

Brain Research Applications on Minsky

OpenPOWER Academia Discussion Group Workshop 2017

Andreas Herten, Forschungszentrum Jülich, 10 November 2017

Outline

Brain Research

History

Today's Challenges

Human Brain Project

HBP Pilot Systems

Motivation

JURON

Eurohack

Applications

TVB-HPC

Arbor

PLI ICA

Others

- Forschungszentrum Jülich, Germany
- Jülich Supercomputing Centre
- POWER Acceleration and Design Centre
- Strong connection to neuroscience (HPCNS)



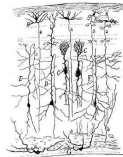
History of Brain Research

A long way down

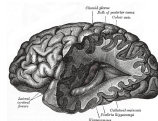
- 1700 BC: Already Egyptians had some knowledge about brain structure
- 17th c.: *Neurology*, status of brain: Thomas Willis (et al)
- 19th c.: Visualization, *neuron doctrine*: Golgi → Ramón y Cajal
- Late 19th c.: neuron electrically excitable
- 20th c.: Brodmann areas (1909); Hodgkin-Huxley model (1952); *neuroscience*



Willis (1664)



Ramón y Cajal (1888)



Gray and Lewis (1918)

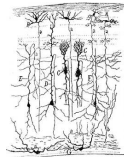
History of Brain Research

A long way down

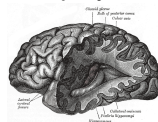
- 1700 BC: Already Egyptians had some knowledge about brain structure
- 17th c.: *Neurology*, status of brain: Thomas Willis (et al)
- 19th c.: Visualization, *neuron doctrine*: Golgi → Ramón y Cajal
- Late 19th c.: neuron electrically excitable
- 20th c.: Brodmann areas (1909); Hodgkin-Huxley model (1952); *neuroscience*
- Today: Brain still not fully decoded
 - Brain atlases in high resolution
 - Models to describe dynamic behavior
 - **Large-scale efforts**



Willis (1664)



Ramón y Cajal (1888)

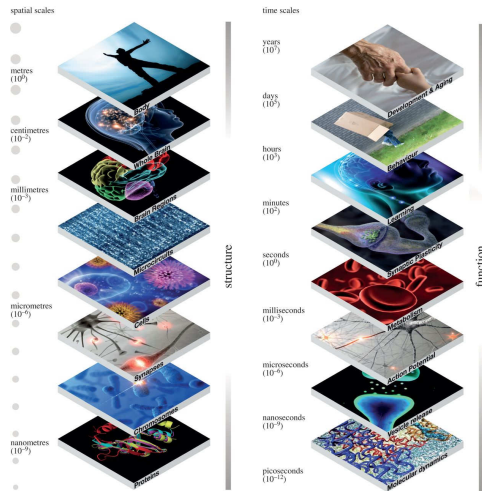


Gray and Lewis (1918)

Today's Brain Challenges

A high complexity

- Many **neurons**: $\mathcal{O}(10^{11})$
- Many **connections**: $\mathcal{O}(10^4)$ synapses per neuron
- **Multi-scale** behavior
 - Molecular level
 - Cellular level
 - Brain regions
 - Whole system
- **Power** efficiency
 - Whole human brain: 30 W
 - Simulation: entire supercomputer to model small region
- Complex **data** collection



Frackowiak and Markram (2015)

Human Brain Project

HBP as a flagship



- 1 Billion € (co-funded EC and national), 10 year endeavor, *2013
- *Future and Emerging Technologies* flagship of European Commission (Horizon 2020)
- 12 sub-projects, covering multiple scales and technologies (SP7: HPC)
- Specific Grant Agreement 1 (2016 - 2018): 114 participants
- **Goal:** Build integrated ICT infrastructure to enable global collaborative effort towards understanding human brain, ultimately emulate its computational capabilities



→ <https://www.humanbrainproject.eu/>

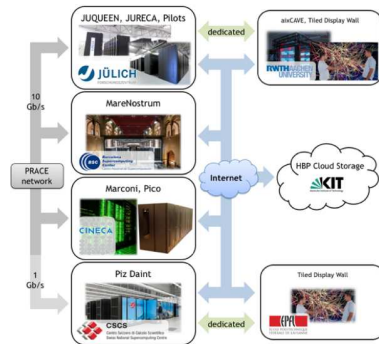
HBP Pilot Systems

- Measuring and simulating brain components: computational intensive!
- High Performance Analytics and Computing Platform (HPAC)

HPAC Platform

High Performance Analytics and Computing

- HPC and data infrastructure services
Currently: loosely coupled, not yet federated
- Current components
 - Supercomputers at BSC (MareNostrum 4), CINECA (Pico, MARCONI), CSCS (Piz Daint), JSC (JUQUEEN, JURECA, Pilots)
 - Cloud services at KIT
 - Visualization services at RWTH and EPFL



