Large-scale brain models and the Nengo framework

Nhat Le and Sugandha Sharma,

Computational Journal Club 06/12/19

At the end of this tutorial, you will be able to…

1. Describe the three basic principles of the Neural Engineering Framework
   - Describe how neuronal populations can **represent** vector quantities
   - Design suitable decoders to implement **transformations** of the encoded quantities
   - Model the **dynamics** of a dynamical system using interactions between neuronal populations

2. Create simple Nengo models using the Nengo simulator interface

Nengo introduction

- A tool for creating large-scale biologically realistic neural models.
- Developed by the Centre for Theoretical Neuroscience at the University of Waterloo

Eliasmith and Anderson (2003) MIT Press

The Neural Engineering Framework (NEF)

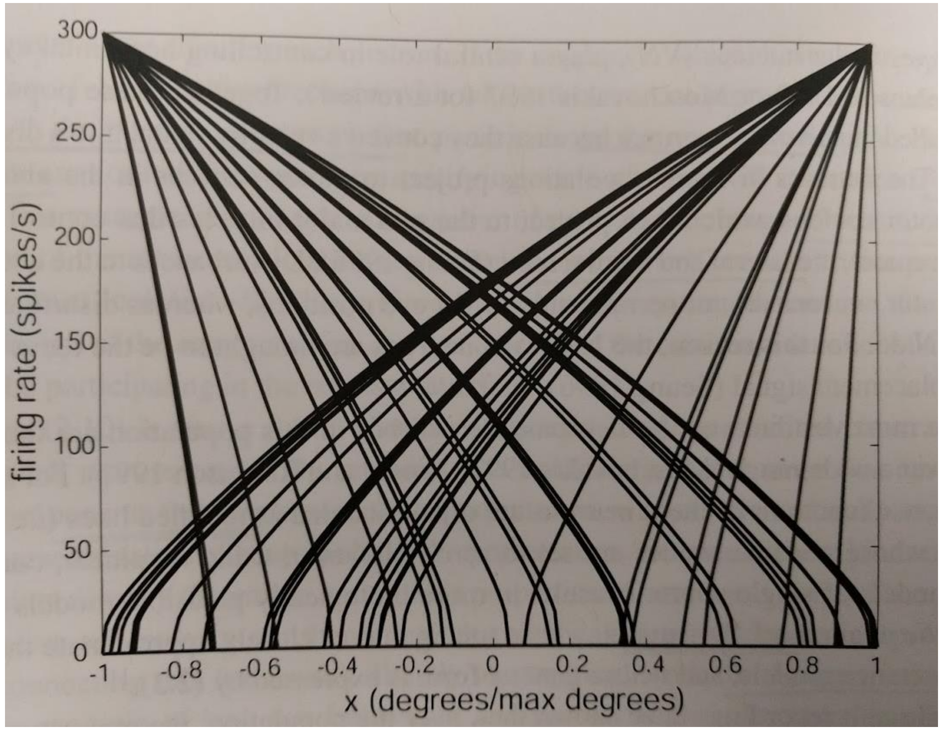
1. Representation
   - Neural representations are nonlinear encodings of vector spaces that can be linearly decoded.

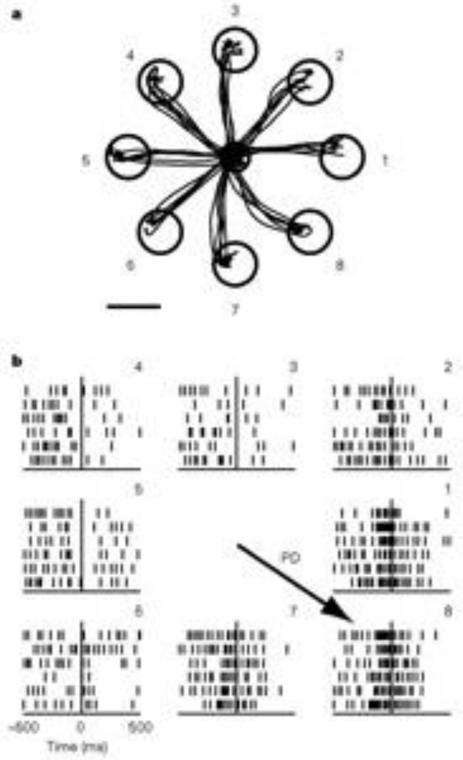
2. Computation
   - Linear decodings of those encodings can compute arbitrary vector functions.

