



GrAI Matter Labs

# NEURONFLOW

REAL WORLD SOLUTIONS FOR LIVE AI

Ingolf Held | CEO

2019 SEPTEMBER 18



# NEURONFLOW TECHNOLOGY

## Architecture Pillars

or

**10x**  
Opportunities

- **Digital Neuromorphics**
- **Dynamic Dataflow**
- **In-memory Compute**



# BIOLOGY BLUEPRINT FOR HUMAN INTELLIGENCE

- Highly connected 3D neurons network
- Computation in network vs 'CPU'
- Event-based processing upon spike
- Analog processing with infinite resolution
- Communication by 'one-bit' spikes



DIGITAL

NEUROMORPHICS

# Neuromorphic Technology Blueprint

for

# Artificial Intelligence

Digital design and packet-switched connectivity

→ Repeatable, shrinkable, scalable, economic

Sparse connected neural networks

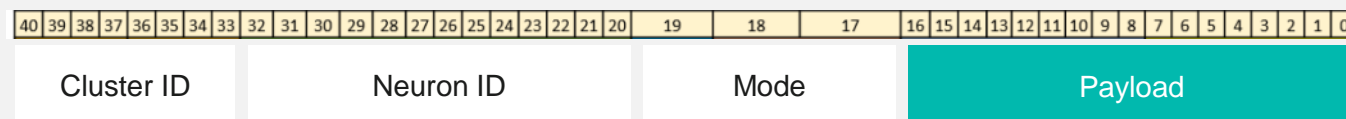
→ Practical in silicon and most algorithms

Valued events instead of spikes

→ Established programming model

**>100x**

Information  
Transfer

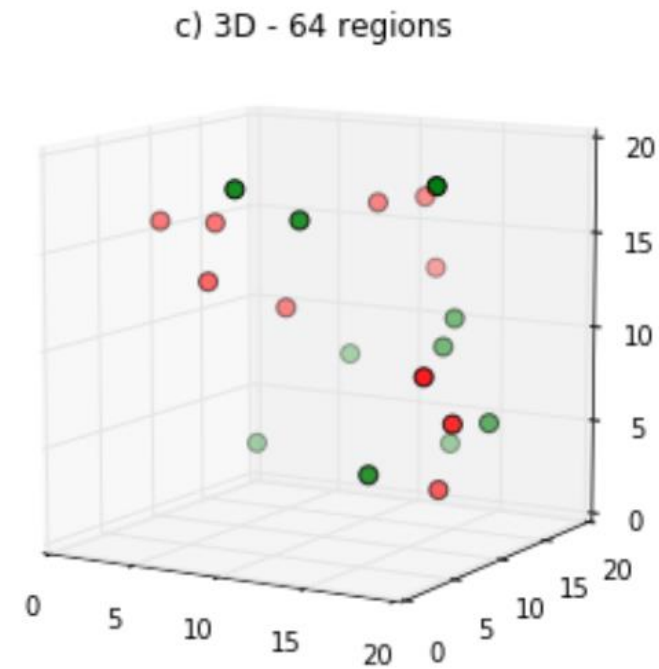
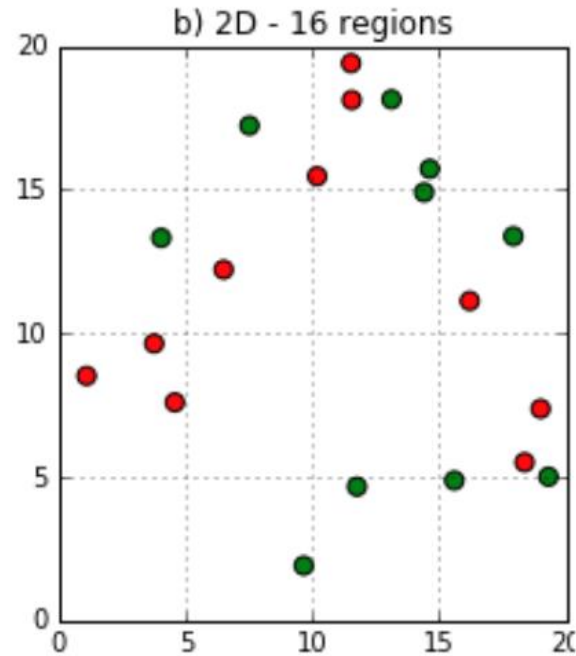
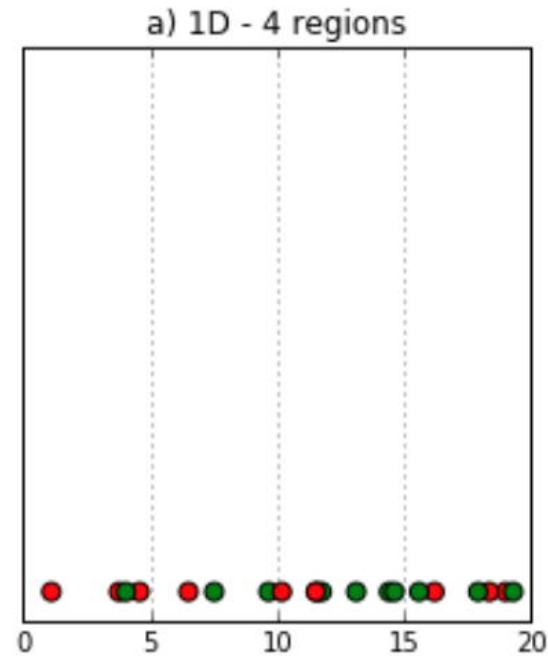


