ICA) is commonly used for this in fMRI data analysis. It assumes a predefined number of components of the activity patterns to be linearly statistically independent (McKeown et al., 1998). To cope with the computational complexity the dimensionality of the input data is typically reduced by PCA (Beckmann, 2005).

Another example is the partial correlation whole-brain analysis methods and seed voxel-based methods (Salvador et al., 2005), where a number of areas of the brain are specified and correlations between these are calculated. Clustering can then be performed on the correlation relations between these small numbers of areas. Our model free approach is the most similar to these methods, but we do not divide the brain into areas but rather treat every voxel independently for maximum resolution.

Clustering techniques have also been used, where a predefined number of clusters are adapted to the data according to some statistical distance metric (Chuang, 1999), either directly on the input data (Golland et al., 2008) or on the frequency spectra (Mezer et al., 2009).

4.2. PET

Pharmacokinetic modeling estimates physiological parameters, e.g. receptor or transporter density, from the measured time course of an injected radiotracer from PET images. The radioactivity concentration in arterial plasma represents the model’s input function. Arterial cannulation is invasive, laborious, sensitive to errors and associated with patient discomfort as well as potential adverse events. In the application considered here this arterial input function is estimated directly from the PET images. We compare against the methods which have shown the best performance in the estimates of cerebral metabolic rate of glucose (CMRglc) (Zanotti-Fregonara et al 2009). In Chen’s method (Chen et al., 1992), regions are manually drawn for blood vessels and their vicinities to extract the time-activity curves. Mourik’s hottest pixel method (Mourik et al., 2008) categorizes the voxels with the highest values over time as part of the carotid blood vessels. Naganawa’s approach (Naganawa et al., 2005) uses ICA to extract two independent components and assumes that one of them is representative for the time-activity curve of blood voxels.

4.3. Graph methods

Often we can treat the analysis of neuroimaging and other neuroscientific viewed from the perspective of statistical network theory.

We will then:

1. Treat each element (such as each area or voxel in the analysis) as a node in a graph.
2. Convert the measured time series between all elements to the edges in this graph. This could be performed by calculating the correlation between a pair of elements or some other relation.

3. Analyze the generated graph. For example, if the edges are correlations we could try to find clusters of strong edges to find groups of elements which go together.

A major obstacle is the computational power and memory needed for neuroimaging data. For a full graph with $N$ nodes, the number of edges will scale as $N \cdot N$. For $10^5$ or $10^6$ voxels, this can quickly become difficult to handle. Often a dimensionality reduction is performed on the initial data. Here we show that by employing supercomputers and parallelized software, in the analysis of fMRI and PET data we can treat every voxel as an element, calculate the full graph edges between all nodes, i.e. an all-to-all calculation between voxels, and analyze the resulting graph.

In the fMRI study (Paper V), we used mutual information as the relation for determining the values of edges and analyzed the graph by the methods described in Section 4.4. In the PET study (Paper VI), we used Pearson correlation as the relation. We further reduced the memory needed by only storing the strong correlations to disk. We analyzed the resulting (sparsified) graph by using an input region, a number of nodes in the graph, and by following their strong edges to other correlated nodes.

4.4. Paper V: fMRI resting-state networks

Resting-state data originate from the fluctuations in brain activity when the subject is at rest and are thought to reflect the underlying connectivity of the brain (Smith et al., 2009). There have been a number of model-free methods suggested for this type of fMRI data analysis. For example, the studies of Beckmann (Beckmann et al., 2005), De Luca (De Luca et al., 2006), Damoiseaux (Damoiseaux et al., 2006) and Smith et al (Smith et al., 2009) all used ICA based approaches to study resting-state functional connectivity, whereas the study by van den Heuvel (van den Heuvel et al., 2008) used an approach based on a normalized cut graph clustering technique.

