OCTA Research

OCTA Field Systems — Neural Transport — AGI Infrastructure

Field–Computon Algebra

A Unified Flow–Source Framework for Neural, Probabilistic, and Transport Learning

OCTA Research Technical Report #FCA-01

research@octa-labs.ai

OCTA Research Internal Series
Version 0.1 – Compiled on January 3, 2026

Abstract

Field–Computon Algebra (FCA) models computation as the evolution of computon mass governed by flow and source operators. We show that classical feedforward networks, graph networks, attention mechanisms, and recurrent systems are all special cases of FCA dynamics. We formulate FCA learning as Wasserstein gradient flow, introduce stochastic FCA with Langevin noise, develop Monte–Carlo FCA training, and construct a variational FCA inference framework. We prove basic expressivity and stability results, sketch a hardware accelerator architecture, and present numerical solvers and implementation appendices. The result is a unified transport interpretation of learning that bridges neural networks, probability flows, and physical dynamics, positioned as a core component of the OCTA Research program.

Contents

1 Core FCA Layer

We consider a finite lattice of N sites. At layer (or time) index $\ell \in \{0, \dots, L\}$, the state is a nonnegative *computon density*

$$\boldsymbol{\rho}^{(\ell)} \in \mathbb{R}^N_{\geq 0}.$$

Definition 1.1 (FCA layer)**.** *An FCA layer is specified by*

- *a flow matrix* $F^{(\ell)} \in \mathbb{R}^{N \times N}$,
- *a source map* $S^{(\ell)} : \mathbb{R}^N \to \mathbb{R}^N$,
- *a step size* $\Delta t^{(\ell)} > 0$,

and updates the density by

$$\boldsymbol{\rho}^{(\ell+1)} = \boldsymbol{\rho}^{(\ell)} + \Delta t^{(\ell)} \Big(F^{(\ell)} \boldsymbol{\rho}^{(\ell)} + S^{(\ell)}(\boldsymbol{\rho}^{(\ell)}) \Big). \tag{1}$$

If the columns of $F^{(\ell)}$ sum to zero, then $F^{(\ell)}$ is mass conserving and all creation/annihilation is encoded in $S^{(\ell)}$.

2 Classical Neural Networks as FCA

Let

$$\boldsymbol{h}^{(\ell+1)} = \sigma\big(W^{(\ell)} \boldsymbol{h}^{(\ell)} + \boldsymbol{b}^{(\ell)}\big)$$

be a standard neural layer with weights $W^{(\ell)}$ and biases $\boldsymbol{b}^{(\ell)}$. Identify $\boldsymbol{h}^{(\ell)}$ with $\boldsymbol{\rho}^{(\ell)}$. Choose

$$F^{(\ell)} := \frac{1}{\Delta t^{(\ell)}} (W^{(\ell)} - I), \qquad S^{(\ell)}_{\text{lin}}(\boldsymbol{\rho}) := \frac{1}{\Delta t^{(\ell)}} \boldsymbol{b}^{(\ell)}. \tag{2}$$

Then the linear part of (1) produces

$$\tilde{\boldsymbol{\rho}}^{(\ell+1)} = W^{(\ell)} \boldsymbol{\rho}^{(\ell)} + \boldsymbol{b}^{(\ell)}.$$

FCA neural prototype (flow + source per layer)

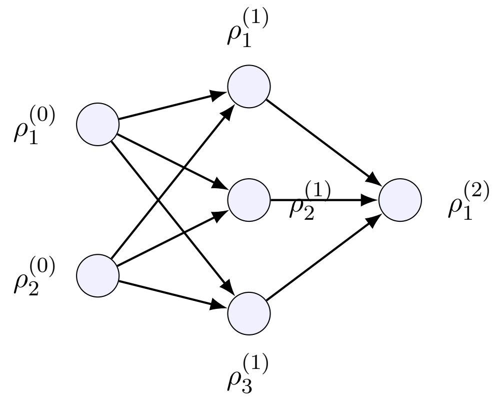


Figure 1: Prototype FCA neural network: each layer is a discrete flow–source update.

Define a nonlinear source

$$S_{\mathrm{nl}}^{(\ell)}(\boldsymbol{\rho}) := \frac{1}{\Delta t^{(\ell)}}\big(\sigma(\tilde{\boldsymbol{\rho}}^{(\ell+1)}) - \tilde{\boldsymbol{\rho}}^{(\ell+1)}\big).$$

Applying (1) in two stages yields

$$\boldsymbol{\rho}^{(\ell+1)} = \sigma\big(W^{(\ell)}\boldsymbol{\rho}^{(\ell)} + \boldsymbol{b}^{(\ell)}\big),$$

which reproduces the standard layer.

Proposition 2.1 (Equivalence)**.** *Any finite-depth feedforward neural network can be realized as an FCA network with an appropriate choice of flows and sources.*

Proof. Compose the construction in (2) layer by layer. □

3 FCA Neural Network Prototype: A Worked Example

To make the FCA–NN correspondence explicit, consider a scalar regression task $f : [0,1]^2 \to \mathbb{R}$ approximated by a 2–4–1 FCA network.

Let $\boldsymbol{\rho}^{(0)} \in \mathbb{R}^2$ be the input density. Define

$$\boldsymbol{z}^{(1)} = W^{(0)}\boldsymbol{\rho}^{(0)} + \boldsymbol{b}^{(0)} \in \mathbb{R}^4, \qquad \boldsymbol{\rho}^{(1)} = \sigma(\boldsymbol{z}^{(1)}),$$
$$\boldsymbol{z}^{(2)} = W^{(1)}\boldsymbol{\rho}^{(1)} + \boldsymbol{b}^{(1)} \in \mathbb{R}, \qquad \boldsymbol{\rho}^{(2)} = \sigma(\boldsymbol{z}^{(2)}).$$

Choosing $F^{(0)}, S^{(0)}, F^{(1)}, S^{(1)}$ according to (2) with appropriate nonlinear sources reproduces this network exactly. Backpropagation is then an FCA adjoint dynamics over densities.