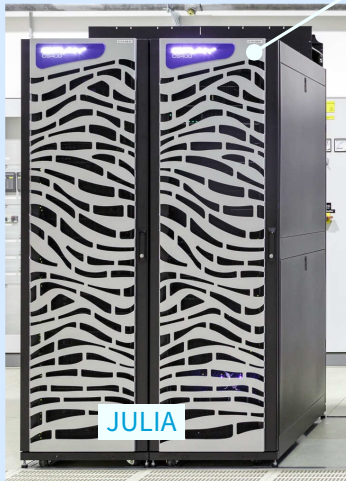
- Measuring and simulating brain components: computational intensive!
- High Performance Analytics and Computing Platform (HPAC)
- Large-scale brain simulations: PFLOP/s, PB
 - Need capability of a supercomputer!

- Measuring and simulating brain components: computational intensive!
- High Performance Analytics and Computing Platform (HPAC)
- Large-scale brain simulations: PFLOP/s, PB
 - Need capability of a supercomputer!
- Special requirements
 - Interactive, scalable visualization (in-situ)
 - Large memory footprint of data (dense memory, fast interconnects)
 - Dynamic resource management, interactive steering
 - Various data sources (eventually: federated services)

- Measuring and simulating brain components: computational intensive!
- High Performance Analytics and Computing Platform (HPAC)
- Large-scale brain simulations: PFLOP/s, PB
 - Need capability of a supercomputer!
- Special requirements
 - Interactive, scalable visualization (in-situ)
 - Large memory footprint of data (dense memory, fast interconnects)
 - Dynamic resource management, interactive steering
 - Various data sources (eventually: federated services)
- Pre-Commercial Procurement of two systems: **JULIA** and **JURON**