We implemented an algorithm which was able to calculate and keep in memory the pairwise mutual information between all voxels in a dataset. If we interpret this from a graph perspective, each node corresponded to one of the voxels in the dataset and the strength of an edge in this graph was calculated from the mutual information between the nodes. In total, this resulted in 121247 nodes and $121247 \cdot 121247 \approx 1.5 \cdot 10^{10}$ edges. We were interested in the clusters as determined from the groups of strong edges in this graph, as this could tell us about which voxels were functionally connected. One way to do this is as follows: Multi-dimensional scaling (Young, 1985) positions the voxels in a lower-dimensional space (from N to M dimensions) where their location is determined by
the strength of the edges (details described in Section X). That is, two voxels have a strong edge between them if they are strongly correlated as determined from mutual information and will be positioned close to each other in the M-dimensional space. In this lower-dimensional space we can rapidly explore how the voxels are positioned relative to each other and cluster them. A cluster will then be composed of voxels which are highly related activity-wise. These clusters correspond to the resting-state networks.

These steps are illustrated in Figure 16. The calculation of mutual information was the most computationally intensive operation together with the derivation of the positions of all voxels in the lower-dimensional space by multidimensional scaling. Once the voxels were positioned in this space, reflecting their statistical relations as determined from mutual information, the clustering to find the high statistical regularities was computationally inexpensive. This opens up for rapid exploration of the statistics in large datasets and rapid visualization of the statistics on different spatial scales.

Input to the algorithm consisted of 300 MR echo-planar image volumes sensitized to blood oxygenation level dependent (BOLD) signal changes acquired during 10 minutes of continuous rest (fixating on a cross-hair) in 10 subjects (Fransson, 2006). Each image volume consisted of 121247 voxels (non-brain voxels excluded). 10 subjects were processed, resulting in a 10 data sets of size 300 x 121247. The multiple datasets from different subjects were combined by averaging their distance matrices.

**Figure 16.** Illustration of the algorithm for one dataset. In this study (Results section), the total number of voxels in each timestep N = 121247 and M was set to 50. 10 datasets from 10 different subjects were combined by averaging their distance matrices.

4.4.1. **Results**

Figure 17 shows the main results from applying the mutual-information based algorithm to the 10 resting-state fMRI datasets for 60 clusters. Not included in the figure are
components that showed a very strong spatial resemblance to the patterns typically caused by cardio/respiratory pulsations, susceptibility-movement related effects, subject movements as well as components that were mainly located in cerebro-spinal fluid and white matter areas. Displayed are the twelve 12 clusters which showed connectivity patterns that were located in gray matter. Figure 17A and B show the resting-state networks typically associated with the left and right dorso-lateral parietal-frontal attention network, respectively. Further, Figure 17C shows a bilateral connectivity pattern that encloses the left and right auditory cortex whereas Figure 17D shows a resting-state network that includes the medial and lateral aspects of the sensorimotor cortex. The precuneus/posterior cingulate cortex together with the lateral parietal cortex and a small portion of the medial prefrontal cortex is included in the network shown in Figure 17E. The most anterior part of the medial prefrontal cortex is depicted in Figure 17F. The occipital cortex was by the algorithm divided into three sub-networks that encompassed the anterior/inferior (Figure 17G), posterior (Figure 17H) as well the primary visual cortex (Figure 17K). Figure 17I shows a network that included the precuneus/posterior cingulate cortex, lateral parietal cortex as well as the medial prefrontal cortex. The network depicted in Figure 17J involved the bilateral temporal pole/insula region. Finally, the cerebellum was included in the network shown in Figure 17L.

As an example of rapid exploration and visualization of the statistics on different spatial scales in this resting-state dataset, we could simply increase the number of generated clusters to split the statistically strong networks into their sub-networks. Figure 18 shows on the left-hand side how the left and right fronto-parietal attention networks were grouped together in a 40 clusters decomposition. At 60 they were separated into their left and right part. Increasing the number of clusters resulted in further decomposition into their sub-networks as in the 80 decomposition. A similar splitting can be seen for the default network on the right-hand side in Figure 18.
Figure 17. Resting-state networks from a 60-part decomposition: (A)/(B) Left/right fronto-parietal attention network, (C) primary auditory cortex, (D) lateral and medial sensorimotor cortex, (E) posterior cingulate cortex/precuneus, (F) medial prefrontal cortex, (G) anterior/inferior visual cortex, (H) lateral/posterior visual cortex, (I) default network, (J) temporal pole/insula, (K) primary visual cortex (L) and the cerebellum. The color coding shows how far from the cluster centre a given voxel is where brighter red-yellowness indicates a shorter distance to the cluster center.
Large-scale data analysis of neuroimaging data