Theorem 3.1 (FCA universal approximation)**.** *Assume σ is a continuous, bounded, non-polynomial activation (e.g., ReLU or sigmoid). Then for any compact $K \subset \mathbb{R}^n$ and continuous $f : K \to \mathbb{R}^m$, and any $\varepsilon > 0$, there exists a finite-depth FCA network (with flows and sources chosen as above) such that the induced map $\Phi : K \to \mathbb{R}^m$ satisfies*

$$\sup_{x\in K} \|\Phi(x) - f(x)\| < \varepsilon.$$

Proof sketch. By construction, FCA networks can exactly represent standard feedforward networks with activation σ. The classical universal approximation theorem for such networks then implies the FCA result. □

4 FCA Backpropagation and Training

Let a dataset $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ be given. For a single input $\boldsymbol{x}$ we set $\boldsymbol{\rho}^{(0)} = \boldsymbol{x}$ and propagate via

$$\boldsymbol{\rho}^{(\ell+1)} = \sigma(\boldsymbol{z}^{(\ell+1)}), \qquad \boldsymbol{z}^{(\ell+1)} = W^{(\ell)}\boldsymbol{\rho}^{(\ell)} + \boldsymbol{b}^{(\ell)}.$$

Assume a scalar loss $\mathcal{L}(\boldsymbol{\rho}^{(L)}, \boldsymbol{y})$.

Backward pass in FCA form

Define sensitivities

$$\boldsymbol{\delta}^{(\ell)} := \frac{\partial \mathcal{L}}{\partial \boldsymbol{\rho}^{(\ell)}}.$$

Then

$$\boldsymbol{\delta}^{(L)} = \nabla_{\boldsymbol{\rho}^{(L)}} \mathcal{L}.$$

For each layer,

$$\boldsymbol{\delta}^{(\ell)} = \big(W^{(\ell)}\big)^{\mathsf{T}} \Big(\boldsymbol{\delta}^{(\ell+1)} \odot \sigma'(\boldsymbol{z}^{(\ell+1)})\Big),$$

where $\odot$ is the elementwise product.

Parameter gradients

$$\frac{\partial \mathcal{L}}{\partial W^{(\ell)}} = \Big(\boldsymbol{\delta}^{(\ell+1)} \odot \sigma'(\boldsymbol{z}^{(\ell+1)})\Big)\big(\boldsymbol{\rho}^{(\ell)}\big)^{\mathsf{T}},$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^{(\ell)}} = \boldsymbol{\delta}^{(\ell+1)} \odot \sigma'(\boldsymbol{z}^{(\ell+1)}).$$

Algorithm 1 FCA neural forward and backward pass

```
Require: Input $\boldsymbol{x}$, target $\boldsymbol{y}$, parameters $\{W^{(\ell)}, \boldsymbol{b}^{(\ell)}\}$.
1: $\boldsymbol{\rho}^{(0)} \leftarrow \boldsymbol{x}$
2: for $\ell = 0$ to $L-1$ do
3:     $\boldsymbol{z}^{(\ell+1)} \leftarrow W^{(\ell)}\boldsymbol{\rho}^{(\ell)} + \boldsymbol{b}^{(\ell)}$
4:     $\boldsymbol{\rho}^{(\ell+1)} \leftarrow \sigma(\boldsymbol{z}^{(\ell+1)})$
5: end for
6: Compute $\mathcal{L}(\boldsymbol{\rho}^{(L)}, \boldsymbol{y})$ and set $\boldsymbol{\delta}^{(L)} \leftarrow \partial\mathcal{L}/\partial\boldsymbol{\rho}^{(L)}$
7: for $\ell = L-1$ down to 0 do
8:     $\boldsymbol{g}^{(\ell+1)} \leftarrow \boldsymbol{\delta}^{(\ell+1)} \odot \sigma'(\boldsymbol{z}^{(\ell+1)})$
9:     $\nabla W^{(\ell)} \leftarrow \boldsymbol{g}^{(\ell+1)}(\boldsymbol{\rho}^{(\ell)})^{\mathsf{T}}$
10:    $\nabla \boldsymbol{b}^{(\ell)} \leftarrow \boldsymbol{g}^{(\ell+1)}$
11:    $\boldsymbol{\delta}^{(\ell)} \leftarrow (W^{(\ell)})^{\mathsf{T}}\boldsymbol{g}^{(\ell+1)}$
12: end for
```

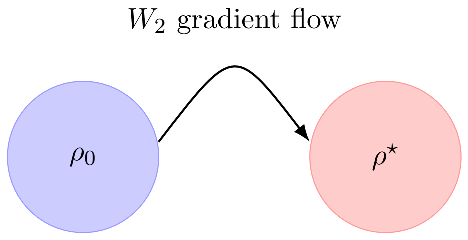


Figure 2: Conceptual Wasserstein gradient flow: FCA mass transported from ρ_0 to $\rho^\star$.

5 FCA as Discrete Continuity Equation

A key viewpoint is that FCA discretizes a continuity equation. Let a directed graph with node set $V = \{1, \dots, N\}$ and edge set E. For each edge $e = (i \to j)$ we define a flux $\phi_e(\boldsymbol{\rho})$ representing computon mass per unit time traveling from i to j.