JULIA & JURON: Human Brain Project *Prototypes*



JULIA

- Cray (CS-Storm)
- 60 nodes, each 1 Intel Xeon Phi KNL
- OmniPath network

JULIA & JURON: Human Brain Project *Prototypes*

JULIA

- Cray (CS-Storm)
- 60 nodes, each 1 Intel Xeon Phi KNL
- OmniPath network

JURON

- IBM-NVIDIA (Minsky)
- 18 nodes, each 2 P8', 4 P100, NVMe SSDs
- InfiniBand EDR



JULIA & JURON: Human Brain Project *Prototypes*

JULIA

- Cray (CS-Storm)
- 60 nodes, each 1 Intel Xeon Phi KNL
- OmniPath network

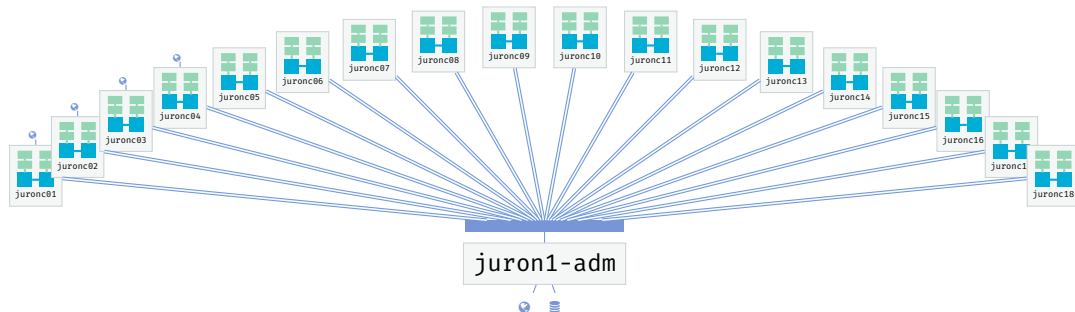
JURON

- IBM-NVIDIA (Minsky)
- 18 nodes, each 2 P8', 4 P100, NVMe SSDs
- InfiniBand EDR

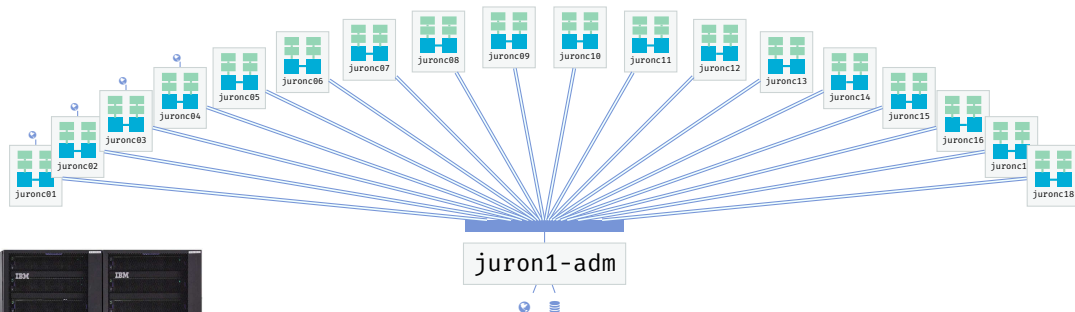
Common local storage

JULIA & JURON: Human Brain Project *Prototypes*

System Configuration



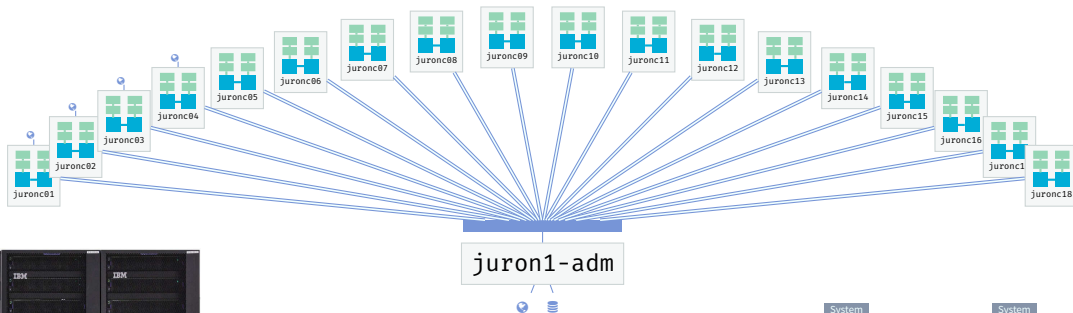
System Configuration



- JURON = Juelich + Neuron
- ≈ 350 TFLOP/s peak (double)
- Memory

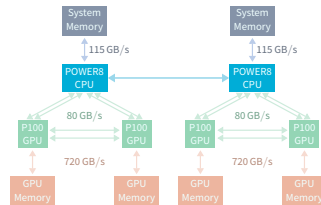
Technology	Capacity / TB	Bandwidth / TB/s
HBM2	1.1	52
DDR4	4.5	4.1
NAND flash	28	0.05

System Configuration



- JURON = Juelich + Neuron
- ≈ 350 TFLOP/s peak (double)
- Memory

Technology	Capacity / TB	Bandwidth / TB/s
HBM2	1.1	52
DDR4	4.5	4.1
NAND flash	28	0.05



GPU Hackathon in Jülich

First HBP Applications on JURON

- **Eurohack**: 1 week of application porting to GPU at JSC in March 2017
 - 10 teams; 3 neuroscience
 - Arbor** Optimizing GPU code of simulation (formerly *NestMC*)
 - TVB-HPC** First port of back-end to CUDA
 - The PLI Guys** Build CUDA back-end to Python simulation
 - >1000 jobs launched, $\frac{2}{3}$ on JURON
 - Every team accelerated code and went home motivated
- Strong interest in GPU and JURON

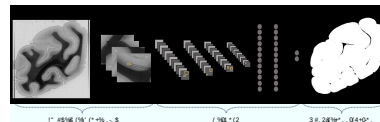


Applications

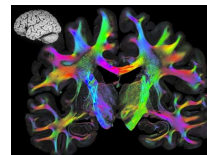
Diverse Set of Applications

Many scales, many steps

- Simulation
 - Regions: The Virtual Brain
 - Neural networks: Nest
 - Neurons with compartments: Arbor, Neuron
- Measurement
 - Coupling of data
 - Electrophysiology
- Post-processing
 - Post-processing of scanned data
 - Automated stitching
 - Matching of images taken with different methods



Khalid (2017)



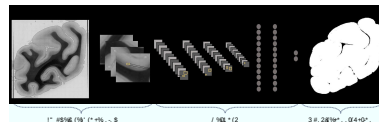
Axer (2017)

Diverse Set of Applications

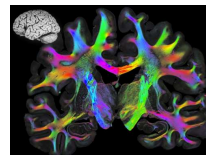
Many scales, many steps

- Simulation
 - Regions: **The Virtual Brain**
 - Neural networks: Nest
 - Neurons with compartments: **Arbor**, Neuron
- Measurement
 - Coupling of data
 - Electrophysiology
- Post-processing
 - **Post-processing** of scanned data
 - Automated stitching
 - Matching of images taken with different methods

nest ::
initiative



Khalid (2017)



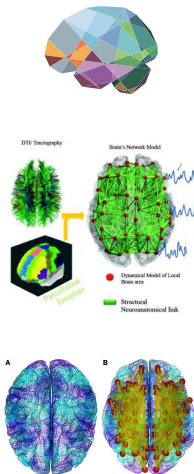
Axer (2017)