Event-Packet Format

# SPARSITY

## PRINCIPLE

As dimensionality of data gets higher, the number of regions occupied by the same number of data points gets larger →  
**data gets sparser**



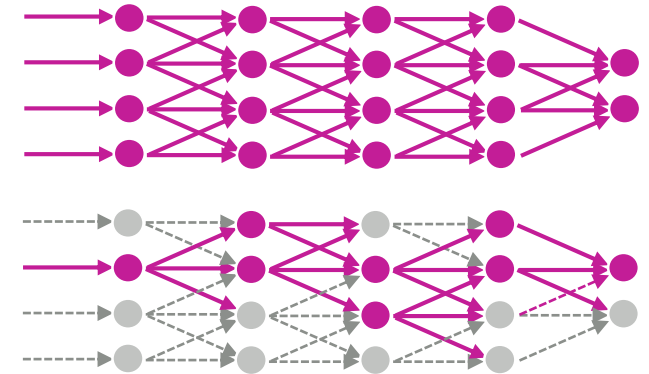
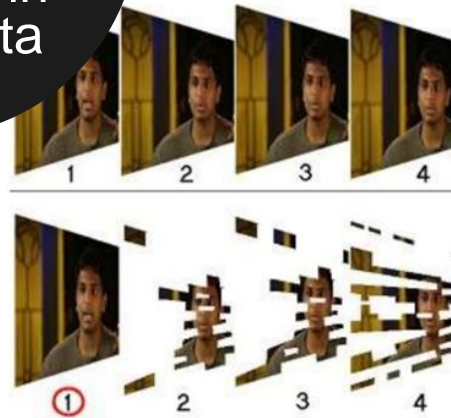