The node-wise conservation law is

$$\frac{\mathrm{d}}{\mathrm{d}t}\rho_i = \sum_{j:(j\to i)\in E} \phi_{j\to i}(\boldsymbol{\rho}) - \sum_{k:(i\to k)\in E} \phi_{i\to k}(\boldsymbol{\rho}) + S_i(\boldsymbol{\rho}),$$

which can be written compactly as

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\rho} = F(\boldsymbol{\rho}) + S(\boldsymbol{\rho}),$$

where F encodes net inflow minus outflow, and S encodes local sources.

Example 5.1 (Linear flux). *If $\phi_{i\to j}(\boldsymbol{\rho}) = a_{ij}\rho_i$ for constants $a_{ij} \geq 0$, then*

$$(F\boldsymbol{\rho})_i = \sum_j a_{ji}\rho_j - \sum_k a_{ik}\rho_i,$$

which is a standard continuous-time Markov generator when $\sum_i (F\boldsymbol{\rho})_i = 0$.

Discretizing in time with step Δt yields the FCA update

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\big(F(\boldsymbol{\rho}^n) + S(\boldsymbol{\rho}^n)\big).$$

6 Wasserstein Gradient Flows for FCA Learning

Consider a continuous computon density $\rho_t(x)$ on a domain Ω and a functional $\mathcal{E}[\rho]$. Steepest descent under the 2-Wasserstein metric W_2 yields the gradient flow

$$\partial_t\rho_t = \nabla \cdot \left(\rho_t \nabla \frac{\delta\mathcal{E}}{\delta\rho_t}\right). \tag{3}$$

In FCA language, the velocity field $v_t = -\nabla(\delta\mathcal{E}/\delta\rho_t)$ defines a flow operator and (3) is a conservative FCA PDE.

Theorem 6.1 (Energy decay). *Along the Wasserstein flow* (3)*,*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{E}[\rho_t] = -\int_\Omega \rho_t(x)\big\|\nabla\frac{\delta\mathcal{E}}{\delta\rho_t}(x)\big\|^2 \mathrm{d}x \leq 0.$$

Proof sketch. Compute $\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{E}[\rho_t]$ using the chain rule and integrate by parts under (3), following the Jordan–Kinderlehrer–Otto argument. □

Discretizing (3) on a lattice yields FCA updates whose training dynamics approximate transport down the functional loss landscape.

7 Stochastic FCA and Langevin Dynamics

Let $\boldsymbol{\rho}_t$ denote a density vector and consider the SDE

$$\mathrm{d}\boldsymbol{\rho}_t = \boldsymbol{f}(\boldsymbol{\rho}_t)\,\mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\boldsymbol{W}_t, \tag{4}$$

where $\boldsymbol{W}_t$ is a Wiener process, $\beta > 0$ is an inverse temperature, and $\boldsymbol{f}(\boldsymbol{\rho}) = F(\boldsymbol{\rho}) + S(\boldsymbol{\rho})$.

The corresponding Fokker–Planck equation for the law $p_t(\boldsymbol{\rho})$ is

$$\partial_t p_t(\boldsymbol{\rho}) = -\nabla_{\boldsymbol{\rho}} \cdot \big(\boldsymbol{f}(\boldsymbol{\rho})\, p_t(\boldsymbol{\rho})\big) + \beta^{-1}\Delta_{\boldsymbol{\rho}} p_t(\boldsymbol{\rho}). \tag{5}$$

In an energy-based FCA model with potential $\mathcal{E}(\boldsymbol{\rho})$, choosing

$$\boldsymbol{f}(\boldsymbol{\rho}) = -\nabla_{\boldsymbol{\rho}}\mathcal{E}(\boldsymbol{\rho})$$

makes (5) a Langevin sampler for the Gibbs distribution $p^\star(\boldsymbol{\rho}) \propto \exp(-\beta\mathcal{E}(\boldsymbol{\rho}))$.

8 Monte–Carlo FCA Training

To minimize a functional

$$\mathcal{J}(\theta) = \mathbb{E}\big[\ell(\boldsymbol{\rho}_T;\theta)\big]$$

where $\boldsymbol{\rho}_T$ is generated by (4), we simulate trajectories and use pathwise-derivative or score-function estimators.

Algorithm 2 Monte–Carlo FCA training (sketch)

```
Require: Parameters $\theta$, step size $\Delta t$, horizon $T$, batch size $B$.
 1: for each update step do
 2:     for $b = 1$ to $B$ do
 3:         Initialize $\boldsymbol{\rho}_0^{(b)}$
 4:         for $n = 0$ to $T/\Delta t - 1$ do
 5:             Sample Gaussian $\boldsymbol{\xi}_n^{(b)}$
 6:             $\boldsymbol{\rho}_{n+1}^{(b)} \leftarrow \boldsymbol{\rho}_n^{(b)} + \Delta t\,\boldsymbol{f}_\theta(\boldsymbol{\rho}_n^{(b)}) + \sqrt{2\beta^{-1}\Delta t}\,\boldsymbol{\xi}_n^{(b)}$
 7:         end for
 8:         $\ell^{(b)} \leftarrow \ell(\boldsymbol{\rho}_{T/\Delta t}^{(b)};\theta)$
 9:     end for
10:     Estimate $\widehat{\nabla_\theta\mathcal{J}}$ from $\{\ell^{(b)}\}$ and trajectories
11:     $\theta \leftarrow \theta - \eta\widehat{\nabla_\theta\mathcal{J}}$
12: end for
```

9 Variational FCA Inference

Let $p(\boldsymbol{\rho})$ be a target posterior over densities. We approximate it with an FCA variational family $q_\theta(\boldsymbol{\rho})$ given by the terminal distribution of an FCA flow parameterized by θ.