Figure 18. Example of the decomposition depending on the total number of clusters chosen. Left: The fronto-parietal attention networks shown in a 40, 60 and 80 decomposition. Right: A splitting of the default network occurred between the 20 and 60 decomposition.

As the method successfully found resting-state networks in resting-state data, it could be used as an alternative to other methods such as ICA on the analysis of these types of datasets. With the last clustering step being computationally inexpensive, the method allows for rapid exploration and visualization of the statistics on various spatial scales.

4.5. Paper VI: Arterial input function from PET image voxels

During a PET study, data is collected after a radioactive tracer is administered intravenously into a subject. This data, in the form of 3-dimensional images, will represent the tracer concentration over time. As the pharmacokinetic modeling, which estimates e.g. receptor or transporter density, uses the radioactivity concentration in arterial plasma as the model’s input function, ~0.1mL of arterial blood is drawn every second during the initial phase of a PET acquisition. Several methods have been proposed to calculate the input function from the PET images, i.e., an image-derived input function (IDIF).

Paper VI proposes a method to find an image-derived input function. Similarly to the fMRI calculation, we could view the dataset as a graph where each node would correspond to one of the voxels in the dataset. We calculated the strength of an edge in this graph from the Pearson product-moment correlation coefficient (PCC) between the nodes. That is, for voxel $i$ and $j$, the PCC between their respective time activity curve, $x_i$ and $x_j$, with $T$ time steps, was calculated as
As the number of voxels in this dataset were on the order of $10^6$, the number of edges were on the order of $10^6 \cdot 10^6 = 10^{12}$. In order to handle this large number of edges, only PCCs of voxel pairs displaying a strong measure of correlation were stored in a sparse correlation matrix $M$, so that $M_{ij} = r_{ij}$, for all $i \neq j$. To avoid extracting noisy voxels located in the vessel, all autocorrelations were zeroed ($M_{ii} = 0$). Figure 19a shows one of these sparse matrices taken from a subject in the study. To extract blood voxels, co-registered MR-images were segmented by applying an intensity threshold (Figure 19b) such that the resulting binary mask only included voxels from the carotid artery. Each extracted voxel was used as an index to the rows in $M$. The voxels in the columns of all selected rows were classified as blood voxels, and constituted a “carotid artery mask”, which were applied to all time frames. The use of the MR-images to generate a carotid artery mask replaced the multi-dimensional scaling and clustering used in the fMRI study to find related voxels. An example of resulting voxel clusters obtained from the method using $[^{11}\text{C}]$flumazenil can be seen in Figure 20.

With a carotid artery mask extracted, an average time activity curve (TAC) from all voxels in the mask was created. The radioactivity concentration was corrected for spill-out on early frames ($t < 3$ min) and the tail of the TAC ($t > 3$ min) was scaled with manual samples, resulting in a final image derived input function (IDIF). This could be compared with a measured input function (MIF) from arterial blood sampling. In this study, their area under curve (AUC) and peak heights were compared. The primary outcome measure was however their distribution volume ($V_T$), defined as the AUC of the target TAC divided by the AUC of the input function, when extrapolated to infinity. $V_T$ was calculated with the Logan graphical analysis approach, using both IDIF and MIF (Logan 2000) and performed with PMOD v.3.2 (PMOD Group, Switzerland).

