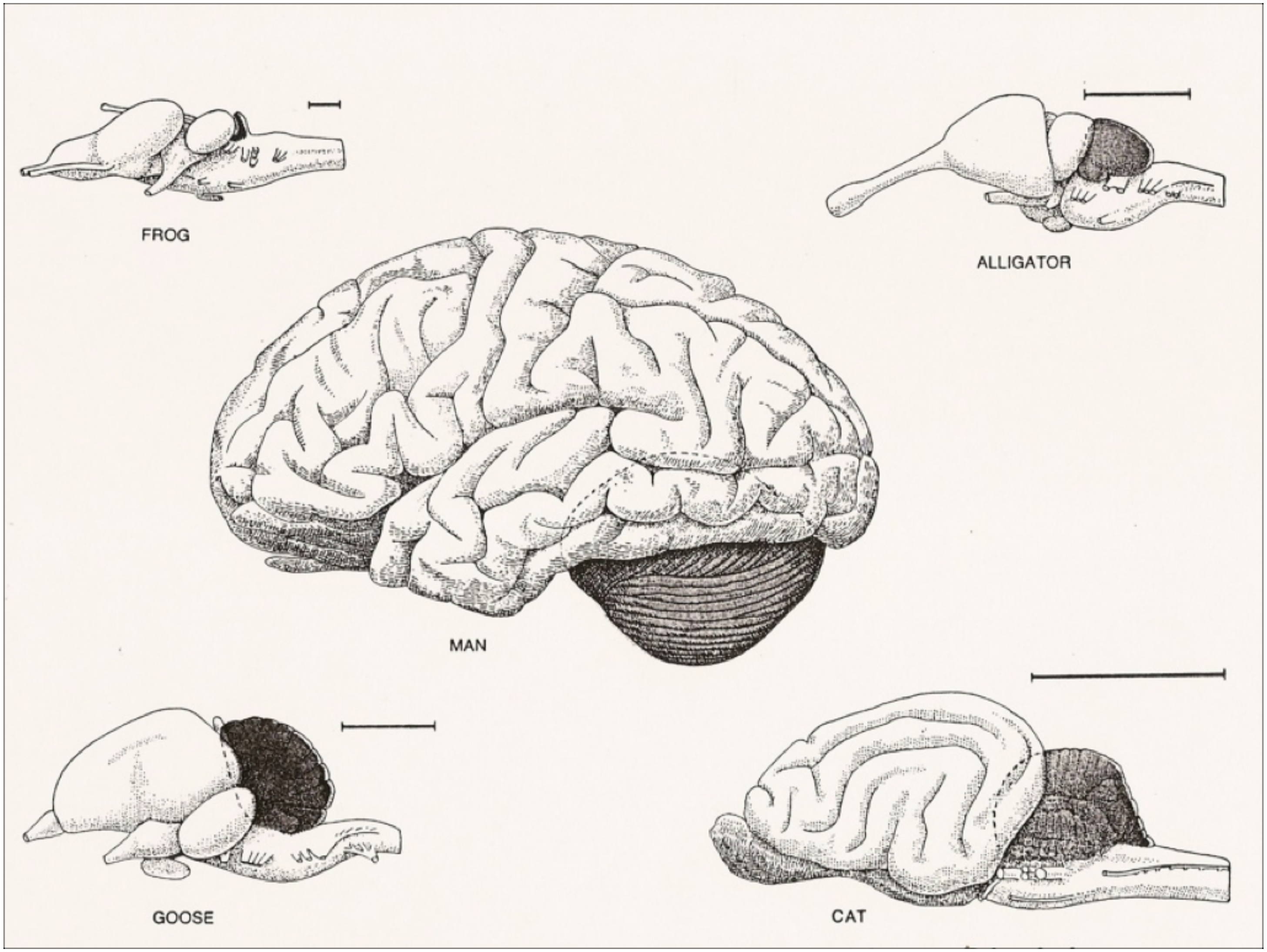
# CEREBELLUM-LIKE MEMORY

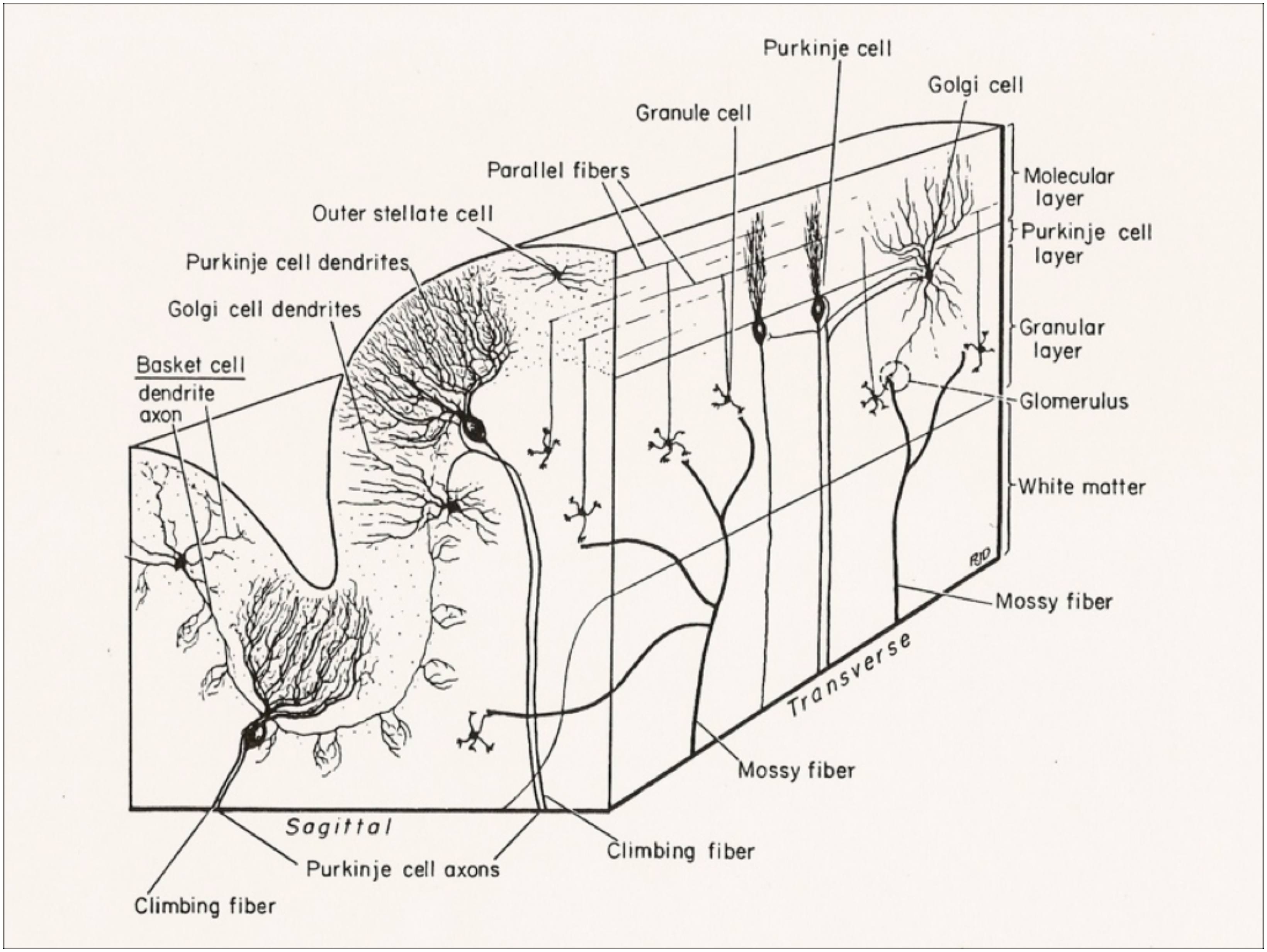
# for COMPUTING with

# HIGH-DIMENSIONAL VECTORS

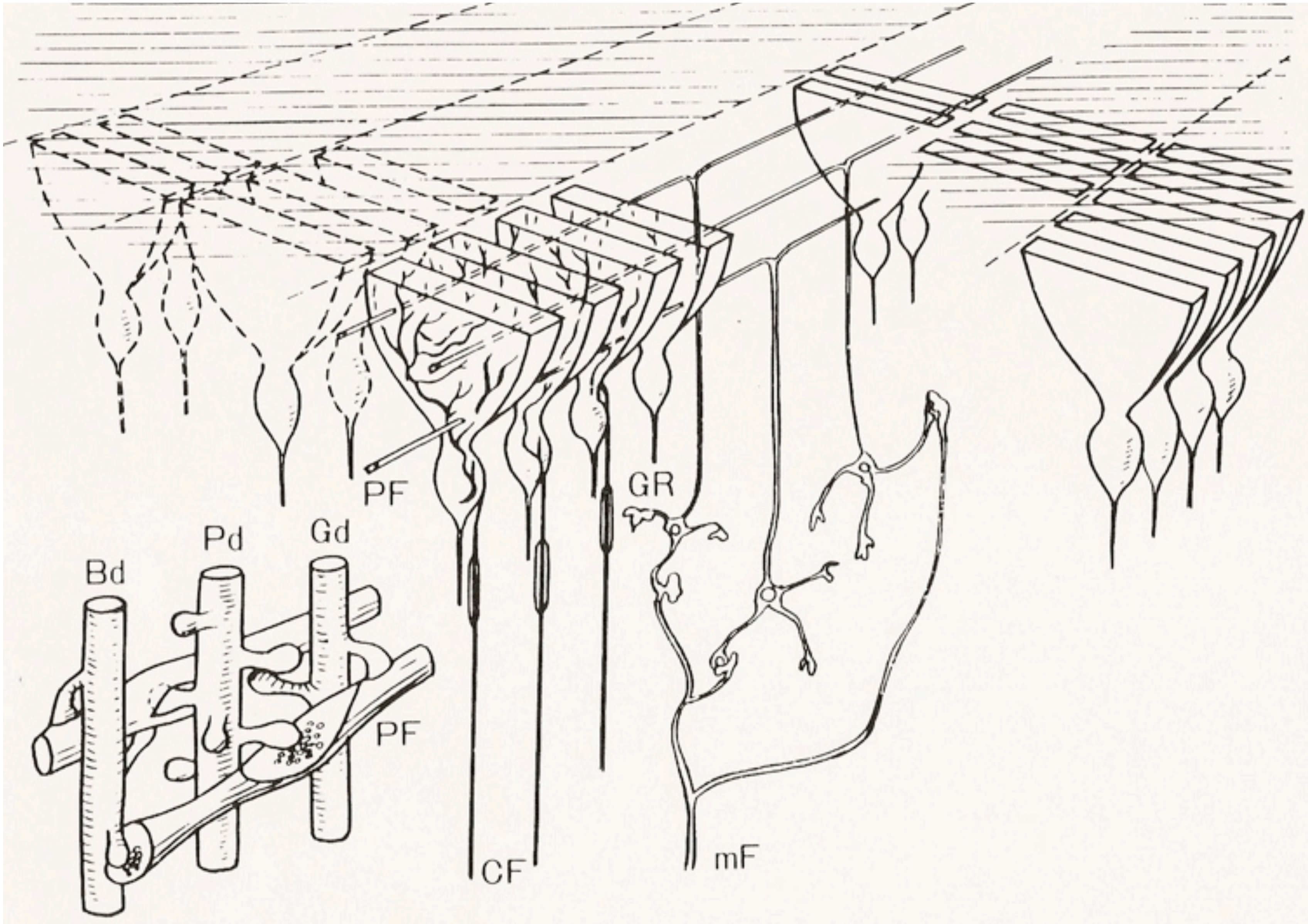
# Cerebellum Figures and Facts



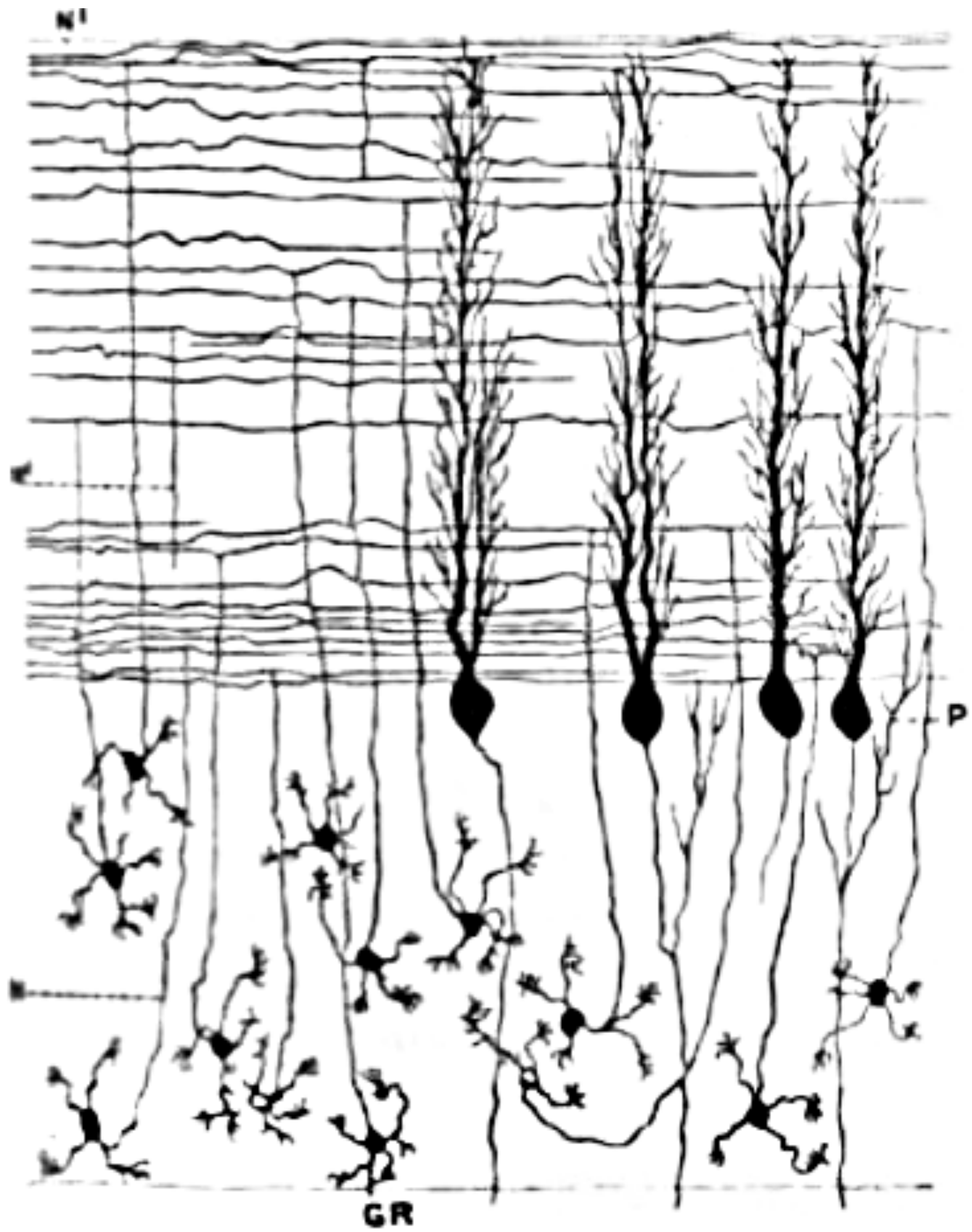

Cerebellum as part of vertebrate brains.




Cerebellum cell types in 3D.




**3D organization of the "main" circuit.**




Side view: parallel fibers and Purkinje cells

**Cerebellum Facts and Figures**

. 200 million Mossy Fibers: external input

. 40 billion Granule Cells, the most numerous

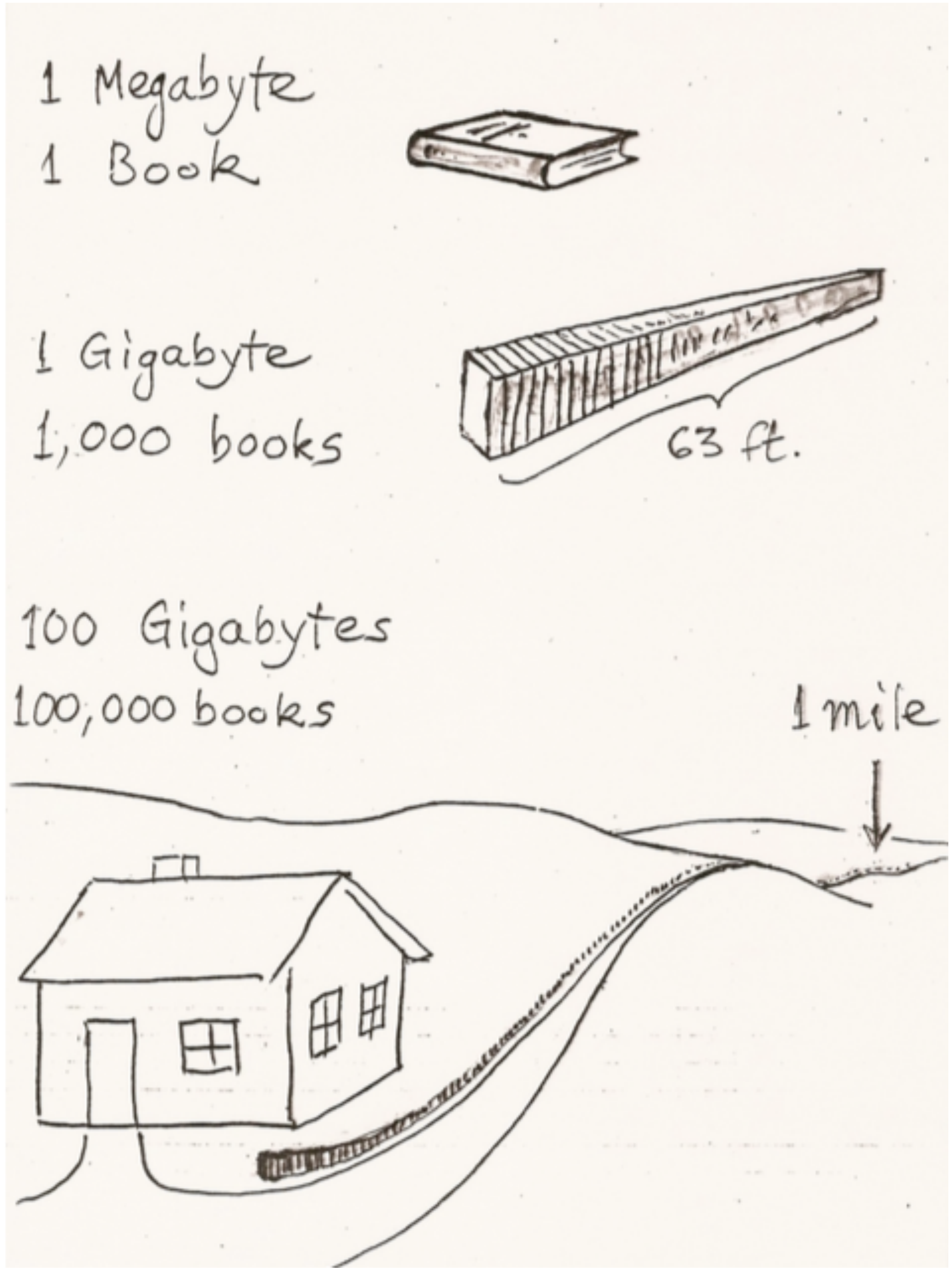
. 15 million Purkinje Cells (PC): sole output

. Climbing Fibers: internal input
-- 1/PC, shared by approx. 10 PCs

100,000 synapses/PC
. 1.5 trillion synapses overall

**How big is 1.5 trillion?**

1.5 trillion synapses @ 1 bit/synapse
= 360,000 books, 400 pages each
= 4 miles of shelf space


1 Megabyte
1 Book
1 Gigabyte
1,000 books
63 ft.
100 Gigabytes
100,000 books
1 mile

# THE CEREBELLUM CHALLENGE

Traditional theories--logic, rule-based AI, artificial neural nets, connectionism, parallel distributed processing, deep learning--leave too much **unexplained** and **unexplored.**

For example, **why the cerebellum, when**

. it has 40 billion neurons
vs. 16 billion in the rest of the brain

. its organization is simple and highly regular

**The cerebellum must fulfill some essential function that computational theories and models of the brain cannot afford to ignore !!**

# TRADITIONAL (VON NEUMANN) MODEL FOR COMPUTING WITH NUMBERS

```
         Central Processing Unit (CPU)
           .-----------------------.
  Input -> |    Program Control    | -> Output
           |-----------------------|
           | Arithmetic/Logic Unit |
           |      + x Boolean      |
           '-----------------------'
                      ^ |        von Neumann
                      | v         bottleneck
.---------------------------------------------.
| # # # # # # # # # # # # # # # # # # # # # # |
| # # # # # # # # # # # # # # # # # # # # # # |
| # # #                                 # # # |
| # # #    Random Access Memory (RAM)   # # # |
| # # #                                 # # # |
| # # # # # # # # # # # # # # # # # # # # # # |
| # # # # # # # # # # # # # # # # # # # # # # |
'---------------------------------------------'
```

