Homotopy Theory and Neural Information Networks

Matilde Marcolli
(Caltech, University of Toronto & Perimeter Institute)

Focus Program on New Geometric Methods in Neuroscience
Fields Institute, Toronto, 2020

• based on ongoing joint work with Yuri I. Manin (Max Planck Institute for Mathematics)

• related work:

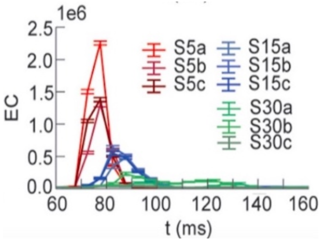
- M. Marcolli, *Gamma Spaces and Information*, Journal of Geometry and Physics, 140 (2019), 26–55.
- Yu.I. Manin and M. Marcolli, *Nori diagrams and persistent homology*, arXiv:1901.10301, to appear in Mathematics of Computer Science.

• building mathematical background for future joint work with Doris Tsao (Caltech neuroscience)

• this work partially supported by FQXi FFF Grant number: FQXi-RFP-1804, SVCF grant number 2018-190467

Motivation N.1: Nontrivial Homology

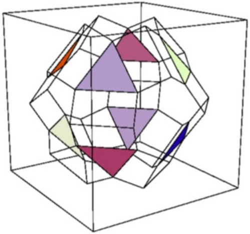
- Kathryn Hess' applied topology group at EPFL: topological analysis of neocortical microcircuitry (Blue Brain Project)



- formation of large number of high dimensional cliques of neurons (complete graphs on N vertices with a directed structure) accompanying response to stimuli
- formation of these structures is responsible for an increasing amount of nontrivial Betti numbers and Euler characteristics, which reaches a peak of topological complexity and then fades
- proposed functional interpretation: this peak of non-trivial homology is necessary for the processing of stimuli in the brain cortex... but why?

Motivation N.2: Computational Role of Nontrivial Homology

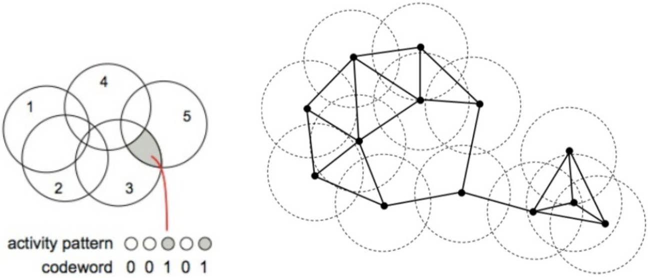
- mathematical theory of concurrent and distributed computing (Fajstrup, Gaucher, Goubault, Herlihy, Rajsbaum, ...)
- initial, final states of processes vertices, $d+1$ mutually compatible initial/final process states d-simplex



- distributed algorithms: simplicial sets and simplicial maps
- certain distributed algorithms require "enough non-trivial homology" to successfully complete their tasks (Herlihy–Rajsbaum)
- this suggests: functional role of non-trivial homology to carry out some concurrent/distributed computation

Motivation N.3: Neural Codes and Homotopy Types

- Carina Curto and collaborators: geometry of stimulus space can be reconstructed *up to homotopy* from binary structure of the neural code



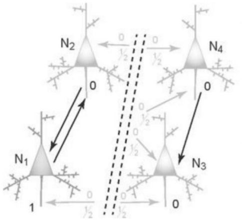
- overlaps between place fields of neurons and the associated simplicial complex of the open covering has the same homotopy type as the stimulus space
- this suggests: the neural code *represents* the stimulus space through homotopy types, hence homotopy theory is a natural mathematical setting

Motivation N.4: Informational and Metabolic Constraints

- neural codes: rate codes (firing rate of a neuron), spike timing codes (timing of spikes), neural coding capacity for given firing rate, output entropy
- metabolic efficiency of a transmission channel ratio $\epsilon = I(X,Y)/E$ of the mutual information of output and input X and energy cost E per unit of time
- optimization of information transmission in terms of connection weights maximizing mutual information $I(X,Y)$
- requirement for homotopy theoretic modelling: need to incorporate constraints on resources and information (mathematical theory of resources: Tobias Fritz and collaborators, categorical setting for a theory of resources and constraints)

Motivation N.5: Informational Complexity

- measures of informational complexity of a neural system have been proposed, such as integrated information: over all splittings $X = A \cup B$ of a system and compute minimal mutual information across the two subsystems, over all such splittings



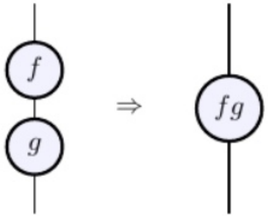
- controversial proposal (Tononi) of integrated information as measure of consciousness (but simple mathematical systems from error correcting codes with very high integrated information!)
- some better mathematical description of organization of neural system over subsystems from which integrated information follows?