3. Dynamics
   - Neural representations are control theoretic state variables in a nonlinear dynamical system. The dynamics of neurobiological systems can be analysed using control theory.

Neuronal populations represent external variables

Examples:
- Directional tuning in motor cortex (Georgopoulos et al.1984, 1986, 1989, 1993)
- Horizontal eye position (Moschovakis 1997)





Demo 1/3: Representation

In the NEF, information is represented by groups of neurons

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Stimulus vector

$$a_1 = f_1(\mathbf{x})$$ Neuron 1

$$a_2 = f_2(\mathbf{x})$$ Neuron 2

$$\vdots$$

$$a_K = f_K(\mathbf{x})$$ Neuron K

Principle 1: Representation

Encoding

preferred direction of neuron i

vector to be encoded

$$a_i = G_i\left[\alpha_i \langle \mathbf{e}_i, \mathbf{x} \rangle_n + J_i^{bias}\right]$$

non-linearity

Principle 1: Representation

Encoding

preferred direction
of neuron i

vector to be encoded

$$a_i = G_i\left[\alpha_i \langle \mathbf{e}_i, \mathbf{x} \rangle_n + J_i^{bias}\right]$$

non-linearity

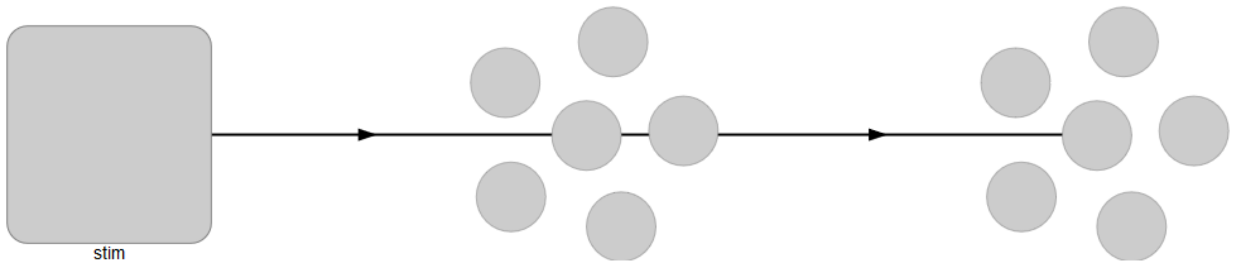
Decoding

$$\hat{x} = \sum_i a_i d_i$$

Q: How to find population decoders d_i ?

A: Minimize $\langle (x - \hat{x})^2 \rangle_x$

Information can be ‘passed’ from one population to another



$\longrightarrow$ **y** = f(**x**)

Represented by a group of neurons

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \xrightarrow{\text{function f}} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Represented by another group of neurons

Stimulus vector

Transformed vector

Question: What should be decoders such that the decoded vector is f(**x**)?

Principle 2: Transformation

Encoding

$$a_i = G_i\left[\alpha_i \langle \mathbf{e}_i, \mathbf{x} \rangle_n + J_i^{bias}\right]$$

Decoding

$$\hat{f}(x) = \sum_i a_i d_i$$

Q: How to find population decoders d_i ?