Applications

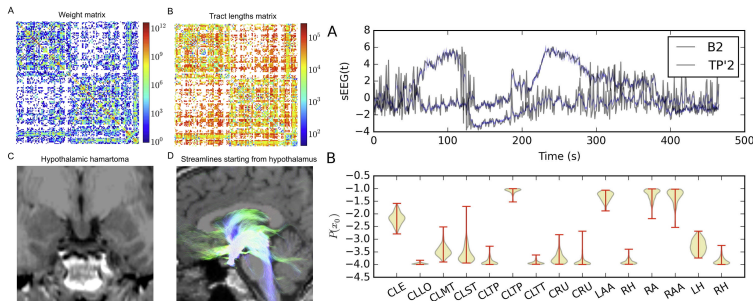
TVB-HPC

- Framework for simulation of brain **dynamics** on large scale [6]
 - Biologically realistic connectivity matrix (*connectome*)
 - Neural mass models, sparse matrix linear solution (60 - 1000 elements), several free parameters
 - Models built on top of clinical data (fMRI, ...)
 - Goal: Help patients with neurological disorders, compare brain
- Current software stack
 - Python simulation core, expendable by Matlab scripts
 - Web-based visual control center
 - Domain-Specific Language to describe brain models (*IDLE*)

→ <http://www.thevirtualbrain.org/tvb/>



Example TVB Science on JURON

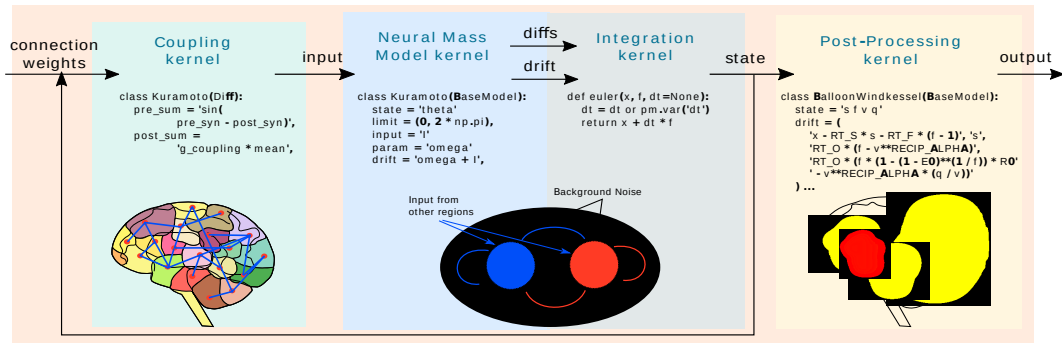


- Recently run: Monte Carlo model inference for clinical epilepsy models
Pictures from “The Virtual Epileptic Patient: Individualized whole-brain models of epilepsy spread” [7]
- Currently single-threaded application; no performance gain yet

- Traditional approach: Serial computation of models
- TVB-HPC: Fast, parallel back-end (parallel in parameters)
- At GPU hackathon:
 - Optimize specific mass, coupling, post-processing models
 - Study data access issues
 - Learn advanced GPU techniques
 - 20× speedup
- **JURON**: CUDA code for *Kuramoto* model as proof-of-concept
- Since then: Automated code generation

Code Generation in TVB-HPC

Targeting different accelerators

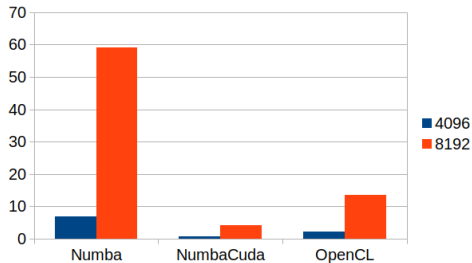


S. Diaz (2017)

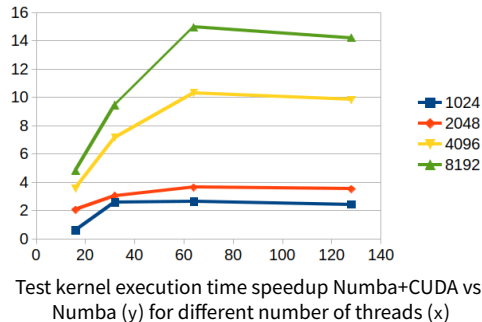
- DSL for models (types, dimensions, flow, dependencies)
- Generate parameter-parallel code with `loo.py`; back-ends: CUDA, OpenCL, Numba, C

Code Generation in TVB-HPC

Results on JURECA



Test kernel execution time (y/ms) for different targets

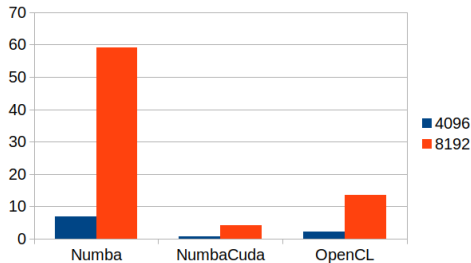


Test kernel execution time speedup Numba+CUDA vs. Numba (y) for different number of threads (x)