Main Idea for a homotopy theoretic modeling of neural information networks

- Want a space (topological) that describes all consistent ways of assigning to a population of neurons with a network of synaptic connections a concurrent/distributed computational architecture ("consistent" means with respect to all possible subsystems)
- Want this space to also keep track of constraints on resources and information and conversion of resources and transmission of information (and information loss) across all subsystems
- Want this description to also keep track of homotopy types (have homotopy invariants, associated homotopy groups): topological robustness
- Why use category theory as mathematical language? because especially suitable to express "consistency over subsystems" and "constraints over resources"
- also categorical language is a main tool in homotopy theory (mathematical theory of concurrent/distributed computing already knows this!)

the language of categories

- category $\mathcal{C}$ two parts of the structure: objects $\mathrm{Obj}(\mathcal{C})$ and morphisms $\mathrm{Mor}_{\mathcal{C}}(A,B)$ between objects $\phi : A \to B$
- composition $\phi \in \mathrm{Mor}_{\mathcal{C}}(A,B)$, $\psi \in \mathrm{Mor}_{\mathcal{C}}(B,C)$ gives $\psi \circ \phi \in \mathrm{Mor}_{\mathcal{C}}(A,C)$, associativity of composition $(\psi \circ \phi) \circ \eta = \psi \circ (\phi \circ \eta)$
- diagrammatic view (string diagrams): objects as wires (possible states) and morphisms as processes



functors between categories

- comparing different categories: $F : \mathcal{C} \to \mathcal{D}$ a map of objects $F(A) \in \mathrm{Obj}(\mathcal{D})$ for $A \in \mathrm{Obj}(\mathcal{C})$ and a compatible map of morphisms $F(\phi) \in \mathrm{Mor}_{\mathcal{D}}(F(A), F(B))$ for $\phi \in \mathrm{Mor}_{\mathcal{C}}(A, B)$
- key idea: morphisms describe how objects can be transformed and functoriality means objects are mapped compatibly with their possible transformations
- comparing functors: natural transformation
- $F, G : \mathcal{C} \to \mathcal{D}$ two functors can be related by $\eta : F \to G$ which means:
    - for all objects $A \in \mathrm{Obj}(\mathcal{C})$ a morphism $\eta_A : F(A) \to G(A)$ in $\mathrm{Mor}_{\mathcal{D}}(F(A), G(A))$
    - for all $\phi \in \mathrm{Mor}_{\mathcal{C}}(A, B)$ compatibility (commutative diagram)

$$\begin{tikzcd} F(A) \arrow[r, "F(\phi)"] \arrow[d, "\eta_A"] & F(B) \arrow[d, "\eta_B"] \\ G(A) \arrow[r, "G(\phi)"] & G(B) \end{tikzcd}$$

Example: Directed Graphs

- category $\mathbf{2}$ has two objects V, E and two morphisms $s, t \in \mathrm{Mor}(E, V)$
- $\mathcal{F}$ category of finite sets: objects finite sets, morphisms functions between finite sets
- a directed graph is a functor $G : \mathbf{2} \to \mathcal{F}$
  - $G(E)$ is the set of edges of the directed graph
  - $G(V)$ is the set of vertices of the directed graph
  - $G(s) : G(E) \to G(V)$ and $G(t) : G(E) \to G(V)$ are the usual source and target maps of the directed graph
- category of directed graphs $\mathrm{Func}(\mathbf{2}, \mathcal{F})$ objects are functors and morphisms are natural transformations

• Note: general philosophical fact about categories: the devil is in the morphisms! (objects are easy, morphisms are usually where the difficulties hide)

Categorical sum

- additional property on a category $\mathcal{C}$, which describes being able to form composite states/systems
- for all A, B objects of $\mathcal{C}$ there is a sum object $A \oplus B$
- this is identified (up to unique isomorphism) by a universal property: there is a unique way of extending morphisms from A and B to morphisms from $A \oplus B$
- there are morphisms $\iota_A : A \to A \oplus B$ and $\iota_B : B \to A \oplus B$ such that, given any morphisms $\phi : A \to C$ and $\psi : B \to C$ there exists a unique morphism $\xi : A \oplus B \to C$ that completes commutative diagram

$$\begin{tikzcd} A \arrow[r, "\iota_A"] \arrow[dr, "\phi"'] & A \oplus B \arrow[d, "\xi"] & B \arrow[l, "\iota_B"'] \arrow[dl, "\psi"] \\ & C & \end{tikzcd}$$

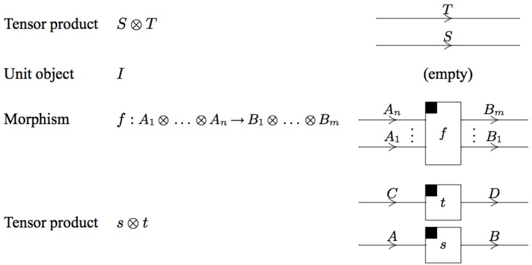
- if you know how to transform two subsystems A, B this determines how to transform their composite system $A \oplus B$

zero object

- an object O that has a unique morphism $O \to A$ to any other object A and a unique morphism $A \to O$ from any other object
- the zero object (when it exists) represents a trivial system
- in particular $O \oplus A \cong A$

unital symmetric monoidal category

- operation of forming composite systems written as $\otimes$
- diagrammatic rules



Example: finite pointed sets

- prototype example of category with sum and zero-object
- objects are finite sets with a choice of a base point (X, x_0)
- morphisms are maps of finite sets that map base point to base point $f : X \to Y$ with $f(x_0) = y_0$
- categorical sum $X \vee Y := X \sqcup Y / x_0 \sim y_0$: union of sets with base points identified
- zero-object single point

• Note: pointed sets are an unavoidable nuisance of homotopy theory! (just think of base point as being there for computational purposes, while "meaning" is attached only to the rest of the set)

Example: category of codes

- C be a $[n,k,d]_2$ binary code of length n with $\#C = q^k$
- category $\mathrm{Codes}_{n,*}$ of pointed codes of length n
  - objects codes that contain 0-word $c_0 = (0,0,\dots,0)$
  - exclude code consisting only of constant words $c_0 = (0,0,\dots,0)$ and $c_1 = (1,1,1,\dots,1)$ (for reasons of non-trivial information)
  - morphisms $f : C \to C'$ functions mapping the 0-word to itself (don't require maps of ambient $\mathbb{F}_2^n$)
  - sum as for pointed sets $C \vee C'$ (glued along the zero-word)
  - zero-object: code consisting only of the zero word
  - role of zero-word is like reference point (for neural code, baseline when no activity detected)

• Note: in coding theory often other form of categorical sum (decomposable codes), but changes code length n

$$C \oplus C' := \{(c,c') \in \mathbb{F}_2^{n+n'} \mid c \in C,\ c' \in C'\}$$

neural codes

- $T > 0$ time interval of observation, subdivided into some basic units of time, Δt
- code length $n = T/\Delta t$: number of basic time intervals considered
- number of nontrivial code words: neurons oberved
- each code word: firing pattern of that neuron, digit 1 for each time intervals Δt that contained a spike and 0 otherwise
- zero-word baseline of no activity (for comparison)
- a neural code for N neurons is a sum $C_1 \vee \cdots \vee C_N$ with $C_i = \{c_0, c\}$ with zero-word c_0 and firing pattern c of i-th neuron

Example: finite probabilities and fiberwise measures

- objects (X, P) finite pointed sets (X, x_0) with probability measure P (with $P(x_0) > 0$)
- morphisms $\phi : (X, P) \to (Y, Q)$ pairs $\phi = (f, \lambda)$ pointed function $f : (X, x_0) \to (Y, y_0)$ with $f(\mathrm{supp}(P)) \subseteq \mathrm{supp}(Q)$ and weights $\lambda_y(x) \in \mathbb{R}_+$ for $x \in f^{-1}(y)$

$$P(A) = \sum_{y \in f(A)} \sum_{x \in f^{-1}(A)} \lambda_y(x) \in Q(y)$$

- category $\mathcal{P}_f$ of finite probabilities with fiberwise measures

Probability measure associated to a code

- binary code C of length n
- $b(c)$ number of digits equal to 1 in the word c (Hamming distance to the reference zero-word c_0)
- probability distribution

$$P_C(c) = \begin{cases} \frac{b(c)}{n(\#C-1)} & c \neq c_0 \\ 1 - \sum_{c' \neq c_0} \frac{b(c')}{n(\#C-1)} & c = c_0 \end{cases}$$

- mapping $C \mapsto P_C$ determines a functor $P : \mathrm{Codes}_{n,*} \to \mathcal{P}_f$
- morphisms: $\lambda_{f(c)}(c) = \frac{P_C(c)}{P_{C'}(f(c))}$ is ok because only code word with $b(c) = 0$ is 0-word c_0 and $b(c) = n$ only for the word c_1 (assuming codes do not contain only c_0 and c_1)
- the probability P_C keeps track of the information transmitted by the code C

Example: weighted codes

- category of weighted binary codes $\mathcal{W}\mathrm{Codes}_{n,*}$
- objects pairs (C,ω) of a code C of length n containing zero-word c_0 and function $\omega : C \to \mathbb{R}$ assigning (signed) weight to each code word, with $\omega(c_0) = 0$
- morphisms $\phi = (f,\lambda) : (C,\omega) \to (C',\omega')$ with $f : C \to C'$ mapping the zero-word to itself and $f(\mathrm{supp}(\omega)) \subset \mathrm{supp}(\omega')$ and weights $\lambda_{c'}(c)$ for $c \in f^{-1}(c')$
- sum $(C,\omega) \oplus (C',\omega') = (C \vee C', \omega \vee \omega')$ with $\omega \vee \omega'|_C = \omega$ and $\omega \vee \omega'|_{C'} = \omega'$
- zero object $(\{c_0\}, 0)$

Example: concurrent/distributed computing architectures

- category of transition systems
  - G. Winskel, M. Nielsen, *Categories in concurrency*, in "Semantics and logics of computation (Cambridge, 1995)", pp. 299–354, Publ. Newton Inst., 14, Cambridge Univ. Press, 1997.
- models of computation that involve parallel and distributed processing
- objects $\tau = (S, \iota, \mathcal{L}, \mathcal{T})$ with S set of possible states of the system, ι initial state, $\mathcal{L}$ set of labels, $\mathcal{T}$ set of possible transitions, $\mathcal{T} \subseteq S \times \mathcal{L} \times S$ (specified by initial state, label of transition, final state)
- directed graph with vertex set S and with set of labelled directed edges $\mathcal{T}$

- $\mathrm{Mor}_{\mathcal{C}}(\tau, \tau')$ of transition systems pairs (σ, λ), function $\sigma : S \to S'$ with $\sigma(\iota) = \iota'$ and (partially defined) function $\lambda : \mathcal{L} \to \mathcal{L}'$ of labeling sets such that, for any transition $s_{in} \stackrel{\ell}{\to} s_{out}$ in $\mathcal{T}$, if $\lambda(\ell) \in \mathcal{L}'$ is defined, then $\sigma(s_{in}) \stackrel{\lambda(\ell)}{\to} \sigma(s_{out})$ is a transition in $\mathcal{T}'$
- categorical sum

$$(S, \iota, \mathcal{L}, \mathcal{T}) \oplus (S', \iota', \mathcal{L}', \mathcal{T}') = (S \times \{\iota'\} \cup \{\iota\} \times S', (\iota, \iota'), \mathcal{L} \cup \mathcal{L}', \mathcal{T} \sqcup \mathcal{T}')$$

$$\mathcal{T} \sqcup \mathcal{T}' := \{(s_{in}, \ell, s_{out}) \in \mathcal{T}\} \cup \{(s'_{in}, \ell', s'_{out}) \in \mathcal{T}'\}$$

where both sets are seen as subsets of

$$(S \times \{\iota'\} \cup \{\iota\} \times S') \times (\mathcal{L} \cup \mathcal{L}') \times (S \times \{\iota'\} \cup \{\iota\} \times S')$$

- zero object is given by the stationary single state system $S = \{\iota\}$ with empty labels and transitions

Categories of Resources

- mathematical theory of resources
  - B. Coecke, T. Fritz, R.W. Spekkens, *A mathematical theory of resources*, Information and Computation 250 (2016), 59–86. [arXiv:1409.5531]
- Resources modelled by a symmetric monoidal category $(\mathcal{R}, \circ, \otimes, \mathbb{I})$
- objects $A \in \mathrm{Obj}(\mathcal{R})$ represent resources, product $A \otimes B$ represents combination of resources, unit object $\mathbb{I}$ empty resource
- morphisms $f : A \to B$ in $\mathrm{Mor}_{\mathcal{R}}(A, B)$ represent possible conversions of resource A into resource B
- convertibility of resources when $\mathrm{Mor}_{\mathcal{R}}(A, B) \neq \emptyset$

Example of optimization of resources: adjoint functors

- a functor $\rho : \mathcal{C} \to \mathcal{R}$ that assigns resources to (computational) systems
- existence of a left adjoint $\beta : \mathcal{R} \to \mathcal{C}$ such that

$$\mathrm{Mor}_{\mathcal{C}}(\beta(A), C) \simeq \mathrm{Mor}_{\mathcal{R}}(A, \rho(C)), \quad \forall C \in \mathrm{Obj}(\mathcal{C}), \; A \in \mathrm{Obj}(\mathcal{R})$$

- adjoint functor is a solution to an optimization problem
- assignment $A \mapsto \beta(A)$ optimal way of associating a computational system $\beta(A)$ in the category $\mathcal{C}$ to a given constraints on available resources, encoded by fixing $A \in \mathrm{Obj}(\mathcal{R})$

- system $\beta(A)$ constructed from resources A: some of resources A used for the manufacturing of $\beta(A)$ so one expects conversion from A to remaining resources available to system $\beta(A)$, namely $\rho(\beta(A))$ (left rather than a right adjoint)
- Freyd's adjoint functor theorem (condition for existence):
  solution set $\{C_j\}_{j\in J}$ (systems optimal for resources A)
  - with morphisms $u_j : A \to \rho(C_j)$
  - for any system $C \in \mathrm{Obj}(\mathcal{C})$ for which there is a possible conversion of resources $u : A \to \rho(C)$ in $\mathrm{Mor}_{\mathcal{R}}(A, \rho(C))$, there is one of the systems C_j and a modification of systems $\phi : C_j \to C$ in $\mathrm{Mor}_{\mathcal{C}}(C_j, C)$ such that conversion of resources $u : A \to \rho(C)$ factors through the system C_j, namely $u = \rho(\phi) \circ u_j$
  - solution set C_i optimal sytems from which any other system that uses less resources than A can be obtained via allowed modifications (morphisms)

Summing functors

- $\mathcal{C}$ a category with sum and zero-object (binary codes, transition systems, resources, etc)
- (X, x_0) a pointed finite set and $\mathcal{P}(X)$ a category with objects the pointed subsets $A \subseteq X$ and morphisms the inclusions $j : A \subseteq A'$
- a functor $\Phi_X : \mathcal{P}(X) \to \mathcal{C}$ summing functor if

$$\Phi_X(A \cup A') = \Phi_X(A) \oplus \Phi_X(A') \quad \text{when} \quad A \cap A' = \{x_0\}$$

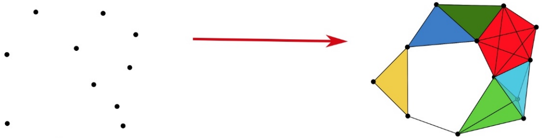
  and $\Phi_X(\{x_0\})$ is zero-object of $\mathcal{C}$
- $\Sigma_{\mathcal{C}}(X)$ category of summing functors $\Phi_X : \mathcal{P}(X) \to \mathcal{C}$, morphisms are natural transformations

• Key idea: a summing functor is a *consistent assignment* of *resources* of type $\mathcal{C}$ to *all subsystems* of X so that a combination of independent subsystems corresponds to combined resources

• $\Sigma_{\mathcal{C}}(X)$ parameterizes all possible such assignments

Segal's Gamma Spaces

- construction introduced in homotopy theory in the '70s: a general construction of (connective) *spectra* (generalized homology theories)
- a Gamma space is a functor $\Gamma : \mathcal{F} \to \Delta$ from finite (pointed) sets to (pointed) simplicial sets



- a category $\mathcal{C}$ with sum and zero-object determines a Gamma space $\Gamma_{\mathcal{C}} : \mathcal{F} \to \Delta$
  - for a finite set X take category of summing functors $\Sigma_{\mathcal{C}}(X)$ and simplicial set given by nerve $\mathcal{N}(\Sigma_{\mathcal{C}}(X))$ of this category

meaning of this construction

- nerve of a category: vertices are objects of the category, edges are morphisms between objects, any chain of compositions of morphisms gives a simplex: example, a 2-simplex

$$\begin{tikzcd} & Z \\ X \arrow[ur, "g\circ f"] \arrow[r, "f"'] & Y \arrow[u, "g"'] \end{tikzcd}$$

  (faces and degeneracies: compose successive morphisms in a chain, insert identity morphism)

- nerve $\mathcal{N}(\Sigma_{\mathcal{C}}(X))$ of category of summing functors organizes all assignments of $\mathcal{C}$-resources to X-subsystems and their transformations into a single topological structure that keeps track of relations between them (morphisms of $\Sigma_{\mathcal{C}}(X)$ and their compositions become simplexes of the nerve)

Systems organized according to networks

- instead of finite set X want a directed graph (network) and its subsystems
- directed graph as functor $G : \mathbf{2} \to \mathcal{F}$ and functorial assignment $X \mapsto \Sigma_{\mathcal{C}}(X)$
- $\Sigma_{\mathcal{C}}(E_G)$ summing functors $\Phi_E : \mathcal{P}(E_G) \to \mathcal{C}$ for sets of edges and $\Sigma_{\mathcal{C}}(V_G)$ summing functors $\Phi_V : \mathcal{P}(V_G) \to \mathcal{C}$ for sets of vertices
- source and target maps $s, t : E_G \to V_G$ transform summing functors $\Phi_E \in \Sigma_{\mathcal{C}}(E_G)$ to summing functors in $\Sigma_{\mathcal{C}}(V_G)$

$$\Phi^s_{V_G}(A) := \Phi_{E_G}(s^{-1}(A)) \qquad \Phi^t_{V_G}(A) := \Phi_{E_G}(t^{-1}(A))$$

  assigns to a set of vertices $\mathcal{C}$-resources of in/out edges
- categorical statement: source and target maps $s, t : E_G \to V_G$ determine functors between categories $\Sigma_{\mathcal{C}}(E_G)$ and $\Sigma_{\mathcal{C}}(V_G)$ of summing functors, hence map between their nerves

Expressing constraints and optimization in categorical form

- limits and colimits in categories
    - diagram $F : \mathcal{J} \to \mathcal{C}$ and cone N, limit is "optimal cone" (dual version for colimits)

$$\begin{tikzcd}
 & N \arrow[d, dashed, "u"] \arrow[ddl, bend right, "\psi_X"'] \arrow[ddr, bend left, "\psi_Y"] & \\
 & L \arrow[dl, "\phi_X"] \arrow[dr, "\phi_Y"'] & \\
F(X) \arrow[rr, "F(f)"'] & & F(Y)
\end{tikzcd}$$

    - special cases of limits and colimits: equalizers, coequalizers
- Example: thin categories $(S, \leq)$ set of objects S and one morphism $s \to s'$ when $s \leq s'$
    - diagram in $(S, \leq)$ is selection of a subet $A \subset S$
    - limits and colimits greatest lower bounds and least upper bounds for subsets $A \subseteq S$
- Key idea: functors compatible with limits and colimits describe constrained optimization

Conservation laws at vertices

- source and target functors $s, t : \Sigma_{\mathcal{C}}(E_G) \rightrightarrows \Sigma_{\mathcal{C}}(V_G)$
- equalizer category $\Sigma_{\mathcal{C}}(G)$ with functor $\iota : \Sigma_{\mathcal{C}}(G) \to \Sigma_{\mathcal{C}}(E_{G^*})$ such that $s \circ \iota = t \circ \iota$ with universal property

$$\begin{tikzcd}
\Sigma_{\mathcal{C}}(G) \arrow[r, "\iota"] & \Sigma_{\mathcal{C}}(E_G) \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] & \Sigma_{\mathcal{C}}(V_G) \\
\mathcal{A} \arrow[u, "\exists u"] \arrow[ur, "q"'] & &
\end{tikzcd}$$

- this is category of summing functors $\Phi_E : P(E_G) \to \mathcal{C}$ with conservation law at vertives: for all $A \in P(V_G)$

$$\Phi_E(s^{-1}(A)) = \Phi_E(t^{-1}(A))$$

  in particular for all $v \in V_G$ have inflow of $\mathcal{C}$-resources equal outflow

$$\oplus_{e:s(e)=v} \Phi_E(e) = \oplus_{e:t(e)=v} \Phi_E(e)$$

- another kind of conservation law expressed by coequalizer

Gamma spaces for networks

- $\mathcal{E}_{\mathcal{C}} : \mathrm{Func}(\mathbf{2}, \mathcal{F}) \to \Delta$ with $\mathcal{E}_{\mathcal{C}}(G) = \mathcal{N}(\Sigma_{\mathcal{C}}(G))$ nerve of equalizer of $s, t : \Sigma_{\mathcal{C}}(E_G) \rightrightarrows \Sigma_{\mathcal{C}}(V_G)$ (equalizer of nerves)

- Example: Linear Neuron
  - category of weighted codes $\mathcal{W}\mathrm{Codes}_{n,*}$
  - summing functors $\Sigma_{\mathcal{W}\mathrm{Codes}_{n,*}}(E_G)$ and $\Sigma_{\mathcal{W}\mathrm{Codes}_{n,*}}(V_G)$
  - directed graph G has a single outgoing edge at each vertex: $\{e \in E_G \,|\, s(e) = v\} = \{\mathrm{out}(v)\}$
  - equalizer condition (categorical version of linear neuron)

$$(C_{\mathrm{out}(v)}, \omega_{\mathrm{out}(v)}) = \oplus_{t(e)=v}(C_e, \omega_e)$$

- Need nonlinearities: thresholds and saturation
  - need a formulation analogous to chamber structures (hyperplane arrangements) and different regions of linearity (as for Hopfield network dynamics, see work of C. Curto and collaborators) lifted to the level of category of weighted codes

Things you can do with this

- Gamma spaces have associated topological invariants (homotopy groups) so can decide if assigning resources of type $\mathcal{C}$ or of type $\mathcal{R}$ to a network G is "inequivalent" on the basis of these invariants $\pi_k(\Gamma_{\mathcal{C}}(G)) \neq \pi_k(\Gamma_{\mathcal{R}}(G))$ for some $k \geq 0$
- any (functorial) transformation of resources $\rho : \mathcal{C} \to \mathcal{R}$ carries over the whole structure
- $\Gamma_{\mathrm{Cones}}(G)$ describes the assignments of binary codes to a network (neural codes): passing from codes to probabilities gives information constraints on the network, functorially assigning computational architectures (transition systems) to codes gives computational resources of network, assigning resources to computational systems gives metabolic constraints on computational power of the network etc

More things that are needed

- this setting is just an empty stage: like in the modelling of physical systems kinematics sets the stage but dynamics gives the actual model
- Gamma spaces are just kinematics for consistent assignments of resources to networks
- need to make this setting dynamical to obtain actual models
- a possible way to incorporate dynamics is as a flow on the nerve space $\mathcal{N}(\Sigma_{\mathcal{C}}(G))$: this means fixing the network geometry by having time-variable resources and other structures (codes, weights, probabilities, etc)
- if also want to make the network G itself dynamical need to also move in the space of $\mathrm{Func}(\mathbf{2}, \mathcal{F})$

Some goals

- obtain estimates of Shannon information, mutual information, and integrated information for static and dynamical networks with associated (static or dynamical) codes
- obtain a (functorial) mapping of networks with associated codes to concurrent/distributed computational systems describing computational capacity of the network
- obtain estimates of metabolic constraints from network and its computational structure
- study behavior of dynamical networks in this categorical setting where dynamics consistently transforms the whole structure

... in progress!