Two different radioligands were used in the present study. These were chosen for different purposes: $[^{11}\text{C}]$flumazenil (GABA$_A$-receptor radioligand) was chosen because its quantification often requires an input function, since the suitability of the pons as a reference region has been questioned (Delforge et al., 1995). $[^{11}\text{C}]$AZ10419369 (5HT$_{1B}$-receptor radioligand) (Pierson et al., 2008) was included to evaluate the approximation of venous sampling as a substitute to arterial sampling. A total of 12 subjects were examined with the HRRT, of which 6 were examined using $[^{11}\text{C}]$flumazenil, and 6 with $[^{11}\text{C}]$AZ10419369.
Figure 19. a) Pearson product-moment correlation coefficient (PCC) between all voxel pairs in an image subset (\(N \approx 10^6\)) stored in a sparse matrix \(M\). The PCC between the time activity curves of voxel \(i\) and \(j\) is highlighted. b) Carotid artery voxels were segmented from coregistered MR-images and used as indices to rows in \(M\) to extract correlated (columnar) voxels.
4.5.1. Results

The image derived blood curves were in good agreement with those obtained from manual arterial sampling (Table 2 and Figure 21). For $[^{11}\text{C}]$flumazenil, the AUC of the input functions was on average slightly underestimated (6.9±14.2%) as compared to the MIF. The AUC of the IDIFs for the $[^{11}\text{C}]$AZ10419369 data was in good agreement with the MIF (-2.3±14.8%). The peak height generally showed larger across subject variability than the AUC for both radioligands; but the impact of peak height on the agreement of the outcome measures was limited. The difference between $V_T$ estimated with IDIF and MIF was on average 2.8±11.8% (Figures 4 and 5 in Paper VI), ranging from 7.3±11.3% for $[^{11}\text{C}]$flumazenil to -1.7±10.7% for $[^{11}\text{C}]$AZ10419369. Linear regression analysis showed good agreement in the estimates of $V_T$ for both radioligands, with intercepts close to 0 and slopes and $R^2$-values close to 1 (Table 2).

The agreement in $V_T$ obtained with the PWC method was better than that obtained with the other methods for both radioligands. For $[^{11}\text{C}]$flumazenil, the PWC method was the
Large-scale data analysis of neuroimaging data

only approach which did not result in significantly different $V_T$ values as compared to MIF, and for $[^{11}\text{C}]$AZ10419369, the across subject variability was smaller.

For $[^{11}\text{C}]$AZ10419369 data, the radioactivity level was similar in venous and arterial plasma at late time points (data not shown). As we only used late time points to scale the IDIF, this means that arterial samples could be replaced with venous samples.

Table 2. AUC and peak difference between IDIF and MIF (calculated as 100x(IDIF-MIF)/IDIF), and linear regression analysis data for all radioligands and methods. Bold values highlight the method providing slope and R2 closest to one, and intercept closest to zero.

<table>
<thead>
<tr>
<th>Input curves properties</th>
<th>Linear regression analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC difference [%]</td>
</tr>
<tr>
<td>[${^11}\text{C}]$flumazenil</td>
<td>Chen</td>
</tr>
<tr>
<td></td>
<td>Mourik</td>
</tr>
<tr>
<td></td>
<td>PWC</td>
</tr>
<tr>
<td>[${^11}\text{C}]$AZ10419369</td>
<td>Chen</td>
</tr>
<tr>
<td></td>
<td>Mourik</td>
</tr>
<tr>
<td></td>
<td>Naganawa</td>
</tr>
<tr>
<td></td>
<td>PWC</td>
</tr>
</tbody>
</table>
Figure 21. Percent difference between $V_T$ estimated with image-derived input function (IDIF) and measured input function (MIF). Note that Naganawa’s method could not be applied to the $[^{11}\text{C}]$flumazenil dataset. The PWC method resulted in the best image-derived input function for the estimation of $V_T$.

In conclusion, we used a prior region as starting points to find the carotid artery regions. This was performed by following the strong correlations as calculated to all other voxels in the dataset. The IDIF generated from the retrieved regions outperformed previously best-performing methods.

4.6. On the use of in-situ visualization in large-scale data analysis

In Paper I and XIII we demonstrated the use of in-situ visualization in large-scale neural simulations. It was proposed as a way to handle the data handling problems one runs into as the generated data increases faster than the network, storage and capabilities of the local machines. Large-scale data analysis run into these problems as well and we can in a similar way handle it by using in-situ visualization to explore the statistics of massive datasets, as illustrated in Figure 22.