## Random-Access Memory (RAM) for Storing Numbers

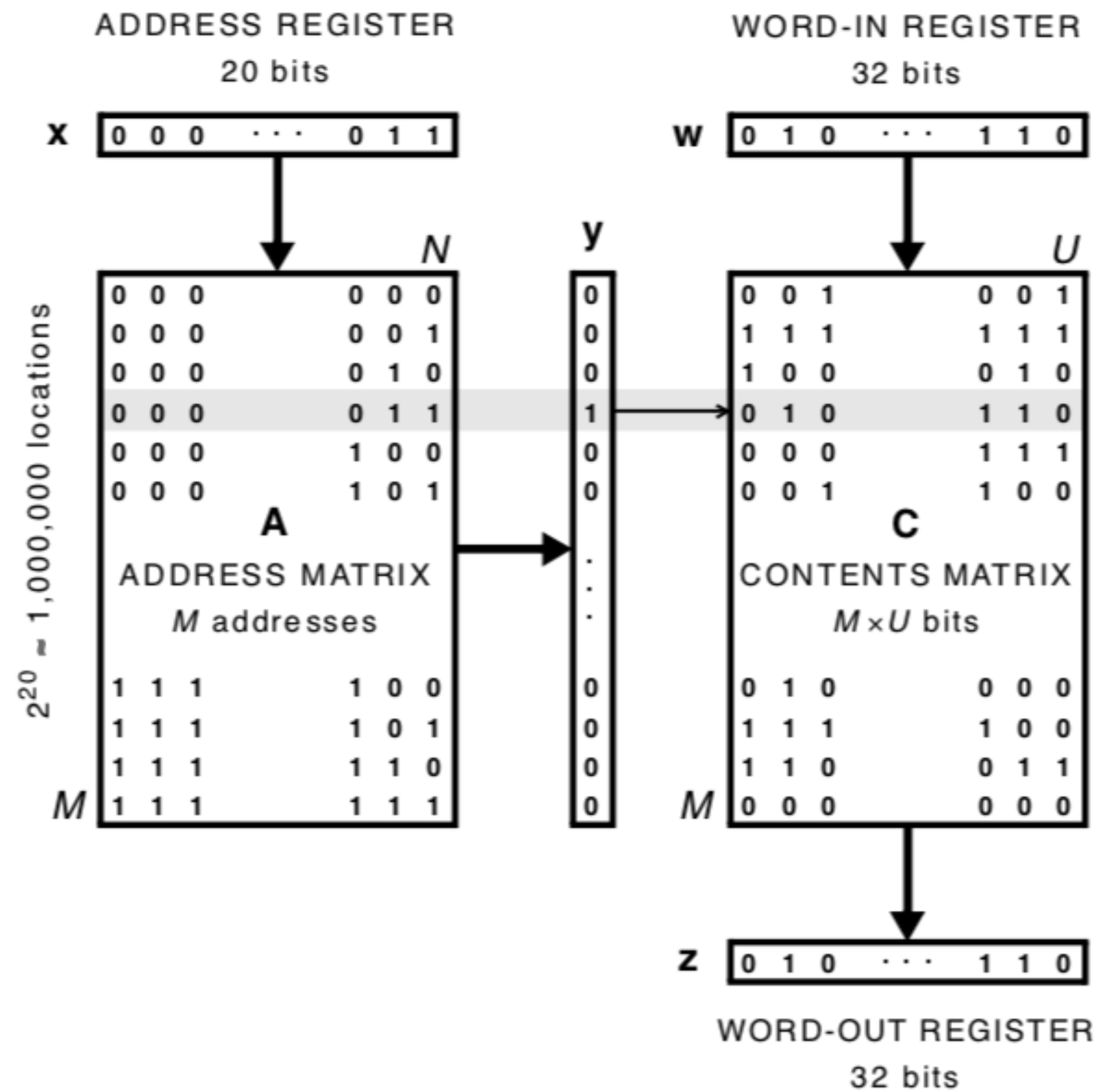
. Millions of addressable memory **locations:**
the **mega**bytes

. Each capable of storing a **number**

. **Address-decoder circuit** selects the addressed
location for storing or retrieving a number

. The **circuit area** of the RAM is much larger than
of the rest of the computer (CPU)

# Random-Access Memory (RAM)



# ASSOCIATIVE MEMORY MODELS OF THE CEREBELLUM

First mathematical models of a major neural circuit

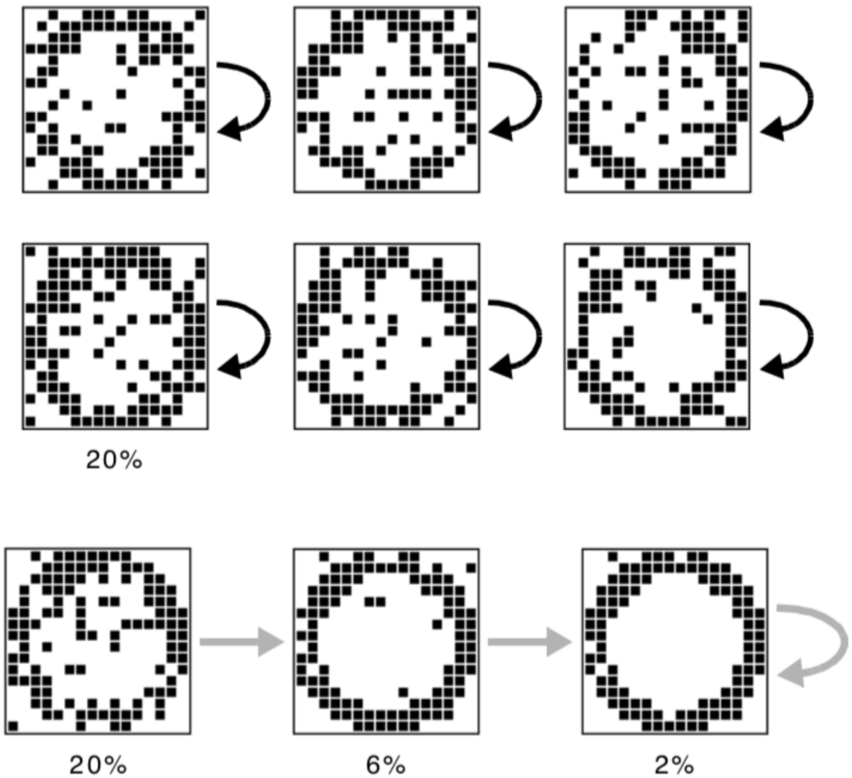
. still today the most comprehensive and credible

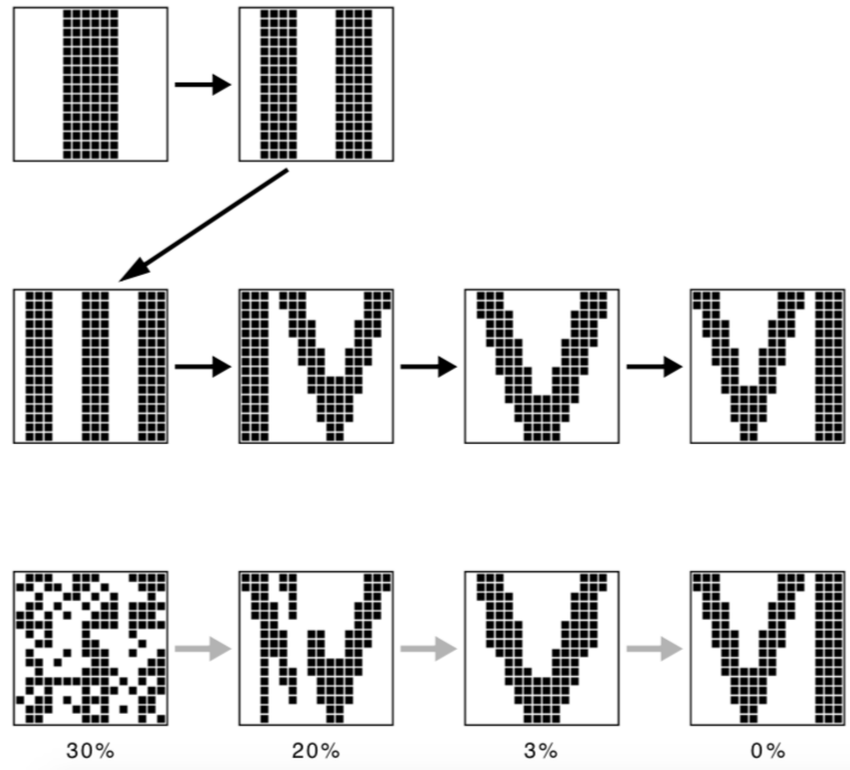

Marr D (1969). A theory of cerebellar cortex. *Journal of Physiology (London)* 202:437-470.

Albus JS (1971). A theory of cerebellar function. *Mathematical Biosciences* 10(1/2):25-61.

Kanerva P (1984/1988). *Sparse Distributed Memory*. MIT Press.

# Sparse Distributed Memory (SDM) as a neural-net associative memory for high-dimensional vectors




**Left**: Six vectors with 20% noise are stored, seventh retrieves a nearly noiseless vector in two iterations. **Right**: Sequence of six vectors is stored. Noisy third vector initiates the retrieval of noise-free sequence.