DYNAMIC

DATAFLOW

95%

Sparsity in  
Real Data



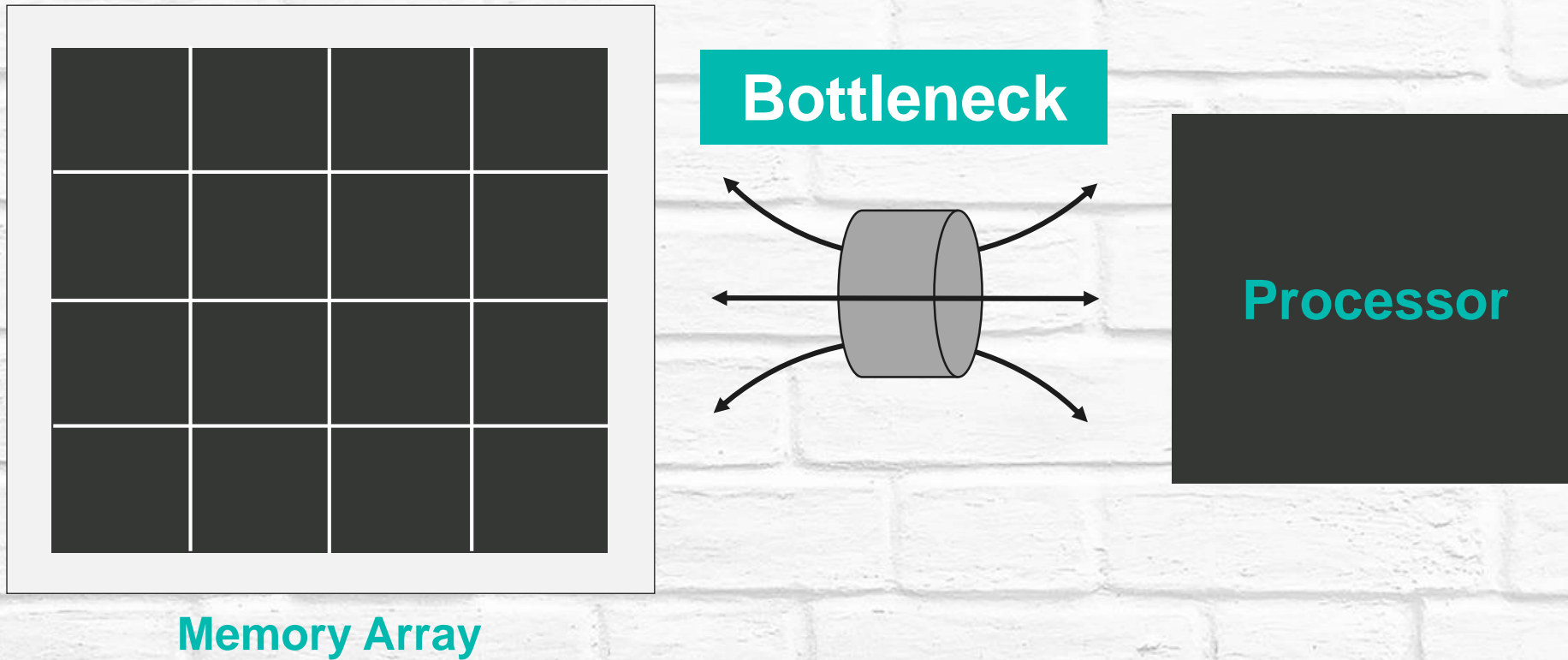
→● = active links and activated neurons

Only process and propagate sparse change **events**

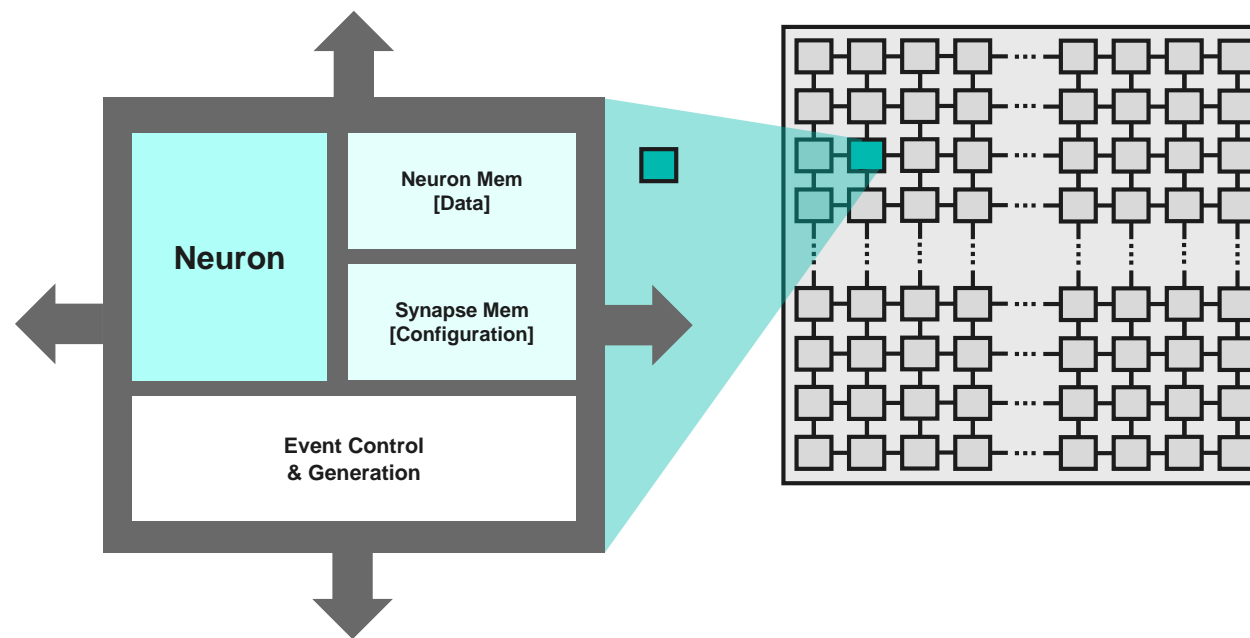
→ **Lower system latency**

→ **Lower power consumption**

# Memory Wall or Von Neumann Bottleneck



# IN-MEMORY COMPUTE



**>10x**

Memory Access  
**Speed**

**<0.01x**

Memory Access  
**Power**

Enables  
**scalability**  
to large  
core counts

Enables  
**sparsity**  
via  
persistent  