Figure 22. In-situ analysis and visualization allow us to remove the data handling during analysis of data. For large-scale data analysis, this can be crucial as the data sizes may be unmanageable to handle on or to transfer to a local machine.
In an in-situ visualization setting in large-scale data analysis, a user would interact to decide what to compute and view results locally, but all data would be localized and analysis would be taken care of remotely. The analysis and visualization could for instance happen on a remote supercomputer, cluster or cloud. There are several reasons why one would want to more deeply integrate the computation with the visualization of results:

- Allows for interactively exploring massive datasets and their statistics in real-time.
- Removes the need to transfer data, store and in other ways handle data
- Completely scalable solution – all compute power needed, both for analysis and visualization, can be on the supercomputer side.
- Immediate feedback from user. For example, in the PET analysis in previous section an expert user could immediately see if the regions selected and output received would be correct.

For the type of data analysis considered here, where there are clinical applications, it has particular advantages: Not only could such settings remove the need for infrastructure in the form of compute power on-site, it also allows for easy distributions of applications to the clinical setting as it can be entirely setup on the remote side.

We extended the libraries developed for the large-scale data analysis in Paper V and Paper VI with bindings to the libSim parallel libraries of VisIt (visit.lanl.gov), similar to how we did it for large-scale simulations (Paper I and Paper XIII). This way, input from a local machine running VisIt could be sent to an instance capable of performing analysis running on a supercomputer and output could be immediately visualized. Figure 23 demonstrates how this can be used: A user can on a local machine visualize the raw dataset which is also in memory on the remote supercomputer. The user can specify and send coordinates (here a bounding box was selected) to the remote machine, which then calculates statistics of specified parts of the dataset. In Figure 23 the dataset was a monkey PET dataset and the user selected a bone region. Remotely the correlations between all voxels in this region to all other voxels in the dataset (≈1M voxels) was calculated. The results were then sent back to the local machine and visualized on top of the original dataset (red regions in Figure 23). This workflow leads to a complete scalable solution to large-scale data analysis: Distributed dataset across a machine, parallelized analysis and parallelized visualization to make sure that all computations are performed where the data is localized, minimizing the need to transfer data.

The parallelized compute engine and bindings to VisIt are part of the software libraries in (VIII).
Figure 23. Prototype of in-situ analysis and visualization on a PET dataset from a monkey. A bone region was selected on a local machine running VisIt by the user from a monkey dataset (green). Correlations between this region to all other voxels in the dataset (+1M voxels) were calculated remotely online by the compute engine (used in Paper V and Paper VI) and the strongest correlations were displayed immediately (red). This allows for a completely scalable solution of the data handling problem in large-scale data analysis. Bindings are part of software package VIII.

4.7. Discussion

With higher-resolution imaging machines, neuroimaging dataset sizes steadily grow in sizes. Often, large datasets are handled by reducing the dimensions. Here we instead developed methods relying on supercomputers and parallelized software to analyze the data on the highest resolution available, the voxel level. Statistical dependencies between all voxels in the datasets were estimated. In the fMRI resting-state dataset we used mutual information as a dependency measure. In the PET dataset, with fewer available time steps to estimate parameters from, we used Pearson correlation.

The resulting full-scale graphs, with calculated all-to-all pairwise dependencies between all voxels, were then analyzed in two application-specific ways:

1) In the fMRI study, multidimensional scaling and clustering were used to find all strong statistical clusters in the dataset. These turned out to be the resting-state networks, positioning the method as an alternative to other methods such as ICA on the analysis of these types of datasets. With the last clustering step being computationally inexpensive, the method allows for rapid exploration and visualization of the statistics on various spatial scales.

2) In the PET study, prior regions e.g. from MR-images could be used as starting points to find the carotid artery regions. This was performed by following the
strong correlations as calculated to all other voxels in the dataset. The IDIF generated from the retrieved regions outperformed previously best-performing methods.