**High-dimensional representation** (e.g., 10,000-bit vectors) **is subtle and counterintuitive**

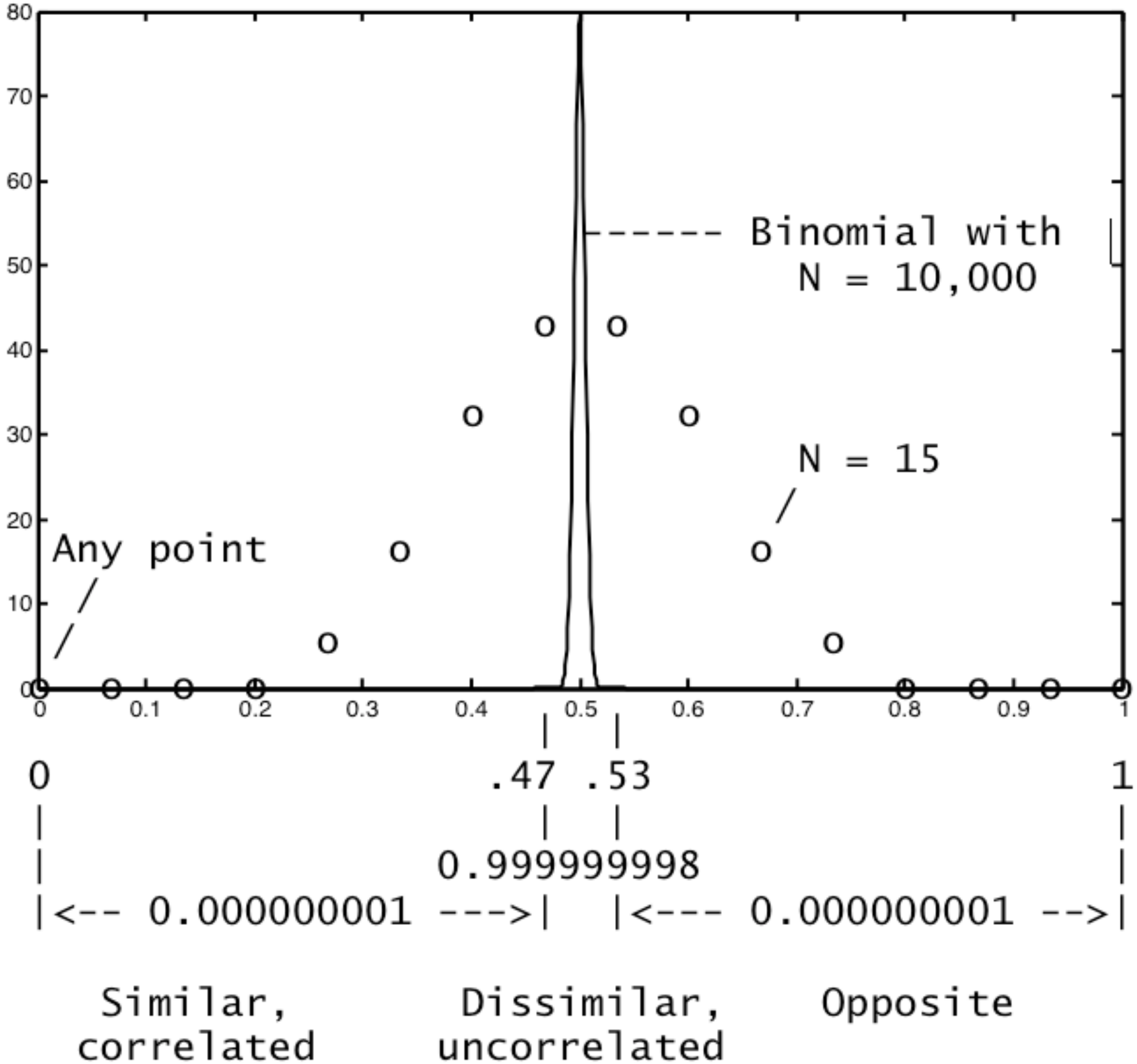
Nearly all pairs of vectors are **dissimilar**
. pairs of random vectors are *approximately orthogonal*
-- makes representation noise-tolerant, **robust**

**Distant concepts** have **similar neighbors**

man ≉ lake
man ≈ fisherman ≈ fish ≈ lake
man ≈ plumber ≈ water ≈ lake
plumber ≉ fish

**Small cues** bring forth complete memories:
"The name begins with T--oh yes, Steven"

Can explain the *tip-of-the-tongue phenomenon*




Binomial distribution, $N$ = 15 and $N$ = 10,000

# VON NEUMANN-LIKE ARCHITECTURE FOR COMPUTING WITH HIGH-D VECTORS

```
         Central Processing Unit (CPU)
           .-----------------------.
  Input -> |    Program Control    | -> Output
           |-----------------------|
          /| Arithmetic/Logic Unit |\
        /  |     + x permute       |  \
      /    '-----------------------'    \
    /                                     \
  /          "In-memory" computing          \
.---------------------------------------------.
| vec vec vec vec vec vec vec vec vec vec vec |
| vec vec vec vec vec vec vec vec vec vec vec |
| vec vec                             vec vec |
| vec vec      Associative Memory     vec vec |
| vec vec                             vec vec |
| vec vec vec vec vec vec vec vec vec vec vec |
| vec vec vec vec vec vec vec vec vec vec vec |
'---------------------------------------------'
```

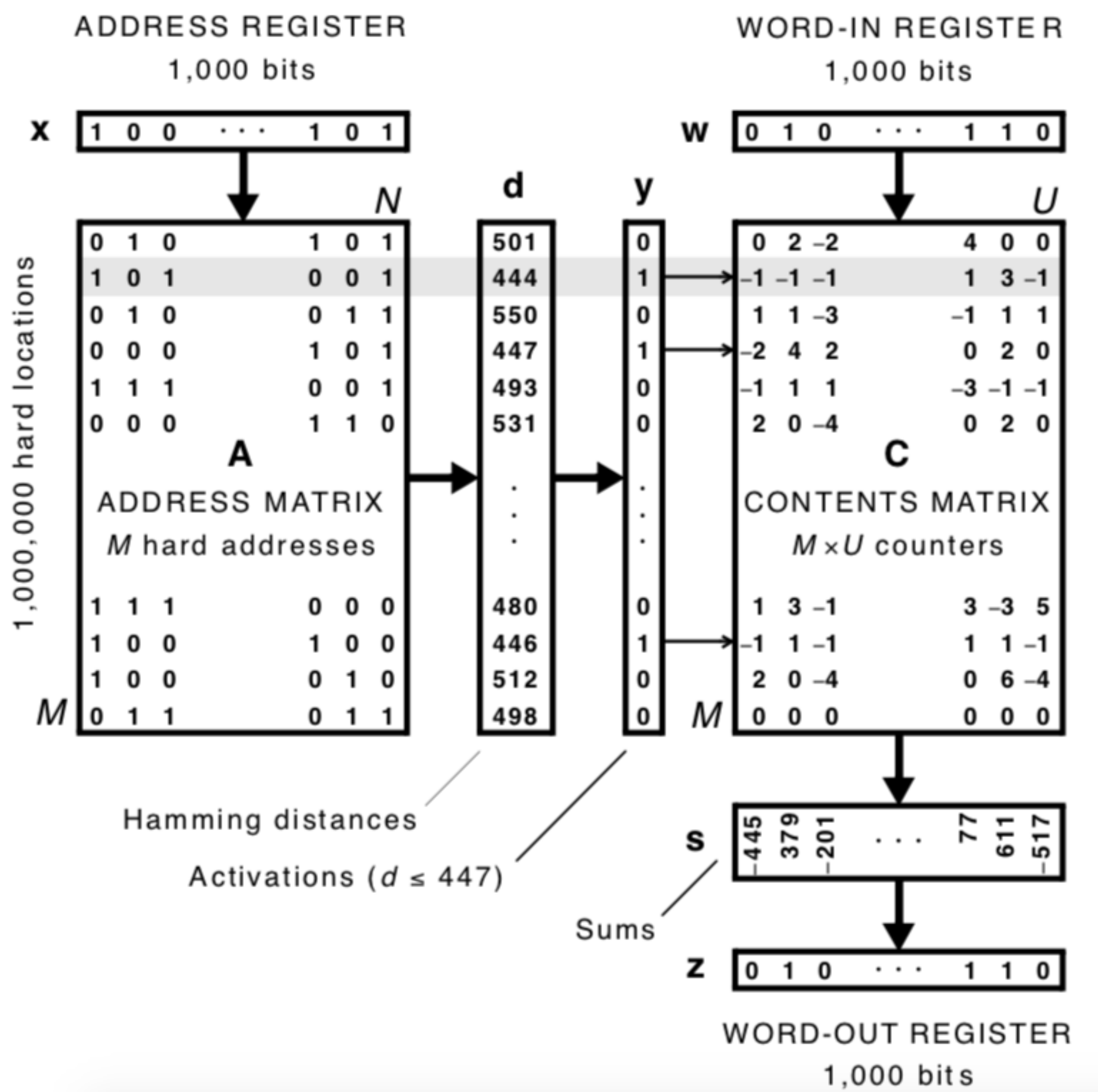
**The architecture is familiar from traditional (von Neumann) computing**

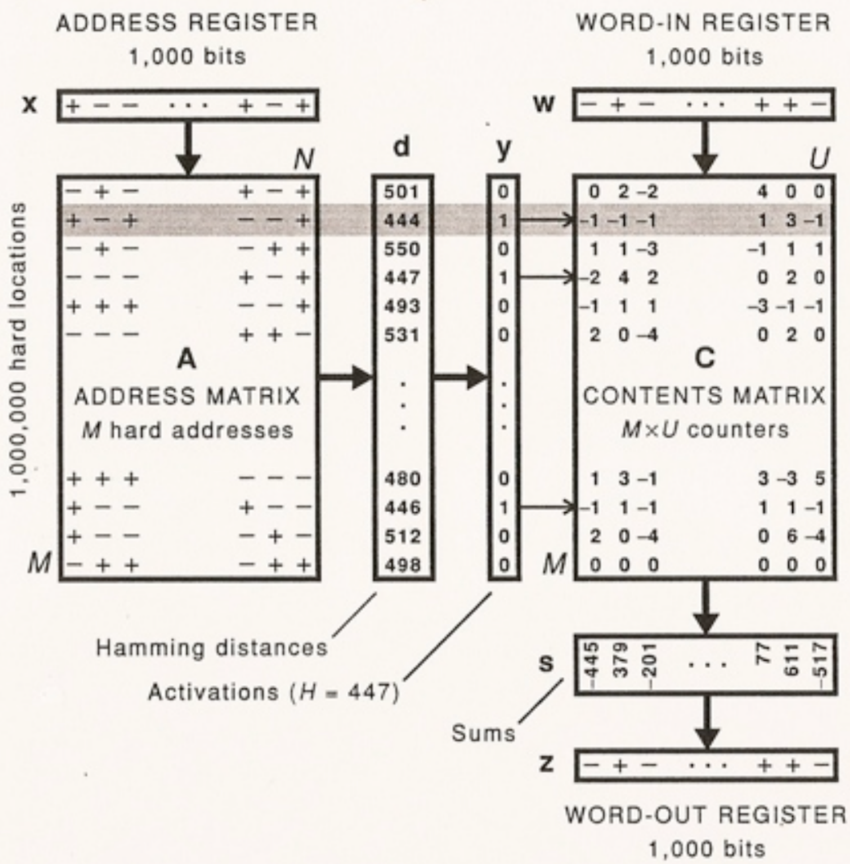
- except that the operations and memory are for high-dimensional vectors

```
      Central Processing Unit (CPU)
          .-----------------------.
 Input -> |     Program Control   | -> Output
          |-----------------------|
          | Arithmetic/Logic Unit |
          |   (ALU)   + * Boolean |
          '-----------------------'
                     ^ |        von Neumann
                     | v         bottleneck
.-----------------------------------------------.
| # # # # # # # # # # # # # # # # # # # # # # # |
| # # # # # # # # # # # # # # # # # # # # # # # |
| # # #                                   # # # |
| # # #    Random Access Memory (RAM)     # # # |
| # # #                                   # # # |
| # # # # # # # # # # # # # # # # # # # # # # # |
| # # # # # # # # # # # # # # # # # # # # # # # |
'-----------------------------------------------'
```