neuron mem





# AUTONOMOUS NAVIGATION

Steering control  
in dynamic  
environments

Latency  
 $< 20_{\mu s}$



# COGNITIVE VOICE & VIDEO ASSISTANT

Understanding of  
human speech and  
gestures



Keywords

Latency  
 $< 10\mu\text{s}$



Hand Gestures

Latency  
 $< 1\mu\text{s}$

# GrAI Matter Labs announces ...

## GrAIFlow

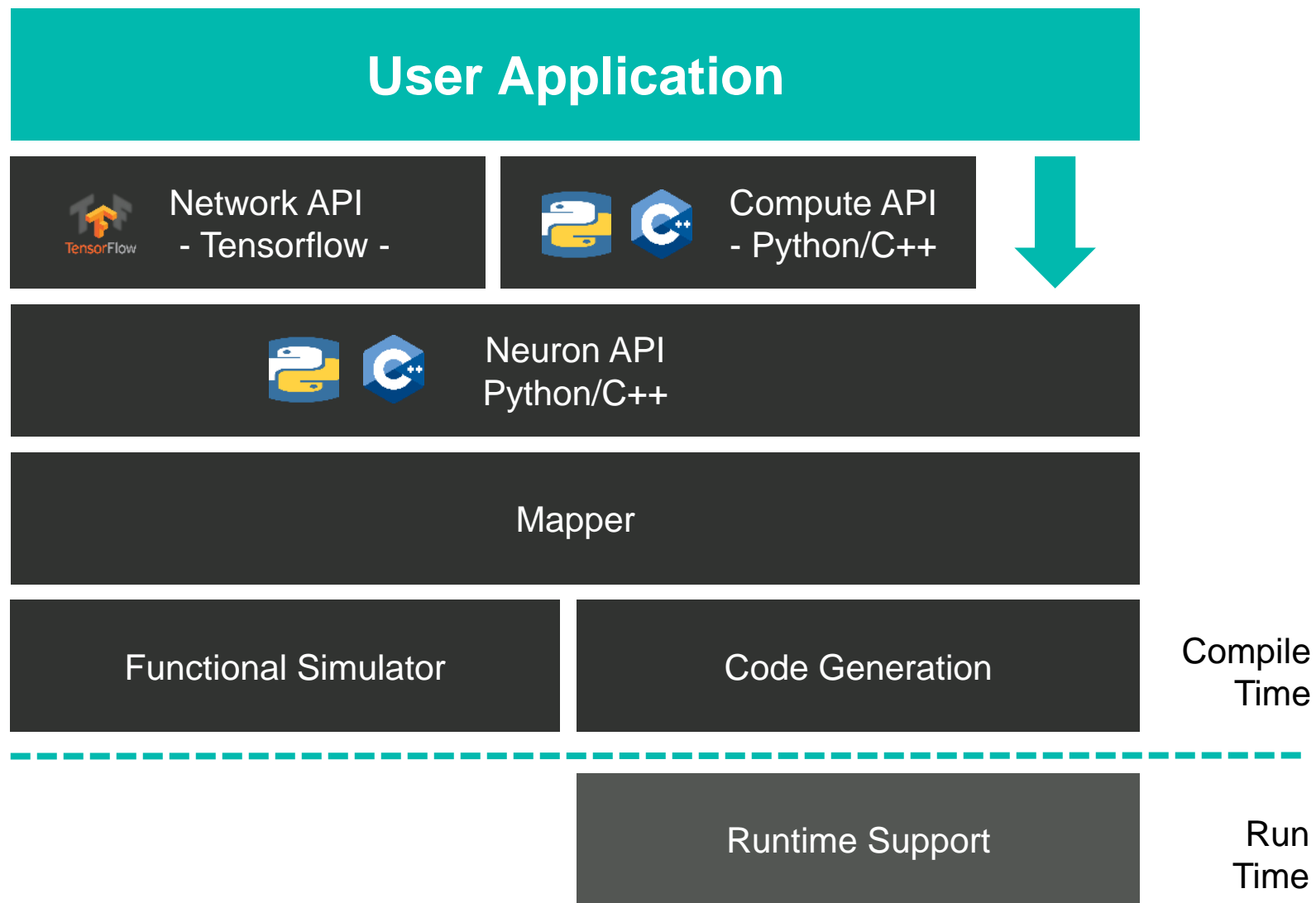


# GrAIFLOW

## SDK

### Key Features

Conventional Programming  
& Machine Learning  
Direct Network Import  
Integrated Simulator  
Graphical Editor