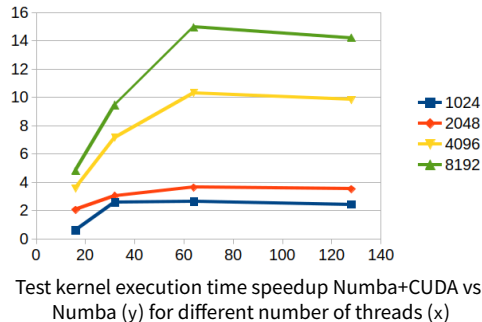
- JURECA node: 2 Intel Haswell CPUs (12 cores), 2 NVIDIA Tesla K80 GPUs

Code Generation in TVB-HPC

Results on JURECA



Test kernel execution time (y/ms) for different targets



Test kernel execution time speedup Numba+CUDA vs. Numba (y) for different number of threads (x)

- JURECA node: 2 Intel Haswell CPUs (12 cores), 2 NVIDIA Tesla K80 GPUs
- **Numba**: Decorator-based auto-acceleration for Python (JIT compilation with `@jit`); different targets

Generated CPU-targeted code

```
@lpy_numba.jit
def loopy_krnl(n, nnz, row, col, dat, vec, out):
    for i in range(0, -1 + n + 1):
        jhi = row[i + 1]
        jlo = row[i]
        for k in range(0, -1 + n + 1):
            acc_j = 0
            for j in range(jlo, -1 + jhi + 1):
                acc_j = acc_j + dat[j]*vec[col[j]]
            out[i] = k*acc_j
```

Generated GPU-targeted code

```
@ncu.jit
def loopy_krnl_in(n, nnz, row, col, dat, vec, out):
    if -1 + -512*bIdx.y + -1*tIdx.y + n >= 0 and -1
    ↪ + -512*bIdx.x + -1*tIdx.x + n >= 0:
        acc_j = 0
        jhi = row[1 + tIdx.x + bIdx.x*512]
        jlo = row[tIdx.x + bIdx.x*512]
        for j in range(jlo, -1 + jhi + 1):
            acc_j = acc_j + dat[j]*vec[col[j]]
        out[tIdx.x + bIdx.x*512] = (tIdx.y +
        ↪ bIdx.y*512)*acc_j
```

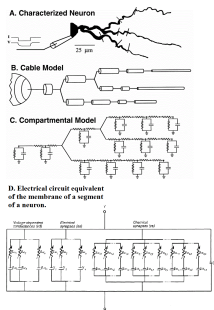
S. Diaz (2017)

- JURECA node: 2 Intel Haswell CPUs (12 cores), 2 NVIDIA Tesla K80 GPUs
- **Numba**: Decorator-based auto-acceleration for Python (JIT compilation with @jit); different targets

Applications

Arbor

- TVB: Large scale; effective models; dynamics
- Nest: Biologically correct; point-like neurons; large, spiking networks
- Arbor: Neurons with internal structure, multi-compartment



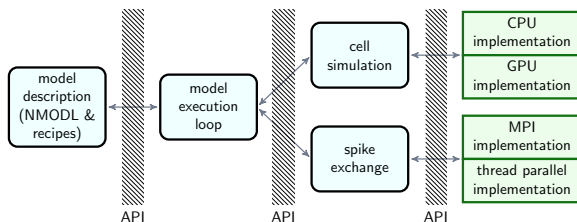
- Hodgkin-Huxley model: network of neurons as circuit [3]
- Neuron: axonic delay, synaptic functions, tree of cables connecting to body
- Cables: electrical compartments (resistance, capacitance)

$$\frac{\partial}{\partial x} \left(\sigma \frac{\partial v}{\partial x} \right) = \left(c_m \frac{\partial v}{\partial t} + r_m (v - e^{\text{rev}}) + \sum_{\text{channels } k} g_k(v, t)(v - e_k^{\text{rev}}) \right) \cdot \frac{\partial S}{\partial x} + \sum_{\text{synapses } k} I_k^{\text{syn}}(v, t) \delta_{x_k} + \sum_{\text{injections } k} I_k^{\text{inj}}(t) \delta_{x_k}$$

→ Neuron is (band) matrix based on known conductance

- Aim: Real-time, morphologically detailed, large-scale simulations
- Optimized for modern HPC systems (parallelism, accelerators)
- Easy to integrate, easy to extend
- Collaboration of JSC, CSCS, BSC
- Open Source Software, modern development methods
- C++, CUDA, Intel Thread Building Blocks, HPX