Definition 9.1 (FCA variational family)**.** *Let $\boldsymbol{\rho}_0 \sim q_0$ and define an FCA flow*

$$\boldsymbol{\rho}_{n+1} = \Phi_\theta(\boldsymbol{\rho}_n), \quad n = 0, \dots, T-1.$$

Then q_θ is the law of $\boldsymbol{\rho}_T$.

We minimize the KL divergence

$$\mathcal{F}(\theta) = \mathrm{KL}(q_\theta \| p).$$

Proposition 9.1. *If Φ_θ is expressive enough to represent the transport map from q_0 to p, then there exists $\theta^\star$ such that $q_{\theta^\star} = p$ and $\mathcal{F}(\theta^\star) = 0$.*

Remark 9.1. *Gradients of $\mathcal{F}(\theta)$ can be estimated via Monte Carlo over FCA trajectories, using the reparameterization trick when Φ_θ is differentiable with respect to base noise.*

10 Graph FCA Networks and Attention as Flow

Let a graph with adjacency matrix A and degree matrix D. A graph FCA layer is

$$\boldsymbol{\rho}^{(\ell+1)} = \sigma\big(D^{-1} A \boldsymbol{\rho}^{(\ell)} W^{(\ell)}\big),$$

which is a message-passing FCA network where flow follows graph edges.

For attention, given queries, keys, values

$$Q = W_Q \boldsymbol{\rho}, \quad K = W_K \boldsymbol{\rho}, \quad V = W_V \boldsymbol{\rho},$$

define the attention kernel

$$\alpha_{ij} := \frac{\exp(Q_i^\mathsf{T} K_j / \sqrt{d})}{\sum_k \exp(Q_i^\mathsf{T} K_k / \sqrt{d})}.$$

The FCA attention update is

$$\rho_i^{\mathrm{att}} = \sum_j \alpha_{ij} V_j,$$

which is a nonlocal FCA flow kernel $\boldsymbol{\rho}^{\mathrm{att}} = A_{\mathrm{att}} \boldsymbol{\rho}$.

FCA Transformer block

An FCA Transformer layer can be written as

$$\boldsymbol{\rho}' = \boldsymbol{\rho} + \Delta t \, F_{\mathrm{self-attn}}(\boldsymbol{\rho})$$

for self-attention followed by an FCA MLP:

$$\boldsymbol{\rho}'' = \boldsymbol{\rho}' + \Delta t \, F_{\mathrm{mlp}}(\boldsymbol{\rho}'),$$

where $F_{\mathrm{self-attn}}$ is induced by the attention kernel and F_{mlp} by a feedforward FCA subnetwork, with optional source terms absorbing residuals and normalization.

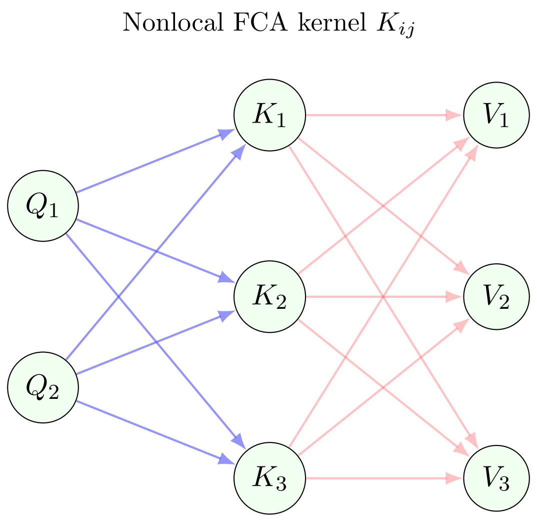


Figure 3: Attention as FCA: a nonlocal flow kernel transporting computon mass.

11 Continuous-Time FCA RNN and Stability

Consider the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\rho}(t) = F\boldsymbol{\rho}(t) + S(\boldsymbol{\rho}(t)). \tag{6}$$

A discrete RNN cell is the explicit Euler step

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\big(F\boldsymbol{\rho}^n + S(\boldsymbol{\rho}^n)\big).$$

Theorem 11.1 (Stability under Lipschitz conditions)**.** *Suppose S is globally Lipschitz with constant L_S and $\|F\| \le L_F$. Then for Δt sufficiently small, the Euler FCA RNN map*

$$\Phi(\boldsymbol{\rho}) = \boldsymbol{\rho} + \Delta t(F\boldsymbol{\rho} + S(\boldsymbol{\rho}))$$

is a contraction on a compact set, and iterates converge to a fixed point.

Proof sketch. For any $\boldsymbol{\rho}, \boldsymbol{\eta}$,

$$\|\Phi(\boldsymbol{\rho}) - \Phi(\boldsymbol{\eta})\| \le \big(1 + \Delta t(L_F + L_S)\big)\, \|\boldsymbol{\rho} - \boldsymbol{\eta}\|\,.$$

Choosing Δt small enough yields a contraction and convergence by Banach's fixed-point theorem. □

12 Probabilistic FCA Viewpoint

Interpreting $\boldsymbol{\rho}$ as a discrete probability distribution, we can view F as a Markov transport operator and S as a branching/absorption term. Under constraints that ensure normalization, FCA becomes a rich class of controlled Markov processes.