# RNN

# IN GRAIFLOW

Graphical Editor  
for RNN  
programming  
and simulation

Browse through  
hierarchies of  
RNN model

Jupyter  
Notebook  
with RNN  
template

jupyter RNN\_Sample\_application Last Checkpoint: a minute ago (autosave)

File Edit View Insert Cell Kernel Widgets Help

Run Stop Restart Clear Cell Output

### 3- Converting the RNN to a gfggraph

Now suppose that we want to convert that Keras RNN into a **network** th  
(for instance as part of a smart listening device).

To do so, we will design a couple of functions using the gfggraph API, tha

#### step 1: gfggraph library and guidelines

```
In [5]: import gfggraph as gfg
```

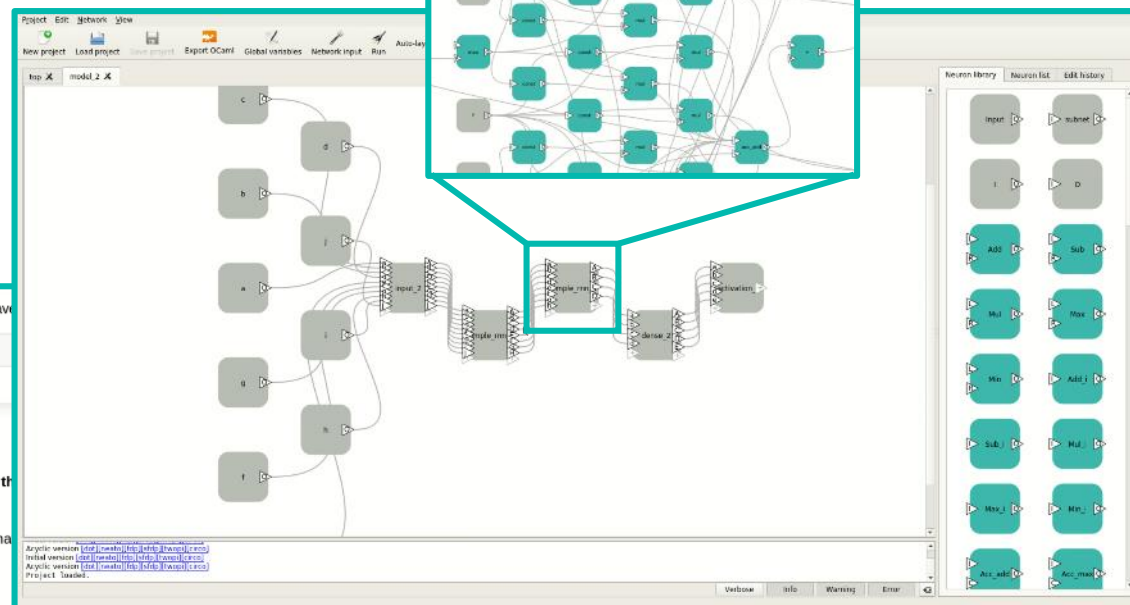
Gfggraph is the low-level API that deals with creating neurons, synapses and networks aggregated into a single .gfggraph protobuffed topology that can later be simulated using a provided set of inputs.

In this notebook, we try and provide you with some **good practices and intuitions** regarding coding styles and design patterns when using the gfggraph API. This is only indicative though , so feel free to experiment on your own! In the code here we impose ourselves the following rules:

- we will design a couple of **functions that are associated to subnetworks creation**. Such functions actually return functions themselves, following the template:

```
In [ ]: def my_function(**kwargs):
```

```
    """  
    My function is not directly applied on input_data, it returns a functor that does.  
    The functor is parametrized using my_function kwargs  
    """
```



BUSINESS

Fabless  
Semiconductor

FOUNDED

2016

FUNDED

Private

# GrAI Matter Labs