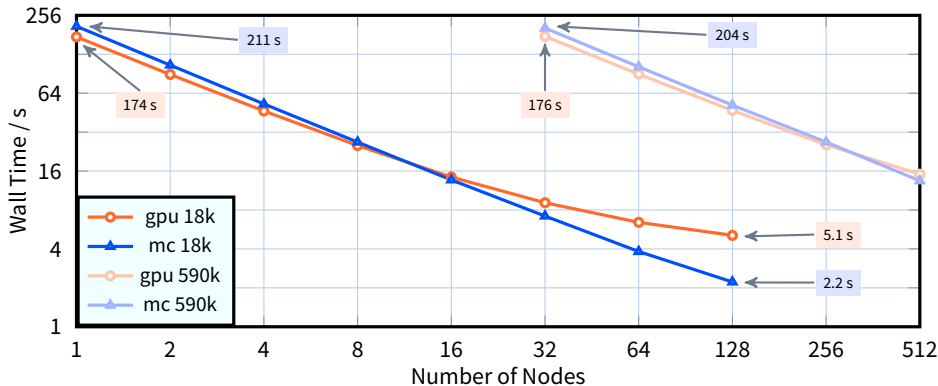
→ <https://github.com/eth-cscs/arbor>



- **Modular:** Substitute models with internal API
- Modeling language: **NMODL** (Neuron) $\Rightarrow$ generate hardware-specific code
- Communication: MPI (global), Intel TBB or C++11 Threads (local threads)
- Backends: CUDA, AVX512, AVX2
- **GPU back-end** available
 - Hackathon project: Optimize sparse matrix computation on GPU
 - Solution: Use padding $\rightarrow 3\times$ to $10\times$ speedup

Current Status

Scaling highlight



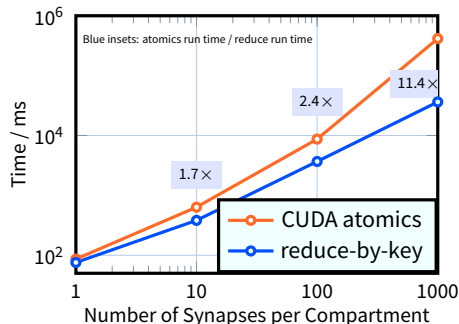
Graph by Ben Cumming

Arbor strong scaling: Time to solution (CPU (Intel Broadwell), GPU (P100), both on [Piz Daint](#)) for small and large model

Current Status

Overview

- Arbor shows good behavior in strong and weak scaling
- GPU acceleration: matrix solver (*Hines* solver); state evolution
- Specific optimizations available, targeting hardware characteristics
Example *reduce-by-key*: Prevent race conditions by warp-synchronous binary reductions
- JURON in use, no dedicated measurements yet



Update time for 10 000 compartments as a function of synapses per compartment

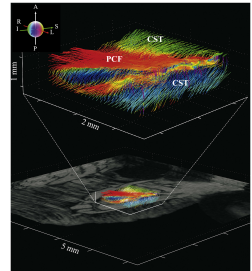
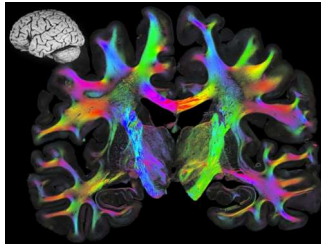
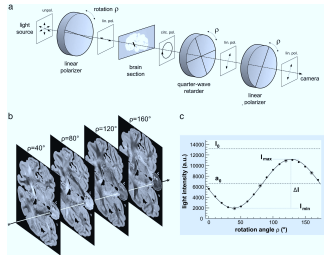
Applications

PLI ICA

3D PLI ICA

PLI...

- 3D Polarized Light Imaging (PLI): Capture brain slices under polarized light
 - Capture at many angles ($18, 0^\circ$ to 170°)
 - Myelin around axons refracts light based on inclination to polarization plane
- Resolve 3D structure of nerve fibers



Images by Axer et al. [8]

3D PLI ICA

...ICA

- Independent Component Analysis (ICA):
Separate complex signal into components;
mixture of sources → individual
contributions
 - Signal-processing method, blind source
separation
 - Basis: Sinusoidal distribution of
measurement basis functions
- Identify noise and artifacts in decomposition
for removal

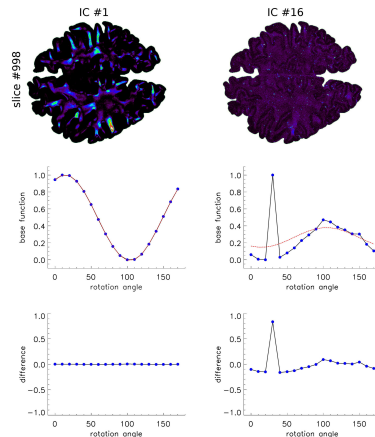


Image by Dammers et al. [9]

- **Computationally intensive** analysis
→ distribute compute and I/O
- **Large data**
750 GB per slice, 325 TB per brain
- Written in **Python**
Cython, numpy, scipy, mpi4py
- Legacy code with many parts
- **GPU Hackathon:**
 - Extract compute intensive part to C
 - Use OpenACC and CUDA for acceleration
 - Prototype-like development