A: Minimize

$$\langle (f(x) - \hat{f}(x))^2 \rangle_x$$

Demo 2/3: Transformation

Represented quantities can display complex dynamics

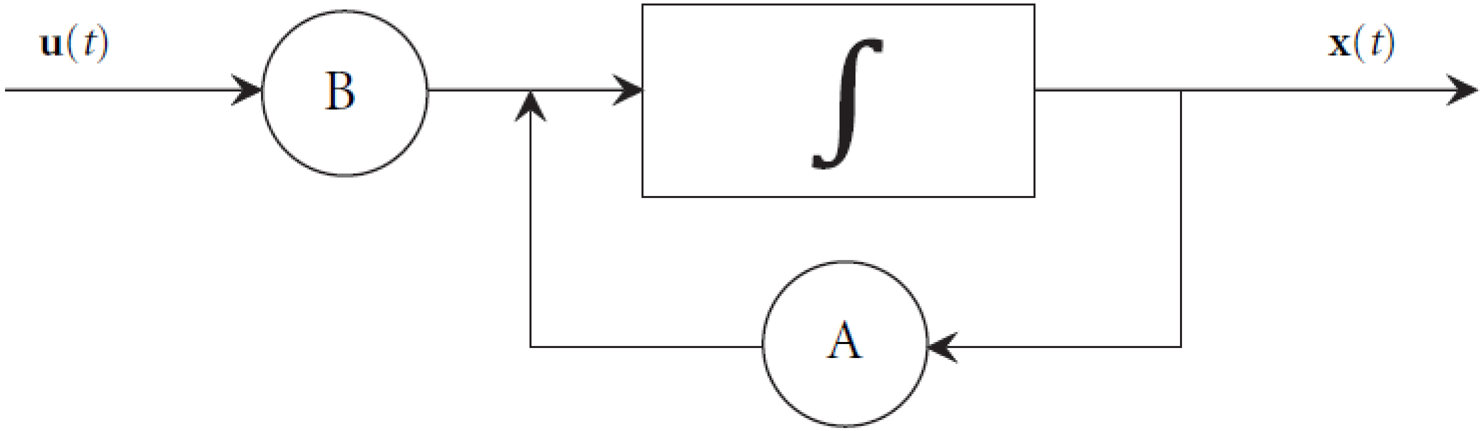
- State transitions
- Locomotion and motor control
- Working memory

Example of dynamics: an integrator

Dynamics can arise from recurrent connections within a population

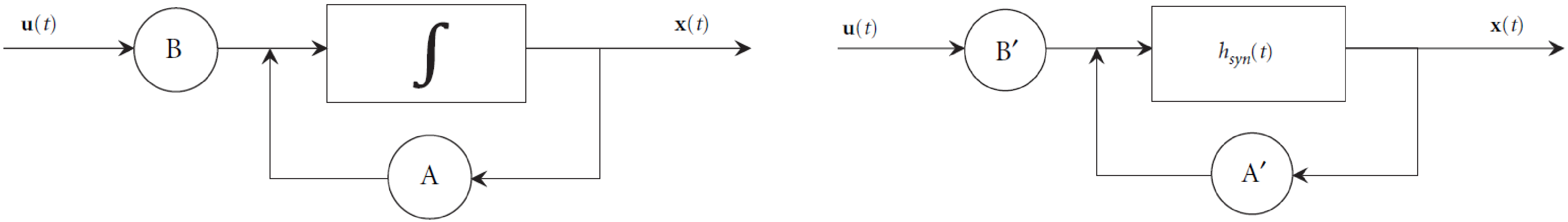
Example:

$$\dot{x}(t) = Ax(t) + Bu(t)$$



Question: Can this dynamical system be implemented by a population of neurons?