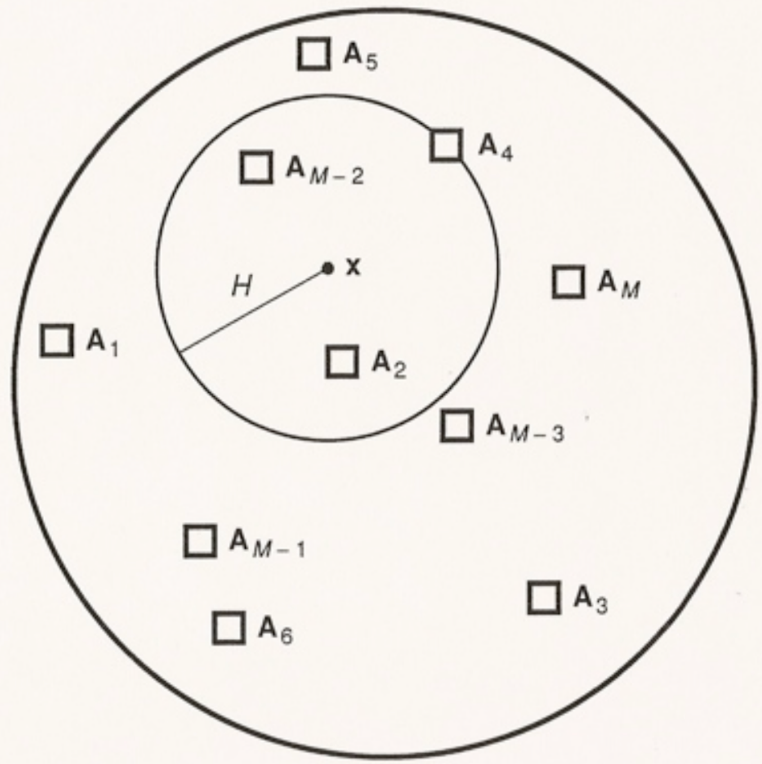
```
          Central Processing Unit (CPU)
             .-----------------------.
    Input -> |     Program Control   | -> Output
             |-----------------------|
            /: Arithmetic/Logic Unit :\
           / :   (ALU)   + * permute :  \
          /  : . . . . . . . . . . . :    \
         /                                  \
        /        "In-memory" computing        \
       /. . . . . . . . . . . . . . . . . . . . .\
       | vec vec vec vec vec vec vec vec vec vec vec |
       | vec vec vec vec vec vec vec vec vec vec vec |
       | vec vec                             vec vec |
       | vec vec      Associative Memory     vec vec |
       | vec vec                             vec vec |
       | vec vec vec vec vec vec vec vec vec vec vec |
       | vec vec vec vec vec vec vec vec vec vec vec |
       '---------------------------------------------'
```

# Sparse Distributed Memory (SDM)

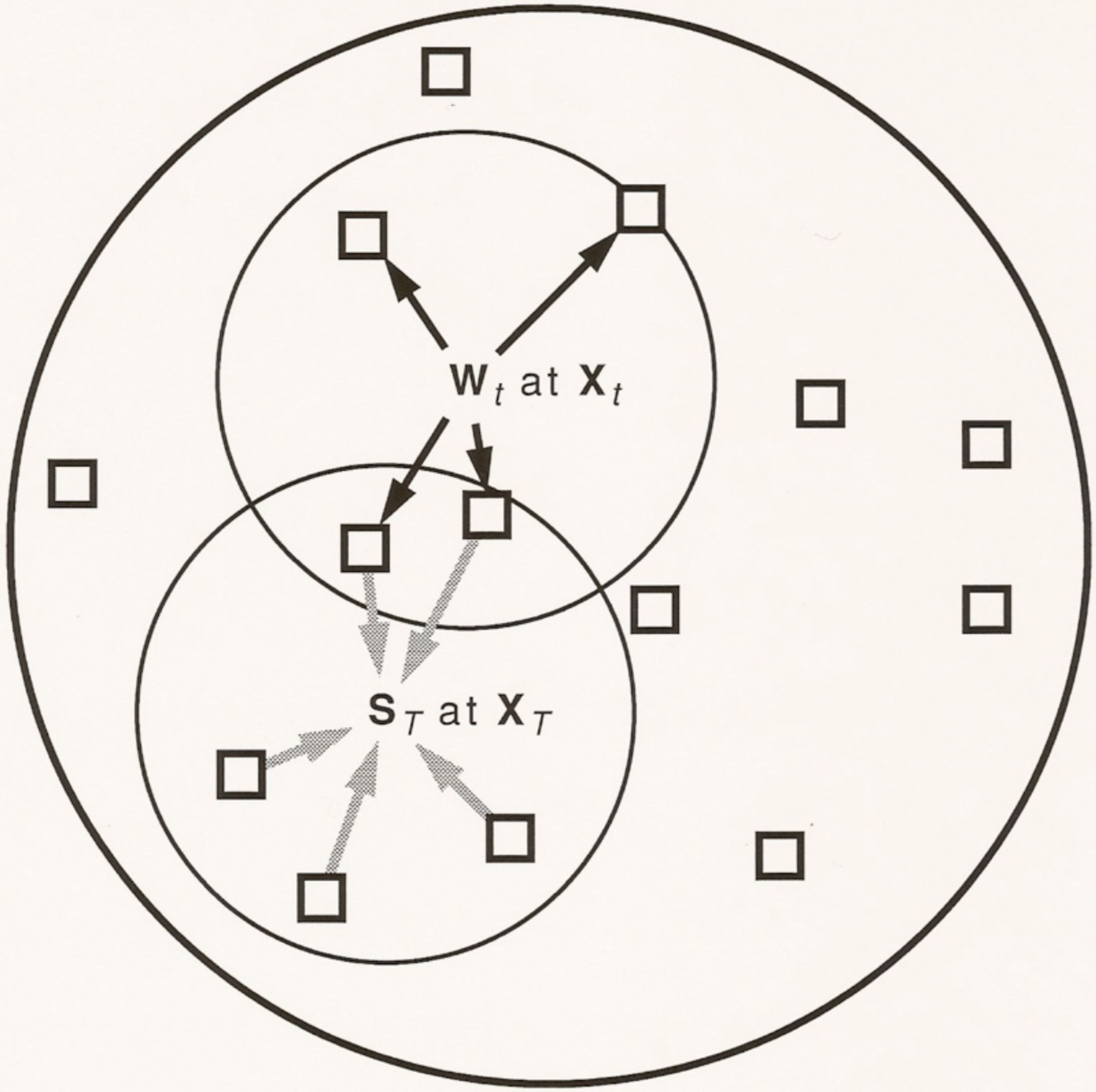





**Sparse Distributed Memory organized as a random-access memory.** The first selected location is shown by shading. Correspondence to the vector-matrix picture of a feed-forward ANN is immediate.

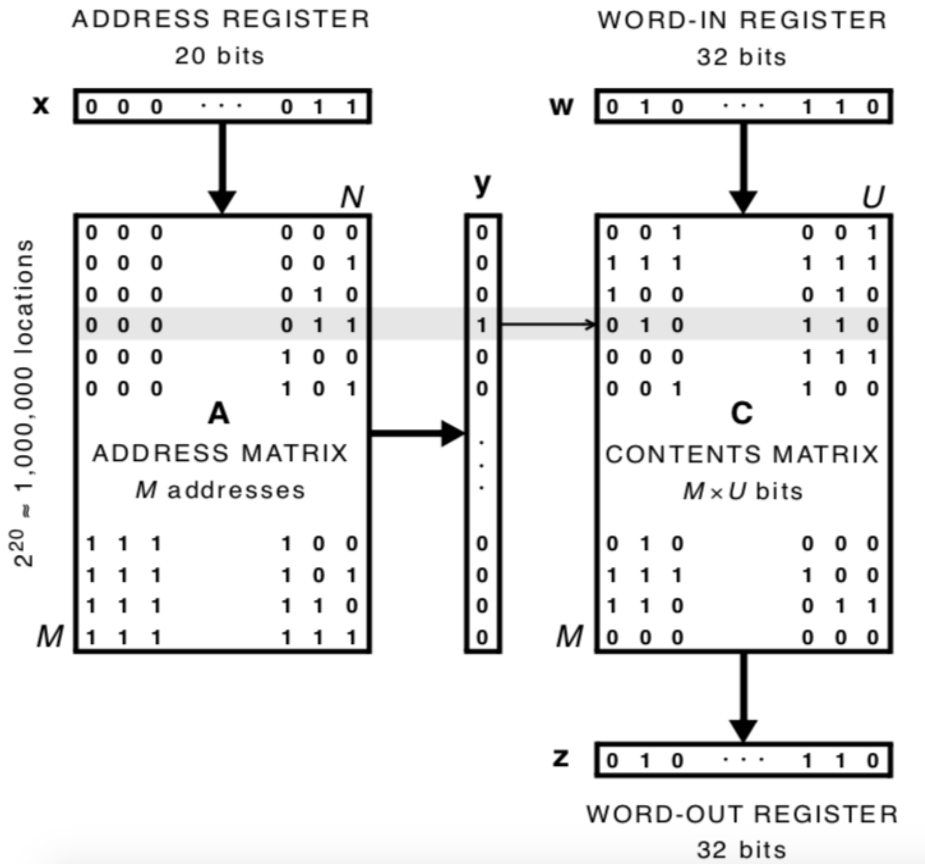


**Address space, hard locations, and the set of locations activated by input pattern x.** *H* is (Hamming) radius of activation.

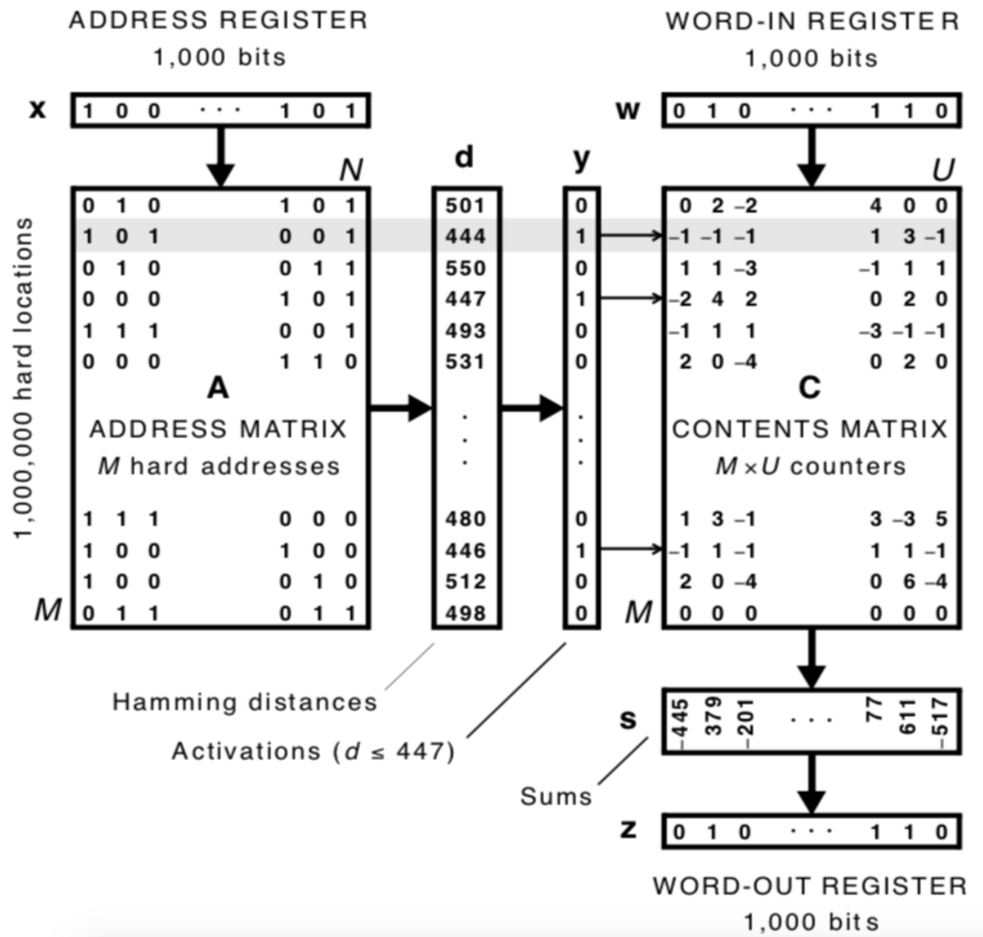



**Activation overlaps as weights for stored words.** Writing $\mathbf{W}_t$ at $\mathbf{X}_t$ stores one copy of $\mathbf{W}_t$ in each location activated by $\mathbf{X}_t$. Reading at $\mathbf{X}_T$ pools the contents of all locations activated by $\mathbf{X}_T$. The pool $\mathbf{S}_T$ will have 2 copies of $\mathbf{W}_t$.