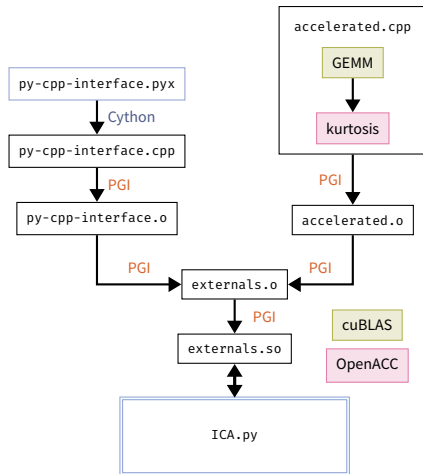
```
e.kurt = stats.kurtosis(np.dot(input_data,  
↪ weights).T, axis=1, fisher=True)
```



```
#pragma acc data copyin(input_v[0:n])  
#pragma acc parallel loop reduction(+:mean)  
for(unsigned int i=0; i < n; ++i)  
    mean += input_v[i]/n;  
#pragma acc parallel loop copyin(mean)  
↪ reduction(+:variance)  
↪ reduction(+:kurtosis)  
for(unsigned int i = 0; i < n; ++i) {  
    double tmp = input_v[i] - mean;  
    variance += (tmp*tmp);  
    kurtosis += pow(tmp,4);  
}  
kurtosis /= (variance*variance);  
return (n*kurtosis-3.0);
```

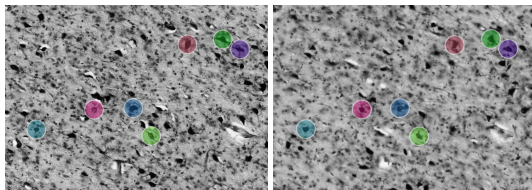
ICA Results of Porting

- Hybrid Python, C, OpenACC, CUDA code
- JURON faster than JURECA
 - Data transfer: $10\times$ (H2D), $6\times$ (D2H)
 - Compute: $7\times$
- Still many parts of program CPU-only $\rightarrow$ limited possible speed-up, data transfer overheads
- Benefit from Hackathon:
 - Create Python interface
 - Speak to experts on libraries
 - Write CUDA prototype
 - Formulate plan for future
- Unfortunately, code is currently rolled back to serial version to fix communication errors



Other HBP Applications on JURON

- Only three applications highlighted (TVB-HPC, Arbor, PLI ICA)
- Many applications more on JURON!
 - 2D → 3D image registration
 - Multi-scale brain image stitching
 - Pattern recognition



Huysegoms (2017)



Khalid (2017)

- Brain research exciting also for computational science
 - Minsky system **JURON** deployed as pilot supercomputer for Human Brain Project
 - System under intensive use (not only by HBP users)
 - TVB-HPC and Arbor: Two brain simulation applications operating on different scales
 - PLI ICA: Cleanup of scanned images
 - Many applications benefit from GPU (and also NVLink)
- HBP helps to drive development of future supercomputing architectures

- Brain research exciting also for computational science
 - Minsky system **JURON** deployed as pilot supercomputer for Human Brain Project
 - System under intensive use (not only by HBP users)
 - TVB-HPC and Arbor: Two brain simulation applications operating on different scales
 - PLI ICA: Cleanup of scanned images
 - Many applications benefit from GPU (and also NVLink)
- HBP helps to drive development of future supercomputers

**Thank you
for your attention!**
a.herten@fz-juelich.de

Appendix

Glossary

References

Appendix

Glossary & References

- API** A programmatic interface to software by well-defined functions. Short for application programming interface. [35](#)
- Arbor** Multi-compartment simulation of neural networks, previously called *NestMC*. [2](#), [32](#), [33](#), [34](#), [35](#), [36](#), [37](#), [43](#), [44](#), [45](#)
- BSC** Barcelona Supercomputing Center, a Spanish supercomputing site. [9](#), [34](#)
- CINECA** An Italian consortium of universities operating supercomputers. [9](#)
- CSCS** The national supercomputing centre of Switzerland. [9](#), [34](#)
- CUDA** Computing platform for [GPUs](#) from NVIDIA. Provides, among others, CUDA C/C++. [20](#), [27](#), [28](#), [29](#), [30](#), [34](#), [35](#), [41](#), [42](#), [48](#)

DSL A Domain-Specific Language is a specialization of a more general language to a specific domain. [28](#)

EPFL École Polytechnique Fédérale de Lausanne, Switzerland. [9](#)

JSC Jülich Supercomputing Centre, the supercomputing institute of Forschungszentrum Jülich, Germany. [9](#), [20](#), [34](#), [48](#)