Measurements other than mutual information or Pearson correlation could be used to study other aspects of the data, such as causal relations as determined by Granger causality (Roebroeck et al., 2005). In the fMRI study, multi-dimensional scaling could be replaced by methods known to conserve nonlinear relationships better, such as the Isomap algorithm (Tenenbaum et al., 2000).

We further demonstrated the use of in-situ visualization in large-scale data analysis. This can be used for interactively and in real-time explore massive datasets and their statistics. In clinical applications, such as the estimation of IDIF in Paper VI, it also has the advantage of removing the need for compute infrastructure on-site and would allow for easy distribution of applications to clinical settings.
Conclusions and outlook

This thesis consisted of three parts related to the study of the brain: Platforms (the included Paper I, and also XII, XIII and XIV and software packages VII and VIII), algorithms and models (Paper II, III and IV, and also IX, X and XI) and applications in neuroinformatics (Paper V and VI). All parts either explored the use of or used supercomputers.

A large-scale neural simulator was developed (Paper I) where techniques for the simulation of neural systems on a high level of biological abstraction using supercomputers were investigated and their scalability was shown on some of the largest supercomputers available. We considered analysis and visualization in large-scale neural simulations and argued for and demonstrated scalable solutions relying on a deep integration between the simulation software and tools in order to minimize communication. Neural simulations approaching the size of real brains and their visualization and analysis are still in their infancy. However, the importance of their scalability can be expected to increase as the communication capacity between compute nodes will increase slower than the compute power for coming supercomputers (Dongarra et al., 2011).

Three models were presented in the thesis: Firstly, a model for setting up connectivity and response properties from the statistics of the incoming sensory data, which could be used to build models of sensory hierarchies of units with gradually larger receptive fields tuned to gradually more complex features (Paper II). Secondly, a full-scale model of the mammalian olfactory system where we suggested a mechanism for mixture segmentation based on adaptation in the olfactory cortex (Paper III). The performance of mixture segmentation in the model was found to be consistent with human behavioral data. Thirdly, we presented a model which used top-down activity to shape sensory representations and which encoded temporal history in the spatial representations of populations (Paper IV). We suggested that the key mechanisms of the model could be used to both form invariant representations and do sequence prediction.
Conclusions and outlook

All models presented were on a high abstraction level in the sense that they disregarded detailed biological implementations. Linking detailed models to models of higher abstraction level is needed to understand the computational principles of the brain. The current technological advancements in characterizing the inner circuits of the brain (e.g. Chung et al., 2013) could be expected to lead to drastically higher accuracy for models on a detailed biological level. This could possibly present unique opportunities to link to models of higher abstraction level to achieve an understanding of the computational principles. Alternatively, when such models on a high abstraction level are used to suggest hypotheses about e.g. computational principles, a continuation from current models and a way to constrain them is to build models which try to explain several diverse functions in one model by the same mechanisms (Eliasmith et al., 2012).

For the first application in neuroinformatics (Paper V), with fMRI resting-state data, we used one of the developed models (Paper II) as a starting point and adapted it for exploratory data analysis. It successfully extracted resting-state networks from such data and, contrary to competing algorithms, kept the clustering resulting in the networks a computationally inexpensive step. This opened up for rapid exploration of the statistics in large datasets and rapid visualization of the statistics on different spatial scales. A natural continuation in exploratory data analysis is to, instead of breaking up such a continuous space by clustering, study the topologies in this space (Carlsson, 2009), similar to how dendograms can be used to understand relations between clusters.

For the second application in neuroinformatics (Paper VI), we developed a method to estimate the radioactivity level in arterial plasma from PET images. This can partly remove the need for invasive arterial cannulation and continuous sampling of arterial blood during a PET examination. We segmented blood vessels from PET images and extracted time curves from these. To achieve maximum resolution and to take all data points in time curves into account, the segmentation was performed by massive voxel-to-voxel correlation calculations. The superior performance of the method compared to previously best-performing reported methods shows the relevance of large-scale data analysis in clinical applications.
References