## RAM




## SDM



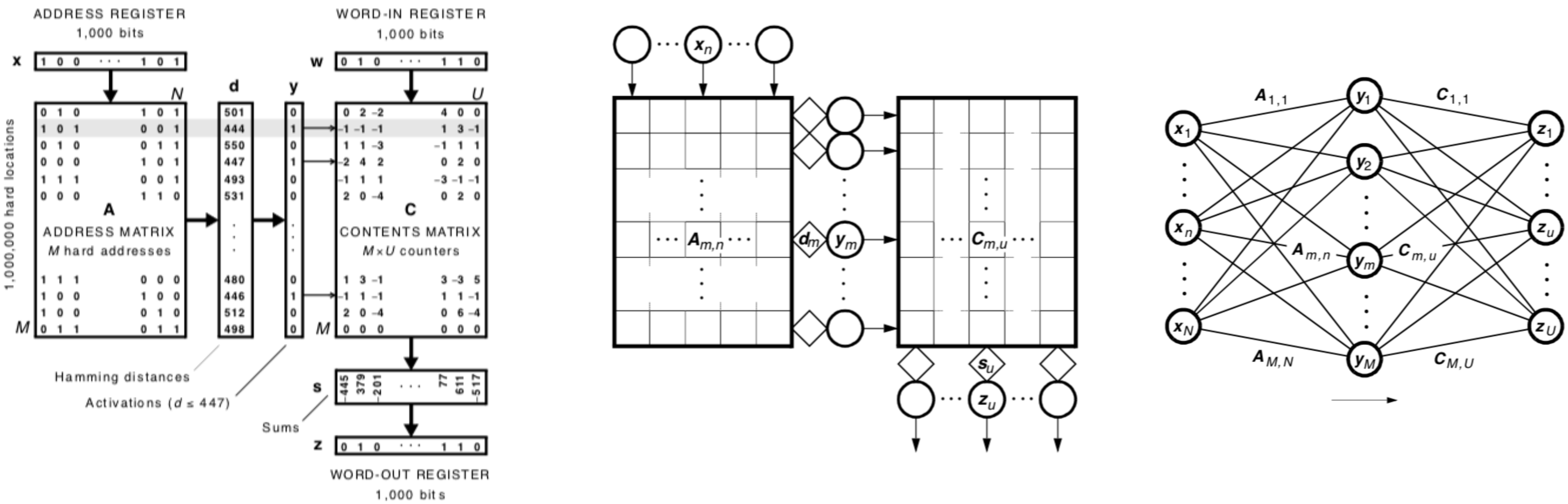

Each read and write activates **one** storage locations

Each read and write activates **multiple** storage locations

# SDM AS A NEURAL NET MORPHING INTO THE CEREBELLUM

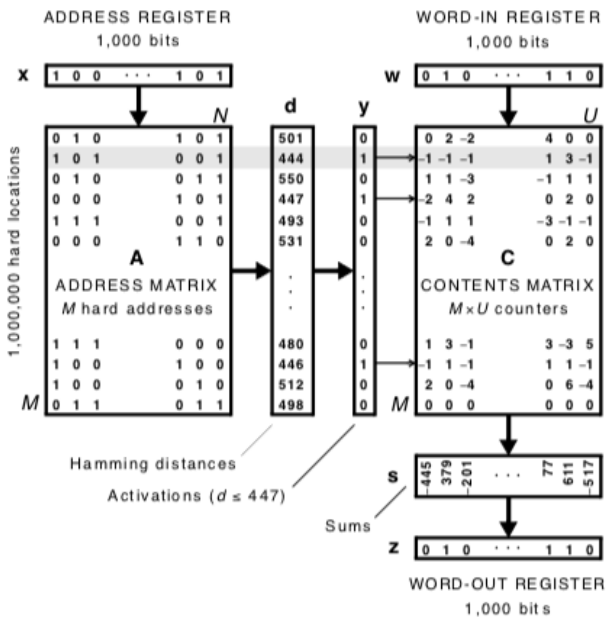
# SDM as a feed-forward Artificial Neural Net

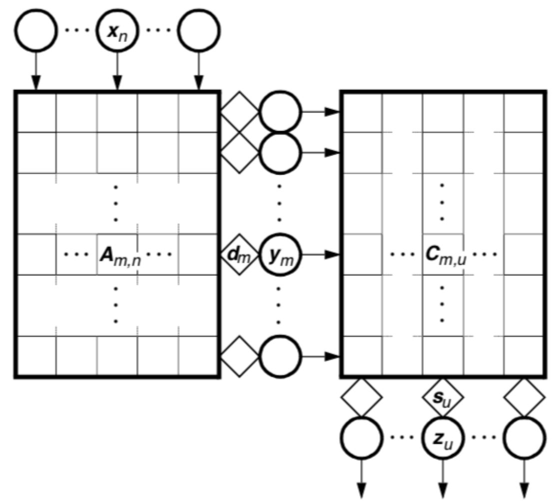
Three depictions of Sparse Distributed Memory

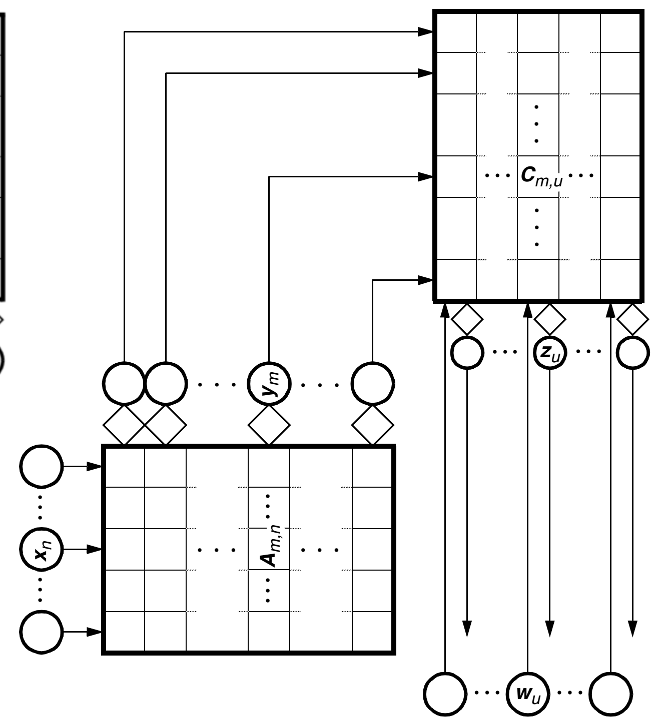



1,000-bit input is *randomly projected* to a million-dimensional, *sparsely* activated hidden layer for computing a 1,000-bit output