JURECA A multi-purpose supercomputer with 1800 nodes at JSC. [29](#), [30](#), [31](#), [42](#)

JURON One of the HBP pilot system in Jülich; name derived from Juelich and Neuron. [20](#), [26](#), [37](#), [42](#), [43](#), [44](#), [45](#)

KIT Karlsruhe Institute of Technology, Germany. [9](#)

MPI The Message Passing Interface, a API definition for multi-node computing. 35

NVIDIA US technology company creating GPUs. 29, 30, 31, 48

NVLink NVIDIA's communication protocol connecting CPU ↔ GPU and GPU ↔ GPU with 80 GB/s. PCI-Express: 16 GB/s. 44, 45, 48

OpenACC Directive-based programming, primarily for many-core machines. 41, 42

OpenCL The *Open Computing Language*. Framework for writing code for heterogeneous architectures (CPU, GPU, DSP, FPGA). The alternative to CUDA. 28

P100 A large GPU with the Pascal architecture from NVIDIA. It employs NVLink as its interconnect and has fast HBM2 memory. 36

Pascal GPU architecture from **NVIDIA** (announced 2016). 48

RWTH RWTH Aachen University, Germany. 9

Tesla The **GPU** product line for general purpose computing computing of **NVIDIA**. 29, 30, 31

TVB-HPC High-Performance Computing sub-project of The Virtual Brain. 27, 28, 29, 30, 31, 43, 44, 45

CPU Central Processing Unit. 29, 30, 31, 36, 42, 48

GPU Graphics Processing Unit. 20, 27, 29, 30, 31, 35, 36, 37, 41, 44, 45, 48

HBP Human Brain Project. 6, 20, 43, 44, 45, 48

ICA Independent Component Analysis. [39](#), [40](#), [41](#), [42](#), [43](#), [44](#), [45](#)

PLI Polarized Light Imaging. [39](#), [40](#), [43](#), [44](#), [45](#)

TVB The Virtual Brain. [25](#), [26](#), [33](#), [48](#)

- [1] T. Willis. *Cerebri anatome: cui accessit nervorum descriptio et usus*. Typis Tho. Roycroft, impensis Jo. Martyn & Ja. Allestry, 1664. URL: <https://archive.org/details/cerebrianatomecu00will> (pages 3, 4).
- [2] S. Ramón y Cajal. *Estructura de los centros nerviosos de las aves*. 1888. URL: <https://commons.wikimedia.org/wiki/File:SparrowTectum.jpg> (pages 3, 4).
- [4] H. Gray and W.H. Lewis. *Anatomy of the Human Body*. Lea & Febiger, 1918. URL: <http://www.bartleby.com/107/illus739.html> (pages 3, 4).
- [5] Richard Frackowiak and Henry Markram. “The future of human cerebral cartography: a novel approach”. In: *Phil. Trans. R. Soc. B* 370.1668 (2015), p. 20140171. DOI: 10.1098/rstb.2014.0171 (page 5).

- [3] A. L. Hodgkin, A. F. Huxley, and B. Katz. “Measurement of current-voltage relations in the membrane of the giant axon of *Loligo*”. In: *The Journal of Physiology* 116.4 (Apr. 1952), pp. 424–448. DOI: [10.1113/jphysiol.1952.sp004716](https://doi.org/10.1113/jphysiol.1952.sp004716). URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1392219/> (pages 3, 4, 33).
- [6] Paula Sanz Leon et al. “The Virtual Brain: a simulator of primate brain network dynamics”. In: *Frontiers in Neuroinformatics* 7 (2013). DOI: [10.3389/fninf.2013.00010](https://doi.org/10.3389/fninf.2013.00010) (page 25).
- [7] V.K. Jirsa et al. “The Virtual Epileptic Patient: Individualized whole-brain models of epilepsy spread”. In: *NeuroImage* 145 (Jan. 2017), pp. 377–388. DOI: [10.1016/j.neuroimage.2016.04.049](https://doi.org/10.1016/j.neuroimage.2016.04.049). URL: <http://www.sciencedirect.com/science/article/pii/S1053811916300891> (page 26).

- [8] Markus Axer et al. “A novel approach to the human connectome: Ultra-high resolution mapping of fiber tracts in the brain”. In: *NeuroImage* 54.2 (Jan. 2011), pp. 1091–1101. DOI: 10.1016/j.neuroimage.2010.08.075. URL: <http://www.sciencedirect.com/science/article/pii/S105381191001178X> (page 39).

- [9] Jürgen Dammers et al. “Automatic identification of gray and white matter components in polarized light imaging”. In: *NeuroImage* 59.2 (Jan. 2012), pp. 1338–1347. DOI: 10.1016/j.neuroimage.2011.08.030. URL: <http://www.sciencedirect.com/science/article/pii/S1053811911009232> (page 40).