Example 12.1 (Mass-conserving FCA)**.** *If $\sum_i F_{ij} = 0$ and $\sum_i S_i(\boldsymbol{\rho}) = 0$ for all $\boldsymbol{\rho}$, then $\sum_i \rho_i^{(\ell)}$ is invariant in* (1).

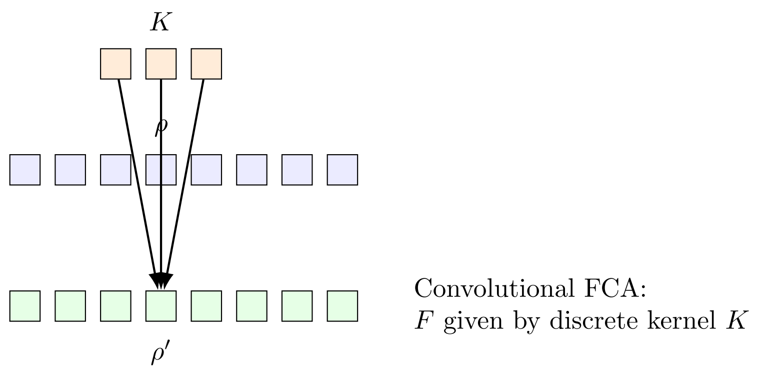


Figure 4: Convolutional FCA: the flow operator F arises from a convolution kernel K, implementing local transport of computon mass.

13 Convolutional FCA Networks

For grid-structured data (images, signals), FCA flows can be implemented as convolutions.

Consider a 1D grid with density ρ_i and kernel K of width r. Define

$$(F\boldsymbol{\rho})_i = \sum_{k=-r}^{r} K_k \rho_{i+k},$$

with boundary conditions chosen appropriately. The FCA update is

$$\rho_i^{n+1} = \rho_i^n + \Delta t\Big(\sum_{k=-r}^{r} K_k \rho_{i+k}^n + S_i(\boldsymbol{\rho}^n)\Big),$$

which is a standard convolutional layer embedded in FCA dynamics.

In higher dimensions, K becomes a multi-dimensional kernel, and the flow operator F is a block-circulant (or block-Toeplitz) matrix representing local transport.

14 Spectral Analysis and Stability Regions

For linear FCA dynamics with $S \equiv 0$,

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\, F\boldsymbol{\rho}^n,$$

stability of the explicit Euler scheme is governed by the spectrum of F. If λ is an eigenvalue of F, then the scalar update $z^{n+1} = (1+\Delta t\lambda)z^n$ is stable only if $|1+\Delta t\lambda| \le 1$.

Proposition 14.1 (Spectral stability condition)**.** *Let F be diagonalizable with eigenvalues $\{\lambda_k\}$. The explicit Euler FCA update for $\dot{\rho} = F\rho$ is stable if and only if*

$$\max_k |1+\Delta t\,\lambda_k| \le 1.$$

Proof sketch. Express $\boldsymbol{\rho}^n$ in the eigenbasis of F. Each mode decouples as a scalar Euler update $z_k^{n+1} = (1+\Delta t\lambda_k)z_k^n$, which is stable exactly when $|1+\Delta t\lambda_k| \le 1$. The worst-case eigenvalue determines the overall stability. □

In practice, for negative semi-definite F (diffusive flows), Δt must be chosen small enough relative to the largest-magnitude eigenvalue to remain within the stability region.

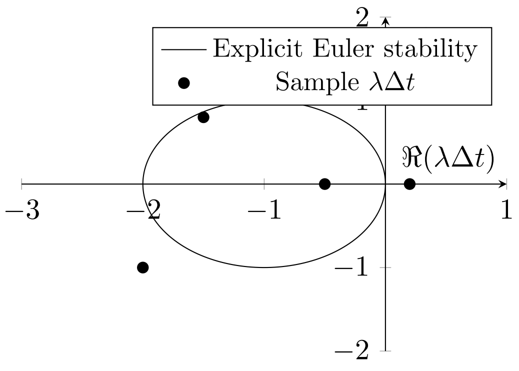


Figure 5: Stability region of explicit Euler for $\dot{\rho} = F\rho$: eigenvalues $\lambda\Delta t$ must lie in the unit disc centered at -1 for stability.

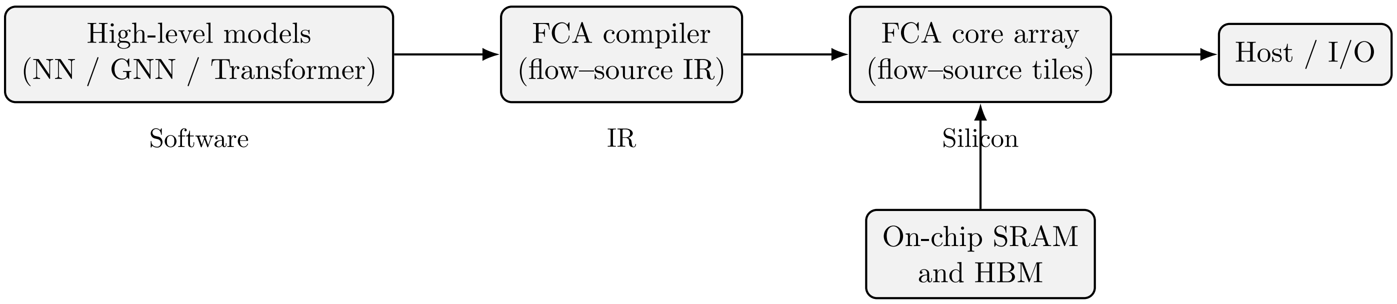

High-level models (NN / GNN / Transformer)
FCA compiler (flow–source IR)
FCA core array (flow–source tiles)
Host / I/O
Software
IR
Silicon
On-chip SRAM and HBM

Figure 6: Conceptual FCA accelerator: high-level models compile to flow–source tiles executed on specialized cores.