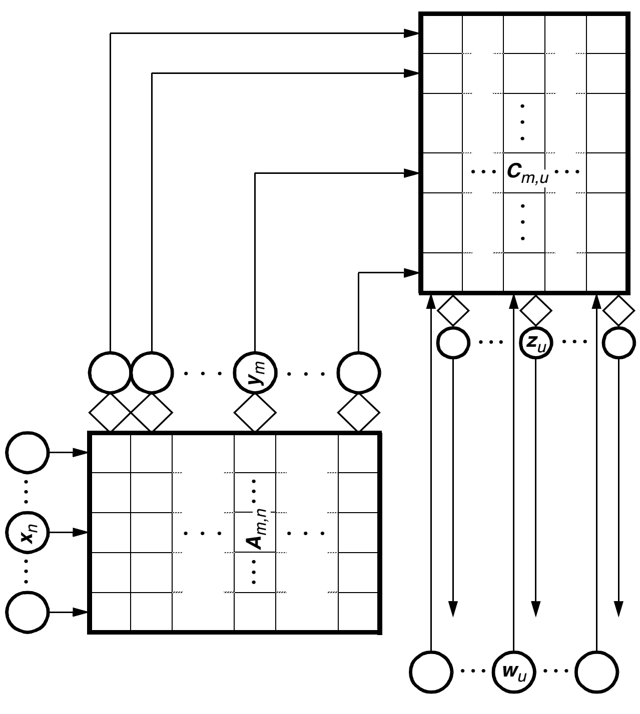
# From SDM to the Cerebellum



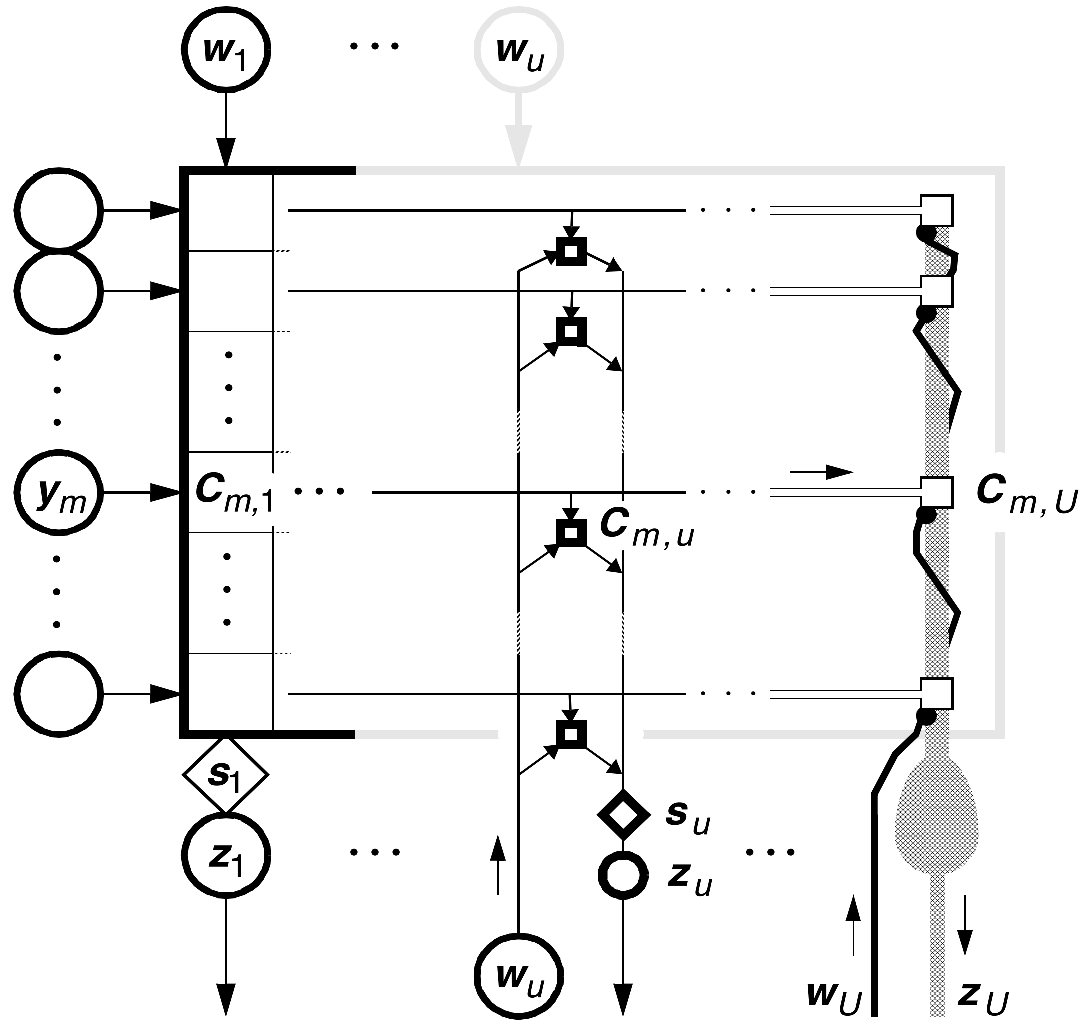



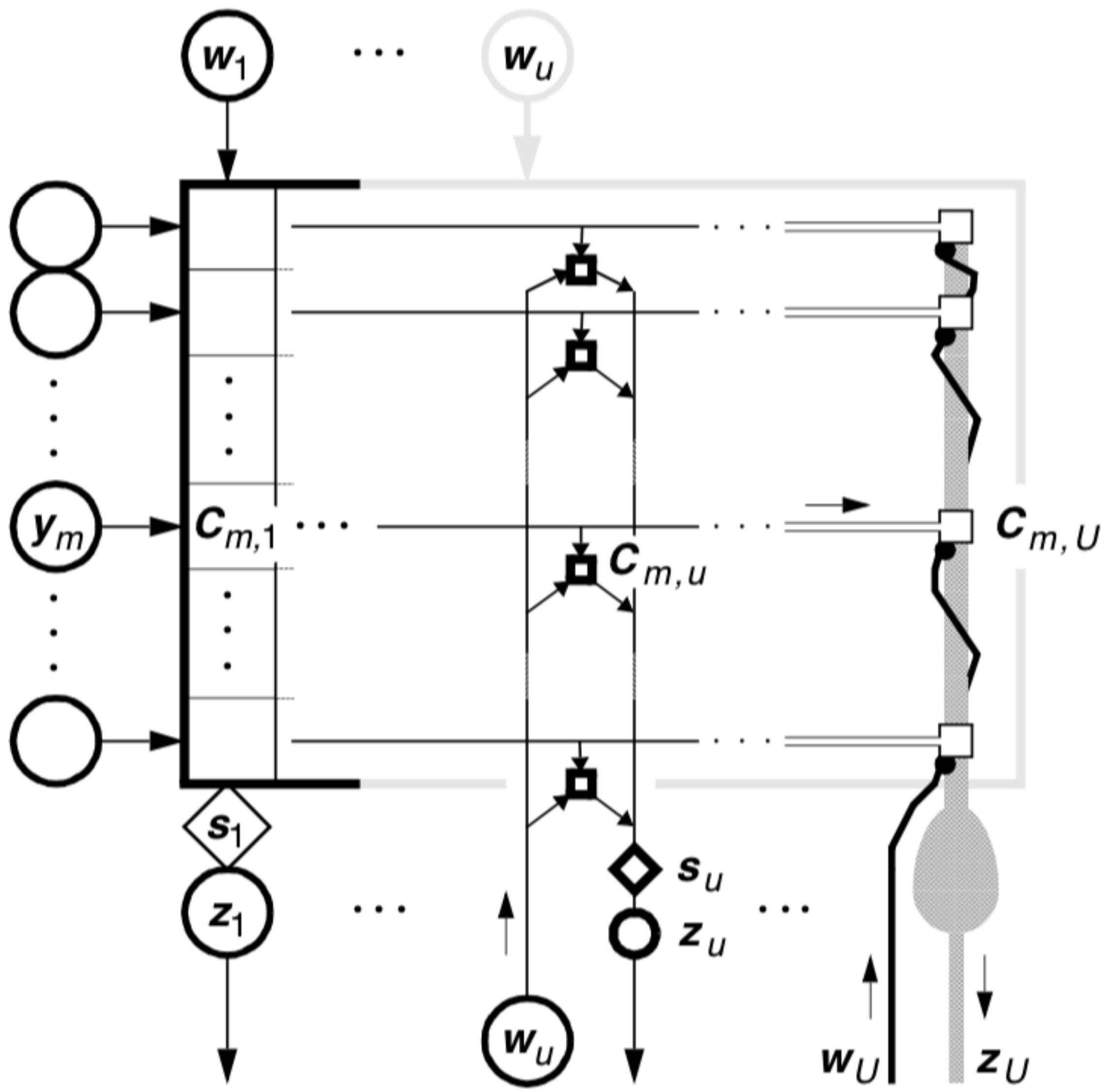



SDM




SDM's contents matrix C






Math, engineering, and neuroscience depictions of modifiable synapses of three Purkinje cells

## Telltale Details

Each Purkinje cell receives input from a **single climbing fiber ...**

... as would be expected of a **training signal**

Granule cells--their **parallel fibers**--represent **memory locations**

Firing of a granule cell activates the location and allows its contents to be read out

The contents are updated when also the climbing fiber fires

. Spike-timing-dependent plasticity

. **Perceptron learning rule**

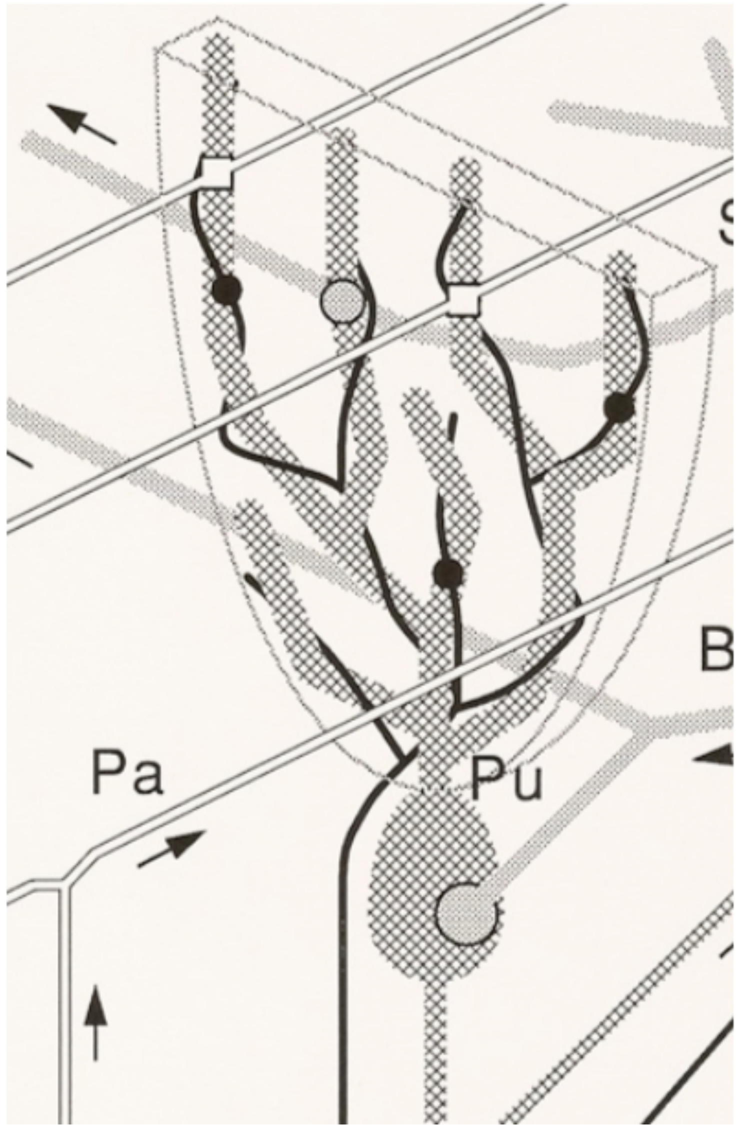
Agrees with models of human memory in **experimental psychology**: Atkinson-Shiffrin Memory Model (Wikipedia)