Principle 3: Dynamics



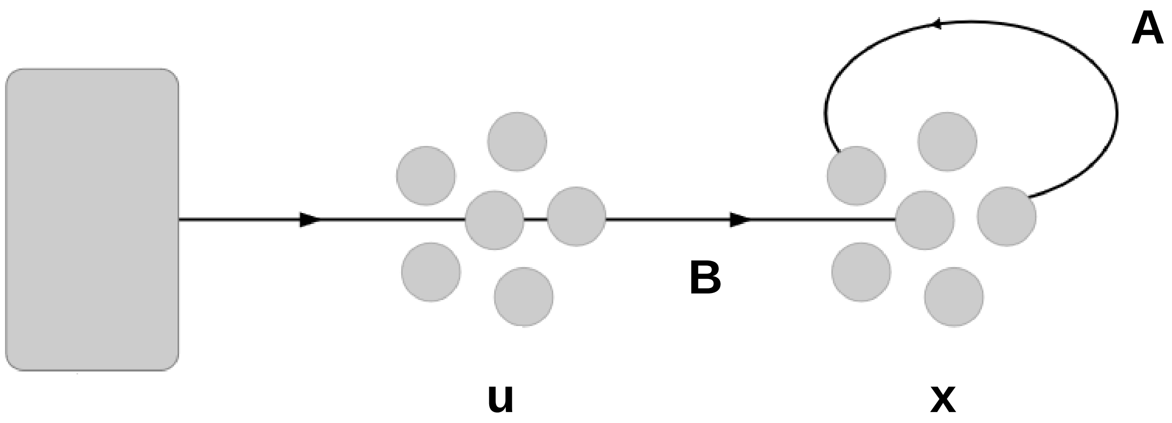
Problem: Neuron dynamics are not perfect integrators, but have dynamics characterized by $h_{syn}(t)$

$$\begin{aligned}\text{For } h(t) &= \frac{1}{\tau}e^{-t/\tau} \\ \mathbf{A}' &= \tau\mathbf{A}+\mathbf{I} \\ \mathbf{B}' &= \tau\mathbf{B}\end{aligned}$$

Demo 3/3: Dynamics

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$\dot{x}(t) = u/\tau - x/\tau$$

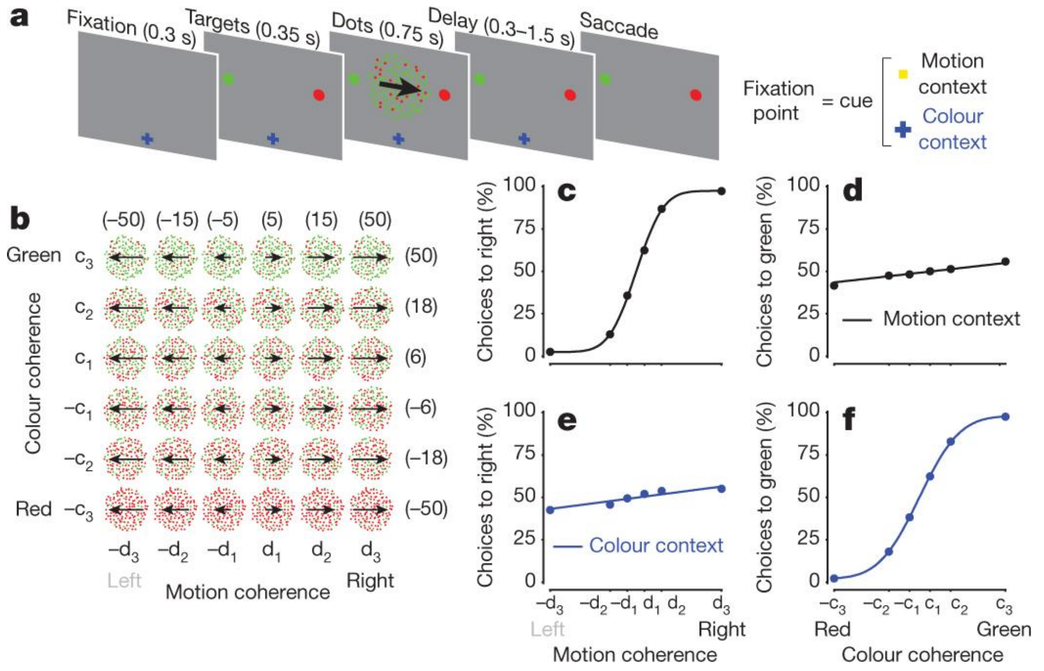


$$\begin{aligned} \mathbf{A}' &= \tau\mathbf{A} + \mathbf{I} \\ \mathbf{B}' &= \tau\mathbf{B} \end{aligned}$$

'synapse'

Demo 3/3: Dynamics

Behavioural task and psychophysical performance.



V Mante *et al.* *Nature* **503**, 78-84 (2013) doi:10.1038/nature12742

nature