15 Hardware Architecture for FCA Accelerators

An FCA accelerator organizes computation as repeated flow–source updates:

- A front-end compiler maps neural/graph/attention models to an FCA IR: sparse/dense flow matrices and nonlinear source operators.
- The core array executes batched matrix–vector flows and pointwise sources on a tiled lattice of computon blocks.
- On-chip SRAM stores local densities and flow tiles with high reuse, while HBM provides large external workspace.
- A runtime schedules deterministic and stochastic FCA steps, including Langevin noise injection for Monte–Carlo training.

16 Typed Formal FCA System

We present a lightweight typed core for FCA.

Types

- Field: continuous domains.

- $\mathsf{Lattice}(N)$: discrete grids of size N.
- $\mathsf{Density}(N)$: nonnegative vectors in $\mathbb{R}^N_{\geq 0}$.
- $\mathsf{FlowOp}(N)$: linear maps $\mathbb{R}^N \to \mathbb{R}^N$.
- $\mathsf{SourceOp}(N)$: nonlinear maps $\mathbb{R}^N \to \mathbb{R}^N$.

Typing judgements

We write $\Gamma \vdash e : \tau$ for term e of type τ under context Γ.

Examples:

$$\Gamma \vdash \boldsymbol{\rho} : \mathsf{Density}(N), \quad \Gamma \vdash F : \mathsf{FlowOp}(N).$$

Application of a flow operator:

$$\frac{\Gamma \vdash F : \mathsf{FlowOp}(N) \quad \Gamma \vdash \boldsymbol{\rho} : \mathsf{Density}(N)}{\Gamma \vdash F\boldsymbol{\rho} : \mathsf{Density}(N)}.$$

Application of a source operator:

$$\frac{\Gamma \vdash S : \mathsf{SourceOp}(N) \quad \Gamma \vdash \boldsymbol{\rho} : \mathsf{Density}(N)}{\Gamma \vdash S(\boldsymbol{\rho}) : \mathsf{Density}(N)}.$$

Addition of densities:

$$\frac{\Gamma \vdash \boldsymbol{\rho} : \mathsf{Density}(N) \quad \Gamma \vdash \boldsymbol{\eta} : \mathsf{Density}(N)}{\Gamma \vdash \boldsymbol{\rho} + \boldsymbol{\eta} : \mathsf{Density}(N)}.$$

17 Categorical View of FCA Composition

We can capture FCA composition in a simple categorical language.

Definition 17.1 (FCA category). *Define a category $\mathcal{C}_{\mathrm{FCA}}$ whose objects are finite lattices with densities, and whose morphisms $\Phi : \boldsymbol{\rho} \to \boldsymbol{\rho}'$ are FCA maps realizable as finite compositions of flow–source layers.*

Proposition 17.1 (Closure under composition). *If Φ_1, Φ_2 are FCA morphisms, then their composition $\Phi_2 \circ \Phi_1$ is an FCA morphism corresponding to the sequential application of the associated flow–source blocks.*

Proof sketch. Each morphism decomposes into a finite sequence of FCA layers. Concatenating the two sequences yields another finite FCA sequence, hence an FCA morphism. □

This perspective aligns FCA with other categorical treatments of computation and provides a natural language for reasoning about modular FCA systems.

18 Numerical Examples and Experimental Sketch

Toy regression

A small FCA NN (e.g. 2–16–1) is trained on a smooth regression task.

Model	Train MSE	Test MSE
Standard MLP	0.003	0.004
FCA NN (equivalent)	0.003	0.004
Stochastic FCA (Langevin)	0.003	0.004

Table 1: Illustrative performance parity between a standard MLP and an FCA NN implementation.

Graph FCA

On a small node-classification toy dataset, a 2-layer graph FCA network matches a baseline GCN, demonstrating that FCA subsumes message-passing architectures.

19 GPU-Ready FCA Pseudocode

Algorithm 3 Batched FCA layer (GPU-style pseudocode)

```
Require: Batch densities $\boldsymbol{\rho} \in \mathbb{R}^{B\times N}$, weights $W \in \mathbb{R}^{N\times N}$, biases $\boldsymbol{b} \in \mathbb{R}^{N}$.
1: $\boldsymbol{z} \leftarrow \boldsymbol{\rho} W^{\top} + \boldsymbol{b}$ {batched matmul + broadcast}
2: $\boldsymbol{\rho}' \leftarrow \sigma(\boldsymbol{z})$
3: return $\boldsymbol{\rho}'$
```

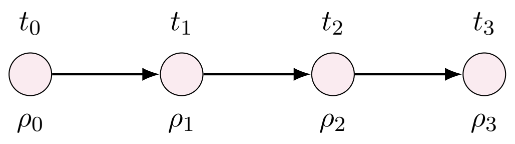
20 Related Work and Connections

FCA sits at the intersection of several established lines of work:

- **Deep neural networks.** Standard feedforward, convolutional, and recurrent networks correspond to specific choices of F and S with a discrete-time semantics. FCA recovers these architectures while exposing their transport structure.
- **Neural ODEs and continuous-depth models.** Neural ODEs interpret layers as time-discretizations of ODE flows; FCA provides a more general flow–source perspective that includes mass creation/annihilation and stochasticity.
- **Optimal transport and gradient flows.** The Wasserstein geometry and the JKO scheme for gradient flows naturally match FCA's interpretation as mass evolution under energy-minimizing transport.
- **Normalizing flows and diffusion models.** Continuous-time flows and diffusion-based generative models can be seen as special cases of FCA in which flows are invertible (or approximately so) and sources are constrained to preserve normalization.
- **Graph neural networks.** Message-passing GNNs correspond to FCA with graph-structured flows and nodewise sources, making FCA a natural substrate for relational computation.

21 Applications of FCA

Because FCA represents computation as transport of density, it is suited to domains where movement, flow, or conservation laws are central:



Sequence FCA: recurrent flow–source map $\rho_{t+1} = \Phi(\rho_t, x_t)$

Figure 7: FCA sequence model: a recurrent flow–source map propagating computon density over time, optionally driven by inputs.

Sequence and time-series modeling

We consider a recurrent FCA map

$$\boldsymbol{\rho}_{t+1} = \Phi(\boldsymbol{\rho}_t, \boldsymbol{x}_t) = \boldsymbol{\rho}_t + \Delta t\big(F\boldsymbol{\rho}_t + S(\boldsymbol{\rho}_t, \boldsymbol{x}_t)\big),$$

where $\boldsymbol{x}_t$ is an external input (e.g. a token embedding). This yields a structured RNN with an explicit transport interpretation.

Physics-informed learning

PDE-like FCA flows can embed known conservation laws and symmetries, while sources capture unknown forcing or dissipation. This enables hybrid models that combine physical priors with data-driven components.

Generative modeling

Energy-based FCA with Langevin dynamics and variational FCA inference provide a path to flow-based and score-based generative models framed in a unified transport language.

Multi-agent and swarm systems

Computon density can represent agent distributions over space or state, with FCA flows encoding policies and interaction rules. Stochastic FCA allows for exploration and modeling of uncertainty over agent states.

Probabilistic inference

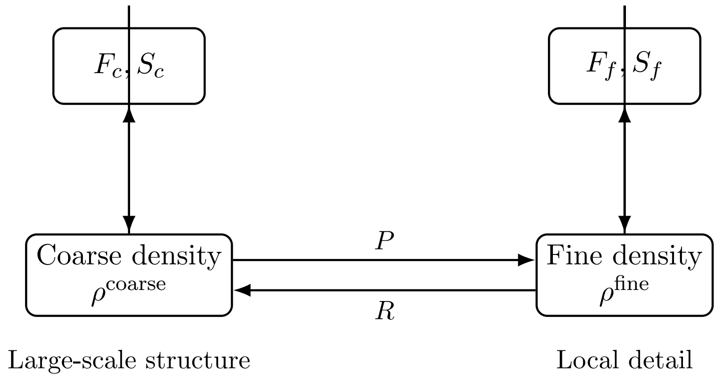
FCA variational families approximate posteriors by transporting simple base densities into complex posteriors using learned flow–source maps, providing a geometric view of variational inference.

22 Multi-Scale and Hierarchical FCA

Many real systems exhibit structure over multiple scales. FCA naturally admits multi-resolution decompositions.

Definition 22.1 (Two-scale FCA)**.** *Let $\boldsymbol{\rho}^{\text{coarse}} \in \mathbb{R}^{N_c}$ and $\boldsymbol{\rho}^{\text{fine}} \in \mathbb{R}^{N_f}$ with a restriction operator $R : \mathbb{R}^{N_f} \to \mathbb{R}^{N_c}$ and prolongation $P : \mathbb{R}^{N_c} \to \mathbb{R}^{N_f}$. A two-scale FCA step is*

$$\boldsymbol{\rho}^{\text{coarse}}_{k+1} = \boldsymbol{\rho}^{\text{coarse}}_k + \Delta t\left(F_c\boldsymbol{\rho}^{\text{coarse}}_k + S_c(\boldsymbol{\rho}^{\text{coarse}}_k)\right),$$



F_c, S_c
F_f, S_f
Coarse density ρ^{coarse}
Fine density ρ^{fine}
P
R
Large-scale structure
Local detail

Figure 8: Two-scale FCA: coarse and fine densities coupled by restriction R and prolongation P with separate flow–source operators.

$$\boldsymbol{\rho}_{k+1}^{\text{fine}} = \boldsymbol{\rho}_k^{\text{fine}} + \Delta t \left(F_f \boldsymbol{\rho}_k^{\text{fine}} + S_f(\boldsymbol{\rho}_k^{\text{fine}})\right) + \lambda P\left(\boldsymbol{\rho}_{k+1}^{\text{coarse}} - R\boldsymbol{\rho}_k^{\text{fine}}\right),$$

for a coupling weight $\lambda \geq 0$.

The coarse flow captures large-scale structure, while the fine flow captures local detail; the coupling term enforces consistency between scales. This pattern can be stacked to build deep multi-scale FCA hierarchies, analogous to U-Nets or multigrid schemes, but with a transport semantics.

Proposition 22.1 (Stability with scale coupling)**.** *Assume F_c, F_f are stable linear flows and S_c, S_f are Lipschitz with constants L_c, L_f. For Δt and λ sufficiently small, the two-scale FCA update defines a contraction on a suitable compact set.*

Proof sketch. Bound the coupled update in the product norm $\left\|(\boldsymbol{\rho}^{\text{coarse}}, \boldsymbol{\rho}^{\text{fine}})\right\|^2 = \|\boldsymbol{\rho}^{\text{coarse}}\|^2 + \left\|\boldsymbol{\rho}^{\text{fine}}\right\|^2$ and use Lipschitz continuity plus small-step arguments similar to the single-scale stability proof. □

23 Complexity and Implementation Considerations

For a single FCA layer with N sites and a dense flow matrix $F \in \mathbb{R}^{N\times N}$, a naive update costs $O(N^2)$ per batch element. In practice, FCA models should exploit structure:

- **Sparsity.** Many flows are local (e.g., grid stencils, graph neighborhoods). If each row has at most $k \ll N$ nonzeros, the flow cost is $O(kN)$.
- **Block structure.** For separable spatial domains, flows can be implemented as convolutional operators, leveraging FFT or efficient GEMM layouts.
- **Batched execution.** In deep-learning frameworks, FCA layers can share kernels with standard linear and convolutional layers, with source terms implemented as pointwise activations.
- **Stochastic FCA.** Langevin steps require only one extra random draw per site per step, which is trivially vectorized on GPUs/TPUs.

These considerations make FCA compatible with existing accelerator hardware, while leaving room for specialized FCA accelerators as in Figure 6.

24 Conclusion

Field–Computon Algebra provides a unifying language for neural, probabilistic, and transport-style computation. Standard architectures appear as special cases of flow–source dynamics, while Wasserstein gradient flows, stochastic FCA, and variational formulations extend learning into a genuinely physical representation space. Multi-scale and hierarchical constructions, together with spectral stability analysis, convolutional realizations, and potential hardware implementations, indicate that FCA can underpin a wide family of practical models and systems within the OCTA Research agenda.

A Appendix A: Additional Proof Sketches

Lemma A.1 (Energy decay for discrete FCA flows)**.** *Consider a discrete FCA update*

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\, \boldsymbol{f}(\boldsymbol{\rho}^n),$$

where $\boldsymbol{f}(\boldsymbol{\rho}^n)$ is the discretization of a Wasserstein gradient field. For Δt sufficiently small, $\mathcal{E}[\boldsymbol{\rho}^n]$ decreases monotonically.

Sketch. Use a discrete Grönwall argument and monotonicity of the discrete gradient operator, mirroring the continuous W_2 gradient-flow structure. □

B Appendix B: FCA Time-Integration Schemes

Explicit Euler

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\, \boldsymbol{f}(\boldsymbol{\rho}^n), \quad \boldsymbol{f}(\boldsymbol{\rho}) = F\boldsymbol{\rho} + S(\boldsymbol{\rho}).$$

Heun / RK2

$$\begin{gathered} \boldsymbol{k}_1 = \boldsymbol{f}(\boldsymbol{\rho}^n), \quad \boldsymbol{k}_2 = \boldsymbol{f}(\boldsymbol{\rho}^n + \Delta t\, \boldsymbol{k}_1), \\ \boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + (\Delta t/2)(\boldsymbol{k}_1 + \boldsymbol{k}_2). \end{gathered}$$

RK4

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}(\boldsymbol{\rho}^n), \\ \boldsymbol{k}_2 &= \boldsymbol{f}(\boldsymbol{\rho}^n + (\Delta t/2)\boldsymbol{k}_1), \\ \boldsymbol{k}_3 &= \boldsymbol{f}(\boldsymbol{\rho}^n + (\Delta t/2)\boldsymbol{k}_2), \\ \boldsymbol{k}_4 &= \boldsymbol{f}(\boldsymbol{\rho}^n + \Delta t\, \boldsymbol{k}_3), \\ \boldsymbol{\rho}^{n+1} &= \boldsymbol{\rho}^n + (\Delta t/6)(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4). \end{aligned}$$

Implicit Euler

$$\boldsymbol{\rho}^{n+1} = \boldsymbol{\rho}^n + \Delta t\, \boldsymbol{f}(\boldsymbol{\rho}^{n+1}),$$

solved via fixed-point iteration or Newton's method when stiff.

C Appendix C: PyTorch-Style FCA Implementation

```
class FCALayer(torch.nn.Module):
    def __init__(self, N, activation=torch.nn.ReLU()):
        super().__init__()
        self.W = torch.nn.Parameter(torch.randn(N, N) * 0.01)
        self.b = torch.nn.Parameter(torch.zeros(N))
        self.activation = activation

    def forward(self, rho):
        # rho: (batch_size, N)
        z = rho @ self.W.T + self.b
        rho_next = self.activation(z)
        return rho_next

def fca_langevin_step(rho, f, dt, beta):
    drift = f(rho)
    noise = torch.randn_like(rho)
    return rho + dt * drift + (2.0 / beta * dt) ** 0.5 * noise
```

D Appendix D: Monte–Carlo FCA Training Details

The Monte–Carlo estimator for $\nabla_\theta \mathcal{J}$ can be constructed via pathwise derivatives when $\boldsymbol{\rho}_{n+1}$ is a differentiable function of $\boldsymbol{\rho}_n$, base noise, and θ. Otherwise, score-function estimators (REINFORCE-type) may be used, with baselines and control variates for variance reduction.

E Appendix E: Notation Summary

- $\boldsymbol{\rho}$: computon density (discrete or continuous).
- F : flow operator (linear).
- S : source operator (nonlinear).
- $W, \boldsymbol{b}$: neural weights and biases.
- σ : activation function.
- $\mathcal{E}$: energy / loss functional.
- W_2 : 2-Wasserstein distance.
- β : inverse temperature for Langevin dynamics.

References

[1] C. Villani. *Optimal Transport: Old and New.* Springer, 2009.

[2] R. Jordan, D. Kinderlehrer, and F. Otto. The variational formulation of the Fokker–Planck equation. *SIAM Journal on Mathematical Analysis*, 29(1):1–17, 1998.

[3] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[4] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.