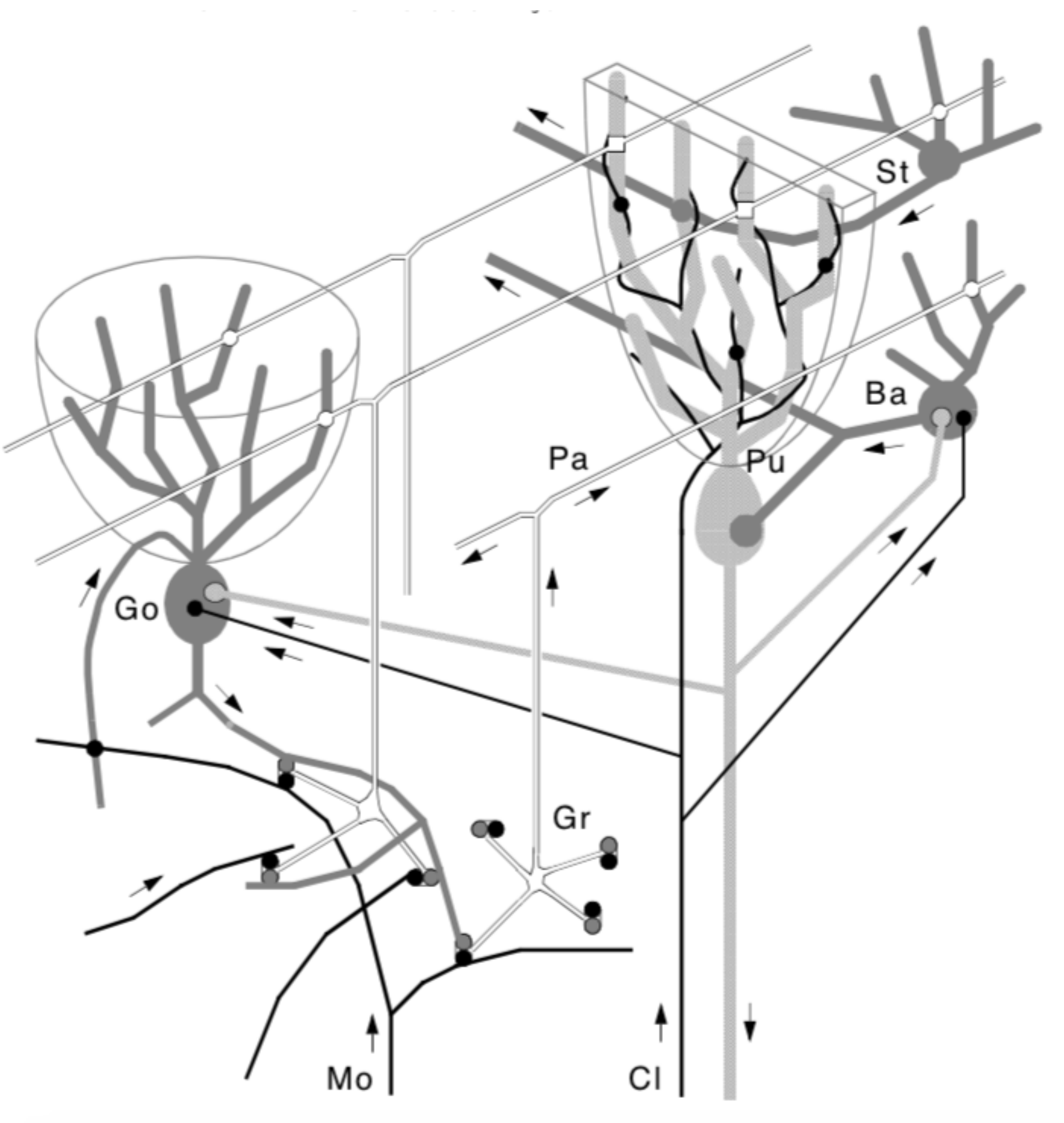
. Short-term working memory

. Long-term data store

. Encoding specificity

# A climbing fiber paired with a Purkinje cell



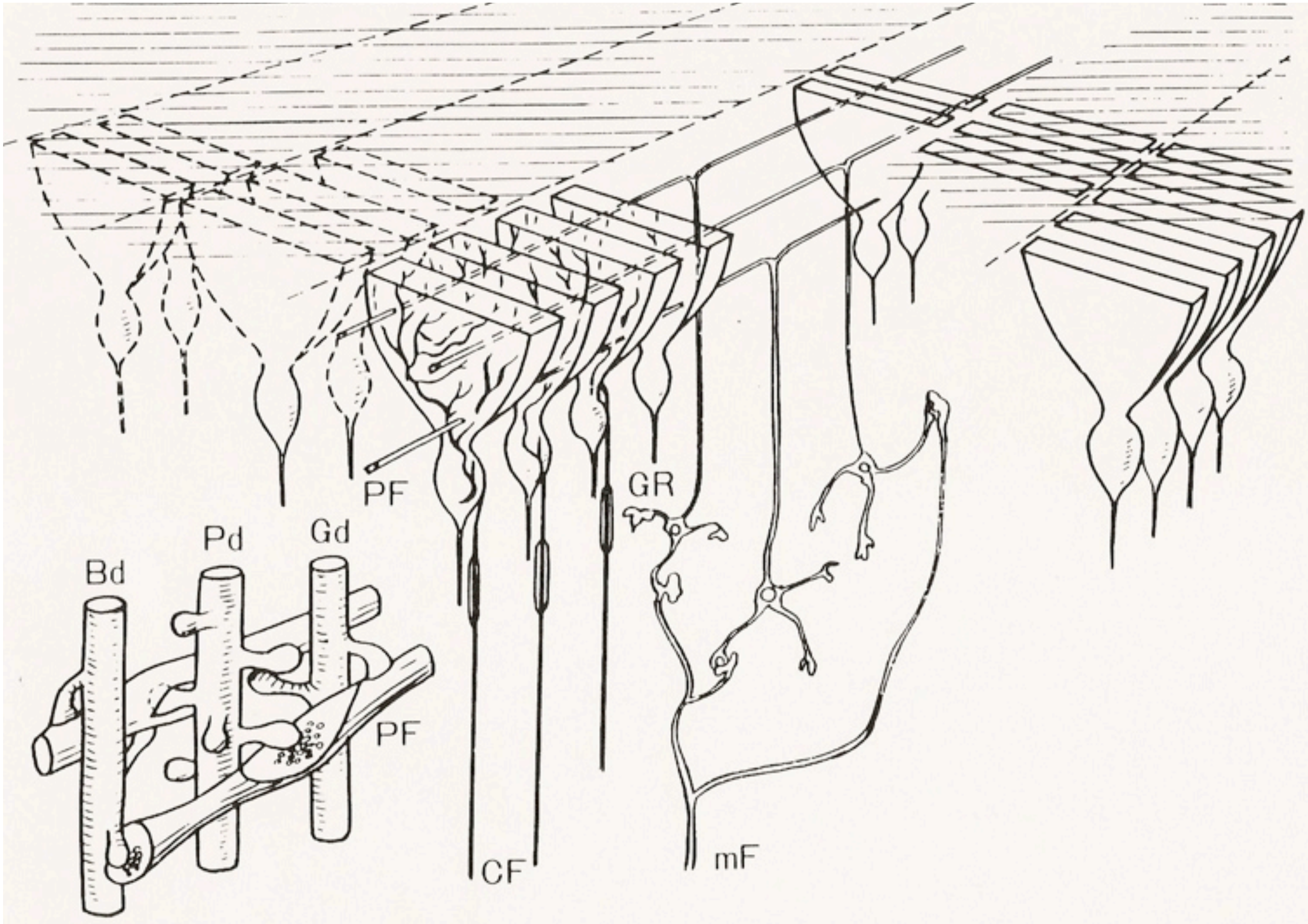


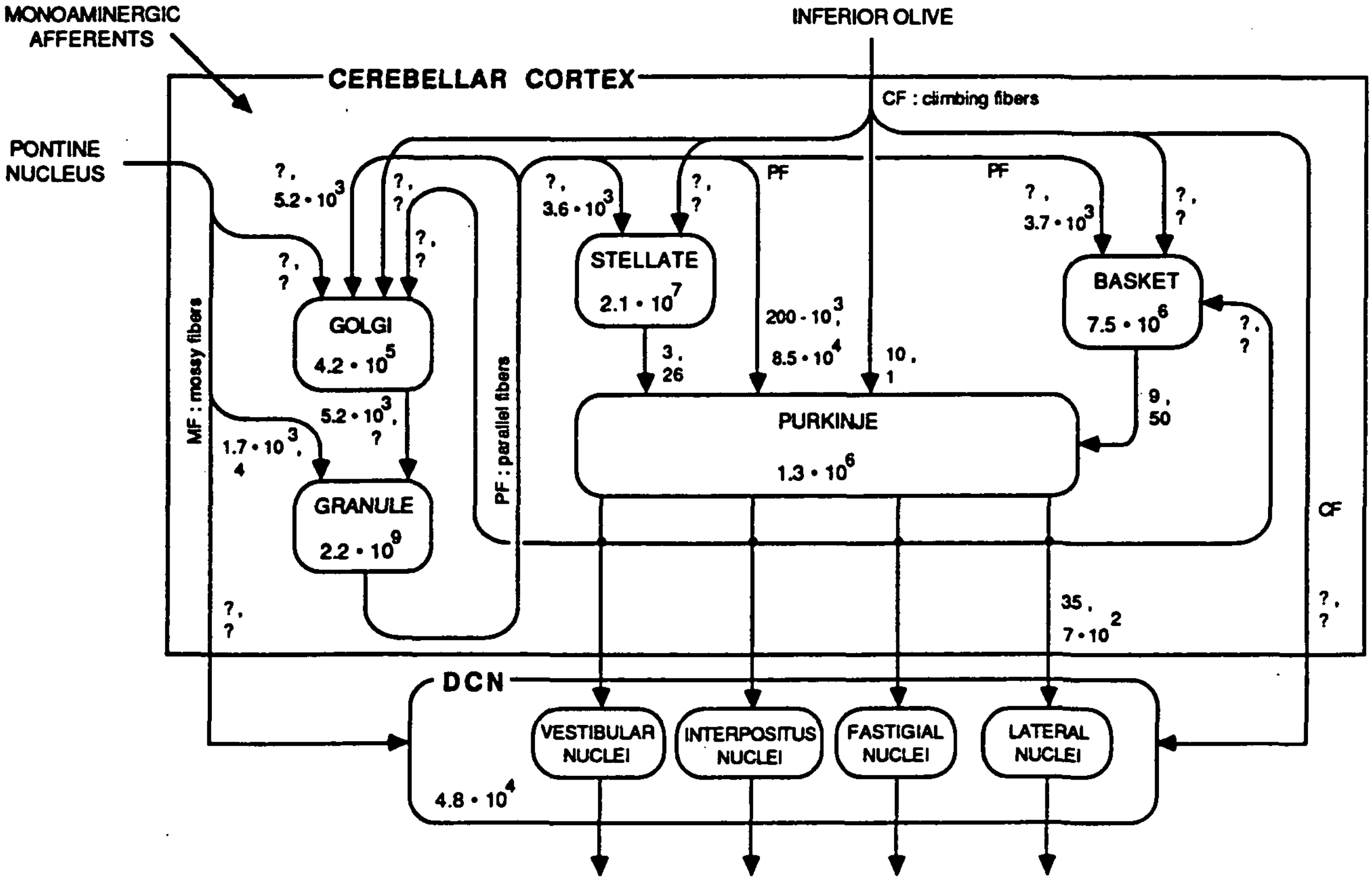

LEGEND

Ba = Basket cell
*Cl = Climbing fiber*
Go = Golgi cell
**Gr = Granule cell**
***Mo = Mossy fiber***
***Pa = Parallel fiber***
**Pu = Purkinje cell**
St = Stellate cell

**Feed-forward circuit in bold**
*Axons in italics*
Golgi, basket and stellate cells are cerebellar interneurons

## Sketch of the cerebellar circuit


PF
GR
Pd
Gd
Bd
PF
CF
mF




Cerebellar interconnect diagram (Loebner, 1989)

# THEnd

Pentti Kanerva <pkanerva@berkeley.edu>
at Bruno's VS265: Computational Neuroscience
3:10-5 PM on November 13, 2024

This work is supported in part by CoCoSys, one of seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA