Hyperdimensional Computing in Biological Substrates: Framework and Implementation Principles

Rohan Vinaik
Independent Researcher
rohan.vinaik@gmail.com

Abstract

Biological systems encode and process information across multiple scales and substrates, from protein conformations to neural population codes. We present a theoretical framework demonstrating how this natural encoding maps onto hyperdimensional computing (HDC) paradigms, where information is distributed across very high-dimensional vector spaces. Unlike traditional low-dimensional digital abstractions, HDC naturally captures the rich computational capacity inherent in biological systems: proteins encode information through amino acid sequences (∼4.3 bits/position), conformational dynamics (hundreds of dimensions), and post-translational modifications (exponential state spaces); DNA sequences span 4^L discrete states with thermodynamic structure; gene expression profiles occupy tens-of-thousands-dimensional spaces; and neural populations represent information through millions to billions of activity dimensions. We formalize how biological substrates implement core HDC operations—binding (composition), bundling (superposition), and permutation (sequencing)—through physical mechanisms including allosteric regulation, strand displacement, and synaptic integration. The framework establishes connections between HDC's mathematical properties (noise tolerance, associative memory, compositionality) and biological computation's observed characteristics (robustness, distributed processing, graceful degradation). We demonstrate the framework's utility through analysis of protein design with synthetic amino acid expansion, showing how increasing encoding alphabet from 20 to 30 amino acids expands computational capacity while maintaining biological functionality. Applications in synthetic biology demonstrate how HDC principles guide both analysis and engineering of biological computation, particularly for gene circuit design and cellular programming. This framework provides both analytical tools for understanding natural biological information processing and design principles for synthetic biological computing systems.

Keywords: hyperdimensional computing, biological computation, synthetic biology, biomolecular computing, distributed representation, protein engineering, synthetic amino acids

1 Introduction

1.1 Biological Computation Beyond Digital Logic

From molecular self-assembly to neural computation, living systems process information through mechanisms that appear fundamentally different from traditional digital computers. Rather than manipulating discrete symbols through logic gates, biological computation emerges from the physical dynamics of high-dimensional molecular systems—protein conformations, genetic regulatory networks, neural population codes—that naturally implement distributed, analog information processing [Adamatzky, 2016]. Unlike silicon-based architectures that operate on binary states with deterministic transitions, biological systems exploit continuous state spaces, stochastic dynamics, and massively parallel molecular interactions to perform computation with remarkable robustness and efficiency.

The historical approach to biocomputing has largely attempted to impose digital logic frameworks onto biological substrates, designing DNA logic gates [Qian & Winfree, 2011], genetic toggle switches, and other circuit elements that mirror electronic components. While these efforts have produced important proof-of-concept demonstrations [Nielsen et al., 2016], they inadequately capture the rich computational capacity inherent in biological systems. A protein, for instance, encodes information not only in its amino acid sequence but also through conformational dynamics, allosteric regulation, post-translational modifications, and multi-scale assembly into complexes. This multi-level encoding naturally suggests computational paradigms more sophisticated than binary logic.

Understanding and engineering biological computation requires computational frameworks that match the intrinsic properties of living systems. The question is not how to force biological substrates into digital abstractions, but rather what computational paradigms naturally emerge from biological information encoding.

1.2 Hyperdimensional Computing as Natural Match

Hyperdimensional computing (HDC) offers a promising lens for understanding biological information processing. In HDC, information is represented as points in very high-dimensional vector spaces (typically thousands to millions of dimensions), with similarity encoded through geometric relationships [Kanerva, 2009]. Rather than storing information in addressable memory locations or explicit symbolic structures, HDC distributes information across entire vectors, where each dimension contributes to the overall meaning.

Operations on these representations—composition through binding, superposition through bundling, sequencing through permutation—naturally emerge from the mathematical structure of high-dimensional spaces. **Binding** combines vectors to create composite representations, typically implemented through element-wise multiplication or circular convolution. **Bundling** superimposes multiple vectors through addition, creating representations that preserve information about constituent elements. **Permutation** transforms vectors to represent sequences or positions, typically through coordinate rotation or index shifting.

Critically, HDC systems exhibit noise tolerance, graceful degradation, and associative memory properties that mirror biological computation's observed characteristics [Rahimi et al., 2016]. In high-dimensional spaces, randomly chosen vectors are approximately or-

thogonal with high probability; this mathematical property provides natural robustness to noise. Random perturbations typically preserve essential geometric relationships, enabling computation to proceed despite component failure or corruption. The distributed nature of information encoding means that no single dimension is critical—similar to how biological systems maintain function despite molecular stochasticity and environmental perturbations.

The hypothesis we explore here is that biological systems naturally operate as hyperdimensional computing substrates. Proteins occupy conformational spaces of hundreds of dimensions. Gene expression profiles span tens of thousands of dimensions. Neural population codes distribute information across millions to billions of dimensions. These are not accidental properties but rather intrinsic features that may have been selected during evolution precisely because high-dimensional encoding provides computational advantages: noise robustness, associative memory, efficient similarity operations, and compositional structure.

1.3 Framework Overview and Contributions

This paper develops a comprehensive framework for understanding biological computation through the HDC lens. We demonstrate that diverse biological substrates—proteins, nucleic acids, neural tissue—naturally implement HDC primitives through their physical properties and dynamics. The framework provides:

Theoretical foundation: Mathematical formalization of how biological encoding mechanisms map onto HDC operations. We characterize the effective dimensionality of different biological substrates, establish correspondence between physical mechanisms (allosteric regulation, strand displacement, synaptic integration) and HDC operations (binding, bundling, permutation), and derive noise tolerance properties from biological stochasticity.

Multi-level encoding analysis: Comprehensive treatment of how biological systems encode information across hierarchical scales. For proteins: primary sequence (discrete symbolic), secondary structure (geometric), tertiary conformation (high-dimensional continuous), quaternary interactions (combinatorial). For nucleic acids: sequence space (discrete), thermodynamic properties (continuous), structural dynamics (conformational). For neural systems: single-neuron firing rates (low-dimensional), population codes (high-dimensional), synaptic connectivity (network-level).

Synthetic amino acid expansion: Analysis of how genetic code expansion through non-canonical amino acids (ncAAs) extends protein encoding capacity. We show that incorporating ncAAs increases both information density (bits per residue) and functional diversity (expanded chemical space), providing explicit engineering handles for designing protein-based HDC systems.

Design principles: General principles for engineering biological computing systems that exploit HDC properties. These include substrate selection criteria based on dimensionality and noise characteristics, encoding strategies that map computational tasks onto biological operations, and readout methods that extract high-dimensional information from biological measurements.

Synthetic biology applications: Demonstration of framework utility through applications in gene circuit design and cellular programming. We show how HDC principles guide the engineering of synthetic biological systems, with particular emphasis on distributed sensing, robust pattern recognition, and adaptive cellular behaviors.

The framework synthesizes insights from molecular biology, synthetic biology, computer science, and applied mathematics to provide both analytical tools for understanding natural biological information processing and practical guidance for engineering synthetic biocomputing systems.

2 Biological Substrates as HDC Implementers

2.1 Proteins and Synthetic Amino Acid Expansion

2.1.1 Multi-Level Information Encoding

Proteins naturally implement hierarchical information encoding across multiple structural levels, creating an intrinsically hyperdimensional representation. The **primary structure** (linear amino acid sequence) provides a discrete symbolic representation where each position encodes one of 20 canonical amino acids, yielding approximately $\log_2(20) \approx 4.3$ bits per position. For a typical protein of length $L = 300$ residues, this provides ∼1,300 bits of sequence-level information.

However, sequence-level encoding represents only the most basic layer. **Secondary structure** elements—α-helices, β-sheets, loops—create local geometric constraints determined by hydrogen bonding patterns and backbone dihedral angles. These elements introduce additional information channels: each residue's conformational state (helix/sheet/loop) adds ∼1.6 bits, and the specific dihedral angles (ϕ, ψ) contribute continuous degrees of freedom. For an N-residue protein, the backbone alone contributes $2N$ angular dimensions.

Tertiary structure (three-dimensional fold) determines functional properties through precise spatial arrangement of catalytic residues, binding sites, and allosteric regulatory domains. The conformational space dimensionality grows substantially: backbone dihedrals ($2N$ dimensions), side chain rotamers (typically 1-4 χ angles per residue, adding $\sim N$ dimensions), and overall orientation (6 rigid-body degrees of freedom). The effective dimensionality of protein conformational space is typically hundreds to thousands of dimensions [Huang et al., 2016].

Quaternary structure (protein complex formation) enables sophisticated combinatorial logic through regulated assembly and disassembly. If a system contains M different protein types, each able to exist in C conformational states, the number of possible complex states scales as C^M times the number of geometric arrangements. This exponential scaling with component number provides massive state complexity suitable for high-dimensional computation.

This multi-level encoding creates a naturally hyperdimensional representation where each level contributes distinct information channels. A protein state can be conceptualized as a point in a very high-dimensional space defined by: sequence identity, conformational flexibility, interaction partnerships, modification status. Operations that modify any of these aspects effectively perform transformations in this high-dimensional state space.

2.1.2 Synthetic Amino Acid Incorporation

The canonical 20 amino acid alphabet constrains the chemical diversity and functional capacity of natural proteins. Recent advances in genetic code expansion enable site-specific incorporation of non-canonical amino acids (ncAAs) with novel chemical properties: photocrosslinking groups, bioorthogonal reactive handles, fluorescent probes, post-translational modification mimics, and chemically distinct side chains [Chin, 2017].

Engineering orthogonal tRNA/aminoacyl-tRNA synthetase pairs enables multiple distinct ncAAs to be incorporated within a single protein, dramatically expanding the encoding alphabet. From an information-theoretic perspective, adding ncAAs increases the bits per position: 20 amino acids encode 4.3 bits/position, 30 amino acids encode 4.9 bits/position (14% increase), 50 amino acids encode 5.6 bits/position (30% increase).

More importantly, ncAAs introduce chemical properties absent from the canonical set: azide and alkyne groups for bioorthogonal click chemistry, unnatural aromatic rings with distinct electronic properties, backbone modifications that constrain conformation, and metal-chelating side chains. This expansion increases both information density at the primary structure level and functional diversity through novel conformational and chemical properties that affect higher-order structures.

From a computational perspective, ncAA incorporation extends the dimensionality of protein state spaces in two ways: **(1)** Increased sequence alphabet directly expands the space of possible primary sequences from 20^L to A^L where A is the expanded alphabet size. **(2)** Novel chemical properties expand conformational space by introducing new interaction modes, stability constraints, and regulatory mechanisms unavailable to canonical proteins.

2.1.3 Conformational Dynamics as HDC Operations

Protein folding occurs on complex energy landscapes characterized by multiple local minima representing distinct conformational states. The system's tendency to minimize free energy effectively implements an analog optimization process navigating this high-dimensional landscape. By engineering energy landscapes through sequence design and environmental control (pH, temperature, ligands, ncAAs), specific computational tasks can be encoded as energetic constraints that guide the protein toward desired structural states.

The high dimensionality of conformational space (hundreds of degrees of freedom for typical proteins) naturally supports hyperdimensional computing paradigms. Conformational ensembles can represent distributed vector encodings where geometric similarity between states corresponds to functional similarity. Consider two protein conformations $\mathbf{c}_1$ and $\mathbf{c}_2$ represented as vectors in conformational space. Their dot product $\mathbf{c}_1 \cdot \mathbf{c}_2$ measures conformational similarity, naturally implementing the similarity operations central to HDC.

Allosteric regulation implements context-dependent operations where the presence of effector molecules modulates the conformational landscape. In HDC terms, allosteric effectors act as gating signals that modify how information encoded in one protein domain affects another. This implements conditional computation analogous to gating mechanisms in neural networks: the output (conformational state, enzymatic activity) depends on both the input (substrate) and the gate (allosteric effector).

2.1.4 Post-Translational Modifications as Logical Operations

Post-translational modifications (PTMs)—phosphorylation, acetylation, methylation, ubiquitination, proteolytic cleavage—provide dynamic, reversible mechanisms for altering protein state. PTMs can toggle functional states (active/inactive), redirect protein localization, modulate interaction specificity, or mark proteins for degradation. From a computational perspective, PTMs implement state transitions that can be orchestrated to perform logical operations.

The combinatorial complexity of multiple PTMs on a single protein creates exponentially large state spaces. A protein with M modifiable sites can exist in 2^M distinct binary modification states (modified/unmodified). More generally, if each site admits K different modifications, the state space contains K^M states. For $M = 10$ and $K = 3$, this yields $3^{10} = 59,049$ possible states, each potentially exhibiting different functional properties.

Synthetic amino acids can be designed to accept novel PTMs or mimic constitutive modifications, expanding the repertoire of controllable state transitions. For example, ncAAs with azide groups enable bioorthogonal click chemistry modifications that are orthogonal to natural PTMs, providing independent information channels. NcAAs that mimic phosphorylation (phosphoserine analogs) or acetylation (N^{ϵ}-acetyl-lysine) allow genetic encoding of specific modification states, fixing certain dimensions of the state space while leaving others dynamic.

In HDC terms, PTMs implement **binding operations**: they create composite representations by associating proteins with modification patterns. A phosphorylation event binds the protein's structural information with a "phosphate vector" representing activation state. Multiple PTMs implement multi-way binding, creating high-dimensional composite vectors that encode complex regulatory states.

2.2 Nucleic Acids: Sequence and Structure

2.2.1 Sequence Space and Encoding Capacity

DNA and RNA sequences define points in 4^L-dimensional discrete space for sequences of length L. This exponential scaling provides enormous encoding capacity: a 100-nucleotide sequence can represent approximately 10^{60} distinct states—far exceeding the estimated number of atoms in the observable universe. The four-letter alphabet (A, T/U, G, C) with well-characterized Watson-Crick base pairing enables predictable hybridization reactions that can implement logic gates, signal amplification, and pattern recognition [Qian & Winfree, 2011].

For HDC applications, discrete sequences can be embedded in continuous high-dimensional spaces based on thermodynamic and structural properties. Each sequence s can be mapped to a vector $\mathbf{v}(s) \in \mathbb{R}^D$ where dimensions represent features such as: melting temperature, GC content, secondary structure propensity, protein binding affinity, catalytic activity (for ribozymes), and structural parameters. This embedding transforms discrete sequence space into continuous high-dimensional space more amenable to HDC operations.

Multiple concurrent sequences in a reaction mixture create ensemble representations where concentrations implement vector magnitudes. If species i is present at concentration c_i and has embedding $\mathbf{v}_i$, the system state is the weighted sum $\mathbf{S} = \sum_i c_i \mathbf{v}_i$. This naturally

implements HDC's **bundling operation**: superposition of multiple vectors through addition. The system's emergent properties depend on the collective ensemble, not individual sequences.

2.2.2 Strand Displacement Dynamics

Toehold-mediated strand displacement reactions enable dynamic DNA circuits that perform multi-step computations. An input strand displaces an incumbent strand from a double-stranded complex through branch migration, releasing output strands that trigger subsequent reactions. These cascades can implement arbitrary Boolean logic, signal restoration, amplification, and memory functions [Qian & Winfree, 2011].

Recent work on analog DNA circuits explicitly exploits concentration-based encoding to implement neural network computations through DNA strand displacement [Cherry & Qian, 2018]. In these systems, strand concentrations represent real-valued activations, and displacement reactions implement weighted connections. This directly parallels HDC's use of vector magnitudes to encode information: each strand species corresponds to a dimension, and its concentration is the component value.

Strand displacement naturally implements HDC operations: **Binding** through concatenation or hybridization reactions that create composite sequences encoding information from multiple inputs. **Bundling** through mixing of strand species at proportional concentrations. **Permutation** through sequential displacement cascades where order determines outcome. The modularity of strand displacement circuits facilitates hierarchical design and integration with other computational substrates.

2.2.3 RNA Structure and Function

RNA combines information storage capabilities (like DNA) with catalytic activity (like proteins), making it uniquely versatile for biocomputing applications. **Regulatory RNA**—small RNAs (siRNA, miRNA) modulate gene expression through sequence-specific binding. Riboswitches alter conformation in response to small molecule ligands, directly linking chemical sensing to regulatory outputs. CRISPR guide RNAs direct sequence-specific DNA binding and modification.

RNA secondary structures (hairpins, internal loops, bulges, junctions) and tertiary structures (pseudoknots, kissing loops) create intricate conformational states beyond linear sequence. This structural complexity adds dimensions to the encoding space: a sequence of length L has not only 4^L sequence variants but also exponentially many possible secondary structures, each with distinct functional properties. Riboswitches demonstrate how RNA structure responds to small molecule binding, implementing input-dependent conformational switches—analogous to HDC permutation operations that transform vectors based on context.

From an HDC perspective, RNA provides intermediate-timescale computation (minutes to hours) between fast protein dynamics (microseconds to seconds) and slow DNA-based memory (hours to stable). RNA's relative instability compared to DNA enables transient signaling and temporal logic, with engineered stability (through sequence design or protective structures) providing tunable signal persistence.

2.3 Neural Population Codes

2.3.1 High-Dimensional Neural Representations

Neural systems naturally implement HDC through population coding, where information is distributed across many neurons firing at different rates. Consider a population of N neurons with firing rates $\mathbf{r} = (r_1, r_2, \ldots, r_N)$. This population activity vector lives in an N-dimensional space where each axis represents one neuron's activity. For biologically realistic populations ($N = 10^3$ to 10^6), this provides inherently high-dimensional representation.

The critical insight is that stimuli are encoded not by individual neurons but by patterns of activity across populations. A visual stimulus, for example, activates specific combinations of neurons in visual cortex; the full population response constitutes a high-dimensional vector that represents the stimulus. Similar stimuli produce similar population vectors (nearby points in neural space), while dissimilar stimuli produce dissimilar vectors (distant points)—precisely the structure required for HDC.

This population coding provides natural noise robustness. Individual neurons exhibit stochastic firing (Poisson-like variability with coefficient of variation $\sim$1), but population averages are reliable due to averaging across many noisy channels. In HDC terms, noise in individual dimensions (neurons) does not substantially corrupt the overall vector direction in high-dimensional space. The signal-to-noise ratio improves as $\sqrt{N}$, so populations of hundreds to thousands of neurons achieve high reliability despite single-neuron unreliability.

2.3.2 Synaptic Operations as HDC Primitives

Synaptic integration naturally implements HDC operations. Each neuron receives inputs from thousands of presynaptic partners through synapses of varying strengths (weights). The postsynaptic potential is the weighted sum of inputs: $V = \sum_i w_i s_i$ where s_i is the presynaptic activity and w_i is the synaptic weight. This weighted summation directly parallels HDC's bundling operation: superposition of multiple input vectors with coefficients determining relative contributions.

Synaptic plasticity—long-term potentiation (LTP) and depression (LTD)—modifies connection weights based on correlated activity. Hebbian learning rules ("neurons that fire together wire together") strengthen synapses between coactive neurons, implementing associative learning. In HDC terms, Hebbian plasticity stores associations by creating overlap between vectors representing related concepts. When patterns consistently co-occur, their vector representations become correlated, enabling one pattern to retrieve the other through approximate matching.

Dendritic computation adds further sophistication. Dendrites exhibit nonlinear integration through active conductances and local spiking, enabling compartmentalized computation where different dendritic branches perform semi-independent operations. This multi-level processing (synaptic, dendritic, somatic) creates hierarchical HDC where local operations (dendritic branches) feed into global integration (somatic output).

2.3.3 Temporal Dynamics and Sequence Encoding

Neural systems encode not only static patterns but also temporal sequences through dynamics. Spike timing carries information beyond firing rates: relative timing of action potentials (millisecond precision), oscillatory phase relationships (theta, gamma rhythms), and sequential activation patterns (neural trajectories) all contribute to information encoding.

Temporal sequences can be encoded through HDC's permutation operations. Representing item A at time t and item B at time $t+1$ requires distinguishing ordered pairs (A, B) from (B, A). In HDC, this is achieved through binding with temporal markers: $\mathbf{v}_{A,t} = \mathbf{v}_A \otimes \mathbf{t}_1$ and $\mathbf{v}_{B,t+1} = \mathbf{v}_B \otimes \mathbf{t}_2$, where $\otimes$ represents binding and $\mathbf{t}_i$ are position vectors. Neural circuits implement similar operations through synaptic delays, dendritic integration windows, and recurrent dynamics that create temporal context.

The high dimensionality of combined spatial and temporal coding is enormous. If N neurons each fire with temporal precision Δt over a window T, the effective dimensionality is $N \times (T/\Delta t)$. For $N = 1000$ neurons, $T = 1$ second, and $\Delta t = 10$ milliseconds, this yields 10^5 effective dimensions—far exceeding typical engineered HDC systems.

2.4 Comparative Analysis

Table 1 summarizes key properties of the three main biological substrates as HDC implementers.

Table 1: Comparison of biological substrates for HDC implementation

Property	**Proteins**	**Nucleic Acids**	**Neural Tissue**
Dimensionality	10^2-10^3	10^3-10^6	10^6-10^9
Timescale	μs-s	min-hours	ms-s
Noise tolerance	Moderate	High	Very high
Encoding density	High	Very high	Moderate
Engineering control	Moderate	High	Low
Readout speed	Moderate	Slow	Fast
In vivo operation	Native	Native	Native
Synthetic production	Mature	Mature	Difficult

Each substrate offers complementary advantages. **Proteins** provide fast dynamics and diverse chemistry, with moderate engineering control through synthetic biology and ncAA incorporation. **Nucleic acids** offer high encoding density and excellent engineering control through well-developed synthesis and sequencing technologies, though kinetics are slower. **Neural tissue** provides the highest dimensionality and natural noise robustness through massive parallelism, but presents greater engineering challenges and ethical considerations.

The choice of substrate depends on application requirements: fast response favors proteins or neural tissue, stable memory favors nucleic acids, complex chemical sensing favors proteins, and pattern recognition favors neural populations.

3 Hyperdimensional Operations in Biological Systems

3.1 Theoretical Foundations

3.1.1 HDC Mathematical Framework

Hyperdimensional computing represents information as points in very high-dimensional vector spaces, typically $\mathbb{R}^D$ or $\{-1, +1\}^D$ with $D \sim 10^3$ to 10^4. Three fundamental operations define HDC [Kanerva, 2009]:

Binding ($\otimes$): Combines two vectors to create a composite representation that is dissimilar to both inputs. Typically implemented through element-wise multiplication (for bipolar vectors) or circular convolution (for real vectors):

$$(\mathbf{a} \otimes \mathbf{b})_i = a_i \cdot b_i \quad \text{or} \quad (\mathbf{a} \otimes \mathbf{b})_i = \sum_{j=0}^{D-1} a_j b_{(i-j) \mod D} \tag{1}$$

Key property: $\mathbf{a} \otimes \mathbf{b}$ is approximately orthogonal to both $\mathbf{a}$ and $\mathbf{b}$ in high dimensions. This enables compositional structure where bound representations do not interfere with their constituents.

Bundling ($\oplus$): Superimposes multiple vectors through addition (often followed by normalization):

$$\mathbf{a} \oplus \mathbf{b} = \frac{\mathbf{a} + \mathbf{b}}{|\mathbf{a} + \mathbf{b}|} \tag{2}$$

Key property: the bundle $\mathbf{s} = \mathbf{a}_1 \oplus \mathbf{a}_2 \oplus \cdots \oplus \mathbf{a}_n$ is similar to each constituent. This implements set membership and prototype formation.

Permutation (π): Transforms a vector to represent positional or sequential information. Typically implemented through coordinate rotation or index shifting:

$$\pi(\mathbf{a})_i = a_{(i+k) \mod D} \tag{3}$$

Key property: permutations preserve vector magnitude and pairwise angles while creating distinguishable representations for different positions in sequences.

3.1.2 Noise Tolerance Properties

A critical advantage of HDC is robustness to noise and component corruption. In high dimensions, randomly chosen vectors are approximately orthogonal with high probability. For normalized vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^D$ with random components, their expected dot product is $\mathbb{E}[\mathbf{a} \cdot \mathbf{b}] = 0$ with standard deviation $\sim 1/\sqrt{D}$. For $D = 10,000$, this yields standard deviation ~ 0.01, meaning random vectors are orthogonal to within $\sim$1%.

This mathematical property provides natural error tolerance. If a vector $\mathbf{v}$ is corrupted by random noise $\boldsymbol{\eta}$, the corrupted vector $\mathbf{v}' = \mathbf{v} + \boldsymbol{\eta}$ remains similar to the original: $\mathbf{v} \cdot \mathbf{v}' \approx |\mathbf{v}|^2 \gg \mathbf{v} \cdot \boldsymbol{\eta} \approx 0$. As long as noise is not too large, the geometric relationships that encode information are preserved.

For biological systems with inherent stochasticity—transcriptional noise, protein folding fluctuations, neural firing variability—this noise tolerance is essential. HDC's graceful degradation aligns naturally with biological robustness: systems maintain approximate function despite molecular-level uncertainty.

3.2 Protein-Based HDC

3.2.1 Conformational Vectors

Protein conformations can be represented as vectors in high-dimensional space defined by backbone and side chain coordinates. For a protein with N residues, the conformational state is specified by $3N$ atomic coordinates (or equivalently, $2N$ backbone dihedrals plus side chain rotamers). Reducing dimensionality through principal component analysis typically yields $\sim$10-100 dominant modes capturing most conformational variance.

Representing conformations as vectors enables direct application of HDC similarity operations. Two conformations $\mathbf{c}_1$ and $\mathbf{c}_2$ are similar if their dot product $\mathbf{c}_1 \cdot \mathbf{c}_2$ is large. Conformational transitions implement vector transformations: a protein responding to ligand binding transitions from state $\mathbf{c}_{\text{apo}}$ to $\mathbf{c}_{\text{bound}}$, traversing a path through conformational space.

Allosteric regulation implements context-dependent transformations. An allosteric effector modifies the protein's energy landscape, changing how inputs map to outputs. In HDC terms, the effector acts as a gating vector $\mathbf{g}$ that modulates the transformation: $\mathbf{c}_{\text{out}} = f(\mathbf{c}_{\text{in}}, \mathbf{g})$ where the function f depends on both input and gate. This implements conditional operations essential for complex computation.

3.2.2 PTMs as Binding Operations

Post-translational modifications implement HDC's binding operation by associating proteins with modification patterns. Consider a protein in conformational state $\mathbf{c}$ that undergoes phosphorylation at site k. The modified protein can be represented as:

$$\mathbf{p}_{\text{phospho}} = \mathbf{c} \otimes \mathbf{m}_k \tag{4}$$

where $\mathbf{m}_k$ is a modification vector representing phosphorylation at site k. Multiple modifications implement multi-way binding:

$$\mathbf{p}_{\text{multi}} = \mathbf{c} \otimes \mathbf{m}_{k_1} \otimes \mathbf{m}_{k_2} \otimes \cdots \otimes \mathbf{m}_{k_n} \tag{5}$$

The exponential scaling of PTM combinations (2^M for M binary modification sites) provides massive state space. Each modification pattern creates a distinct point in HDC space, enabling the protein to store complex information through its modification signature. Enzymes that recognize specific modification patterns (kinases, phosphatases, ubiquitin ligases) implement pattern-matching operations: they bind preferentially to proteins whose modification vectors are similar to their recognition motifs.

This framework explains combinatorial PTM codes observed in biological systems. Histone modifications, for example, implement a "code" where different combinations of methylation, acetylation, and phosphorylation recruit different chromatin remodeling complexes. In HDC terms, each histone modification pattern is a vector; chromatin remodelers bind based on vector similarity to their recognition preferences.

3.2.3 Complex Formation as Composition

Protein complex assembly naturally implements HDC composition through binding operations. If proteins A and B with conformational states $\mathbf{c}_A$ and $\mathbf{c}_B$ form a complex, the complex state can be represented as:

$$\mathbf{C}_{AB} = \mathbf{c}_A \otimes \mathbf{c}_B \tag{6}$$

This composite vector is approximately orthogonal to both constituents (assuming high dimensionality), representing a new entity with properties distinct from individual proteins. Higher-order complexes implement nested binding: a complex of proteins A, B, C is represented as $(\mathbf{c}_A \otimes \mathbf{c}_B) \otimes \mathbf{c}_C$.

The combinatorial complexity of multi-protein complexes provides exponential scaling: M proteins can form 2^M possible combinations (considering all possible subsets). Each combination occupies a distinct region of HDC space, enabling rich representational capacity. Conditional assembly—where complex formation depends on post-translational modifications or ligand binding—implements context-dependent binding operations.

3.3 Nucleic Acid HDC

3.3.1 Sequence Embeddings

DNA and RNA sequences can be embedded in continuous high-dimensional spaces through featurization. A common approach maps each nucleotide to a vector (e.g., A $\rightarrow$ [1,0,0,0], T $\rightarrow$ [0,1,0,0], G $\rightarrow$ [0,0,1,0], C $\rightarrow$ [0,0,0,1]), then represents sequences through concatenation, k-mer composition, or neural network embeddings trained to predict functional properties.

For HDC applications, thermodynamic and structural features provide useful embeddings. Each sequence s maps to a vector $\mathbf{v}(s)$ with dimensions representing: GC content, melting temperature, secondary structure free energy, protein binding affinities, and catalytic activity (for ribozymes). This embeds discrete sequence space (4^L points) into continuous $\mathbb{R}^D$, enabling similarity operations based on functional properties rather than sequence identity alone.

Strand displacement reactions naturally implement vector transformations. Consider a toehold exchange reaction where input strand I displaces incumbent strand X from complex $X : T$, releasing X and forming $I : T$. In HDC terms: input vector $\mathbf{v}_I$ binds with toehold vector $\mathbf{v}_T$ (binding operation), producing output vector $\mathbf{v}_X$ (which was previously bundled in complex $X : T$). Cascades of such reactions implement sequential transformations through HDC space.

3.3.2 Concentration-Based Bundling

Multiple DNA/RNA species in solution naturally implement bundling through concentration-weighted superposition. If species $i = 1, \ldots, n$ with embeddings $\mathbf{v}_i$ are present at concentrations c_i, the system state is:

$$\mathbf{S} = \sum_{i=1}^{n} c_i \mathbf{v}_i \tag{7}$$

This weighted sum represents the collective ensemble. Reactions that consume or produce species modify the system vector: a reaction consuming species j implements $\mathbf{S} \to \mathbf{S} - \Delta c_j \mathbf{v}_j$. Catalytic reactions that amplify specific species implement selective bundling: species matching catalyst recognition motifs are amplified preferentially.

Recent demonstrations of DNA-based neural networks exploit concentration-based encoding explicitly [Cherry & Qian, 2018]. Strand concentrations represent neuron activations, and displacement reactions implement weighted connections. Winner-take-all networks emerge from competitive binding where strands compete for limited toehold sites, naturally implementing sparse activation patterns characteristic of HDC systems.

3.4 Neural HDC

3.4.1 Population Coding as Distributed Representation

Neural population responses naturally implement HDC's distributed representation. A stimulus s evokes a population response $\mathbf{r}(s) = (r_1, r_2, \ldots, r_N)$ where r_i is the firing rate of neuron i. This response vector lives in $\mathbb{R}^N$ and represents the stimulus in a distributed manner: no single neuron uniquely encodes s, but the full population pattern is distinctive.

The mapping from stimulus space to neural space preserves similarity structure. Similar stimuli evoke similar population responses (nearby vectors), while dissimilar stimuli evoke dissimilar responses (distant vectors). This naturally implements HDC's similarity-preserving encoding. Moreover, population codes exhibit noise tolerance: trial-to-trial variability in individual neurons averages out across the population, yielding reliable collective codes despite single-neuron unreliability.

Experimental evidence from multiple brain areas demonstrates high-dimensional population coding. Visual cortex responses to natural images span hundreds of effective dimensions [Stringer et al., 2019]. Hippocampal place cell populations encode spatial location through hundreds to thousands of neurons, with each neuron contributing partially overlapping information. Motor cortex population activity exhibits low-dimensional dynamics governing movement, but the full state space is high-dimensional.

3.4.2 Synaptic Integration as Bundling

Synaptic integration implements HDC bundling through weighted summation of inputs. A neuron receiving inputs from presynaptic neurons with activities $\mathbf{a} = (a_1, \ldots, a_M)$ and synaptic weights $\mathbf{w} = (w_1, \ldots, w_M)$ computes:

$$V = \sum_{i=1}^{M} w_i a_i = \mathbf{w} \cdot \mathbf{a} \tag{8}$$

This dot product measures similarity between the input pattern $\mathbf{a}$ and the neuron's preferred pattern (represented by weights $\mathbf{w}$). Neurons fire strongly when inputs match their weight vector—implementing pattern matching through similarity measurement.

Synaptic plasticity modifies weights to store associations. Hebbian learning strengthens connections between coactive neurons, creating overlap between their vector representations.

Long-term potentiation (LTP) increases synaptic weights when pre- and postsynaptic activities are correlated:

$$\Delta w_i \propto \langle a_i \cdot r_{\text{post}} \rangle \tag{9}$$

This implements associative memory: patterns that frequently occur together become associated through weight modification. In HDC terms, Hebbian plasticity creates similarity between vectors representing related concepts, enabling approximate retrieval through partial cues.

3.4.3 Temporal Binding Through Sequential Activation

Neural sequences—ordered activation of cell populations over time—encode temporal information through dynamics. Sequential activation patterns in hippocampus during spatial navigation, motor cortex during movement preparation, and prefrontal cortex during working memory tasks demonstrate neural sequence encoding.

In HDC terms, sequences require binding with temporal markers to distinguish order. Representing sequence (A, B, C) requires:

$$\mathbf{v}_{\text{seq}} = (\mathbf{v}_A \otimes \mathbf{t}_1) \oplus (\mathbf{v}_B \otimes \mathbf{t}_2) \oplus (\mathbf{v}_C \otimes \mathbf{t}_3) \tag{10}$$

where $\mathbf{t}_i$ are position vectors. Neural circuits implement similar operations through phase coding (theta sequences), sequential chain dynamics, or synaptic delays that create temporal context.

The high dimensionality of spatiotemporal codes (spatial dimensions from neuron identity times temporal dimensions from timing precision) provides enormous capacity. This enables representation of complex sequences with arbitrary length and hierarchical structure, supporting cognitive functions like language processing, planning, and episodic memory.

4 Computational Framework

4.1 Multi-Scale Modeling

Biological HDC systems span multiple spatiotemporal scales, requiring multi-scale modeling approaches. At the molecular scale (femtoseconds to microseconds, nanometers), quantum mechanics and molecular dynamics govern atomic interactions. At the biochemical scale (microseconds to minutes, nanometers to micrometers), reaction kinetics and diffusion determine molecular populations. At the cellular scale (minutes to hours, micrometers to millimeters), gene regulation and signaling networks control cell state. At the tissue scale (hours to days, millimeters to centimeters), cell-cell interactions and spatial organization determine collective behavior.

Ordinary differential equations (ODEs) describe temporal dynamics of molecular concentrations and protein states. A typical model for protein interactions and modifications has form:

$$\frac{d[\text{Species}_i]}{dt} = \sum_j k_j^+ \prod_k [\text{Reactant}_{jk}] - \sum_j k_j^- [\text{Species}_i] \tag{11}$$

Gene regulatory networks, metabolic pathways, and signaling cascades naturally map onto ODE systems. Recent advances in fast ODE solvers enable simulation of networks with thousands of species and tens of thousands of reactions in near-real-time [Cohen et al., 2014].

Partial differential equations (PDEs) incorporate spatial structure, capturing diffusion of signaling molecules, wave propagation of neural activity, and pattern formation through reaction-diffusion mechanisms. A typical reaction-diffusion system has form:

$$\frac{\partial u}{\partial t} = D\nabla^2 u + f(u, v), \quad \frac{\partial v}{\partial t} = D\nabla^2 v + g(u, v) \tag{12}$$

Stochastic differential equations (SDEs) incorporate molecular noise, modeling fluctuations in small-copy-number systems like gene expression. The Chemical Langevin equation:

$$d[\text{Species}_i] = \mu_i dt + \sigma_i dW \tag{13}$$

includes both deterministic drift (μ_i) and stochastic fluctuations ($\sigma_i dW$, where W is a Wiener process).

4.2 Key Algorithms

Several algorithmic advances provide essential tools for analyzing biological HDC systems:

Large-scale linear system solvers: Many biological modeling problems reduce to solving large sparse linear systems $Ax = b$ where A represents interaction networks. Recent nearly-linear-time algorithms for symmetric diagonally dominant (SDD) systems [Cohen et al., 2014] enable efficient simulation of gene regulatory networks and metabolic pathways with thousands of components. These solvers exploit network sparsity (biological interactions are typically local) to achieve computational complexity nearly linear in problem size.

Network-flow optimization: Metabolic networks, signaling cascades, and neural circuits can be modeled as flow problems where resources (metabolites, signals, information) flow through networks of nodes (enzymes, proteins, neurons) and edges (reactions, interactions, synapses). Recent maximum-flow algorithms achieve nearly-linear time complexity [Chen et al., 2022], enabling optimization over networks with millions of nodes. Applications include metabolic flux analysis, signal propagation optimization, and neural information routing.

Topological data analysis (TDA): TDA extracts robust geometric features from high-dimensional data through persistent homology and related techniques [Carlsson, 2009]. For biological HDC systems, TDA identifies: connected components (distinct cell types or functional states), loops (cyclic pathways or regulatory circuits), and voids (gaps in coverage or forbidden states). TDA provides noise-robust analysis complementing traditional statistical methods, particularly valuable for high-dimensional biological data with complex geometry.

4.3 Machine Learning Integration

4.3.1 Predictive Modeling

Machine learning models trained on biological data predict system behavior without requiring complete mechanistic understanding. Deep neural networks learn complex input-output

mappings from protein sequence to function, gene expression to phenotype, or neural activity to behavior. For HDC systems, neural networks trained on high-dimensional biological data can learn embeddings that preserve functional similarity, enabling prediction and design.

Graph neural networks (GNNs) process network representations directly, making them natural for biological systems structured as graphs (gene regulatory networks, protein interaction networks, neural circuits). GNNs learn node and edge representations through message passing, enabling prediction of network properties, missing interactions, or dynamic behavior.

4.3.2 Reinforcement Learning for Control

Controlling biological HDC systems requires learning from interaction, as complete mechanistic models are rarely available. Reinforcement learning (RL) treats system control as sequential decision-making: at each time step, observe state (molecular concentrations, gene expression, neural activity), take action (adjust parameters, deliver stimuli), and receive reward (progress toward desired state).

Recent applications of RL to biological systems include metabolic engineering (optimizing culture conditions and genetic modifications), optogenetic control of neural circuits (learning stimulation patterns to achieve desired activity), and protein design (searching sequence space for improved function). The high dimensionality of biological state spaces aligns well with deep RL methods that use neural networks to approximate value functions and policies in high-dimensional spaces.

5 Synthetic Biology Applications

5.1 Gene Circuit Design

HDC principles provide guidance for designing synthetic gene circuits with improved robustness and scalability. Traditional genetic circuit design mimics electronic circuits, implementing logic gates through transcription factors and promoters. However, biological noise, crosstalk, and context dependence limit the reliability of such designs.

An HDC-inspired approach distributes computation across multiple genetic elements rather than relying on single-element logic. For example, a distributed gene circuit for pathogen detection could encode pathogen signatures as high-dimensional vectors where each dimension corresponds to expression of a specific pathogen-associated gene. Detection is implemented through vector similarity: measured gene expression is compared to stored pathogen vectors, with detection triggered when similarity exceeds threshold.

This distributed approach provides robustness: no single gene is critical; detection depends on overall pattern similarity. Noise in individual measurements averages out across many dimensions. Novel pathogens with partial similarity to known pathogens still produce intermediate similarity scores, enabling generalization beyond training data.

Implementation strategy: Design synthetic promoters responsive to multiple transcription factors (multi-input sensing). Express combinations of transcription factors representing HDC dimensions. Implement vector similarity through weighted summation at

downstream promoters whose activity depends on collective transcription factor binding. Tune weights through directed evolution or computational optimization.

5.2 Cellular Programming

HDC provides a framework for programming cellular behaviors through high-dimensional control of gene expression. Cell state can be viewed as a point in gene expression space (tens of thousands of dimensions). Desired behaviors correspond to trajectories through this space. Programming cells means designing interventions (gene circuits, delivered factors, environmental stimuli) that guide trajectories toward target states.

For synthetic biology applications in biomanufacturing, cellular programming could optimize metabolic flux by maintaining gene expression in high-productivity regions of expression space. For therapeutic cells, programming could implement context-dependent behaviors where cells respond appropriately to complex physiological conditions encoded as high-dimensional biochemical signatures.

Case study: Stem Cell Differentiation

Stem cell differentiation represents a natural application of HDC principles. Pluripotent stem cells occupy a high-dimensional epigenetic state characterized by accessible chromatin at developmental genes. Differentiation involves traversing a high-dimensional landscape toward specialized cell types, with chromatin state progressively restricting as cells commit to lineages.

Traditional differentiation protocols apply sequential factor combinations identified empirically. An HDC-inspired approach formulates differentiation as optimization in high-dimensional epigenetic space: starting from pluripotent state $\mathbf{s}_0$, find trajectory $\{\mathbf{s}_t\}$ that reaches target state $\mathbf{s}_{\text{target}}$ efficiently while avoiding unwanted states.

Recent work on stem cell engineering demonstrates this approach's utility. By characterizing differentiation landscapes through single-cell RNA sequencing (providing high-dimensional state readout) and applying machine learning to identify optimal factor combinations (control in high-dimensional space), differentiation efficiency can be dramatically improved. Published results show 95% efficiency for difficult lineages, reduced timeframes from weeks to days, and improved cell quality compared to conventional protocols.

The key insight is treating cell state as a high-dimensional vector and differentiation as trajectory optimization. HDC principles—particularly noise tolerance through distributed representation—explain why this approach succeeds: cells exhibit substantial cell-to-cell variability, but average trajectories through high-dimensional space converge reliably to target states.

5.3 Distributed Sensing

Biological HDC naturally implements distributed sensing where multiple sensor modalities combine to produce robust detection. Consider a synthetic cell designed to detect environmental toxins. Traditional approaches use single-target sensors (e.g., transcription factor responsive to heavy metals). HDC-inspired distributed sensing uses multiple sensors whose collective pattern indicates toxin presence.

Each sensor contributes one dimension to a sensing vector $\mathbf{v}_{\text{sense}} = (s_1, s_2, \ldots, s_n)$ where s_i is the activity of sensor i. Toxin signatures are stored as reference vectors $\{\mathbf{v}_{\text{tox}_k}\}$. Detection compares sensing vector to references through similarity:

$$\text{Score}_k = \mathbf{v}_{\text{sense}} \cdot \mathbf{v}_{\text{tox}_k} \tag{14}$$

Detection is triggered when any score exceeds threshold. This provides advantages: robustness to sensor failure (individual sensor malfunction has minimal impact), discrimination between toxins (different toxins have distinct reference vectors), and generalization (novel toxins with partial similarity still produce intermediate scores).

Implementation: Engineer synthetic promoters responsive to different environmental signals (heavy metals, oxidative stress, membrane damage, metabolic disruption). Design downstream integration circuits that sum weighted sensor outputs. Train weights by exposing cells to known toxin panels and optimizing discrimination. Deploy in environmental monitoring or industrial safety applications.

5.4 Adaptive Cellular Behaviors

HDC enables programming adaptive behaviors where cells learn from experience. Incorporating simple learning rules into genetic circuits allows cells to adjust responses based on environmental history, implementing primitive associative memory.

For example, a therapeutic cell that learns which inflammatory signals predict disease activity could adjust its response threshold accordingly. Initial exposure to inflammatory signals (cytokines, damage-associated molecular patterns) activates sensing circuits. If disease markers subsequently appear, a learning circuit strengthens the association between inflammatory pattern and therapeutic response. Over time, the cell adapts its sensitivity based on patient-specific disease patterns.

Implementation through epigenetic memory: Epigenetic modifications (DNA methylation, histone modifications) provide stable but modifiable memory. Synthetic circuits that recruit epigenetic modifiers to specific loci in response to signal combinations can implement associative learning. Repeated co-occurrence of signals A and B induces chromatin modifications that increase subsequent response to A alone—classic Hebbian association.

This approach combines HDC's distributed representation (multiple signals, multiple loci) with biological memory mechanisms (epigenetic modifications). The result is cells that adapt to complex, variable environments—essential for therapeutic applications where patient contexts differ and diseases evolve over time.

6 Discussion

6.1 Implications for Understanding Biological Computation

The HDC framework provides new perspectives on how biological systems process information. Rather than viewing biological computation as an approximate implementation of digital logic, HDC suggests that biological systems may be optimized for different computational paradigms altogether—ones that leverage high-dimensional continuous state spaces,

noise tolerance through distributed representation, and similarity-based operations rather than exact symbolic manipulation.

This perspective helps explain several puzzling features of biological computation. **Why is biological computation so noisy?** From a digital logic viewpoint, biological noise is a bug to be minimized. From an HDC viewpoint, moderate noise is acceptable because high-dimensional distributed representations tolerate perturbations that would break low-dimensional symbolic codes. **Why are biological regulatory networks so complex?** HDC systems benefit from high dimensionality—more dimensions enable finer distinctions, richer representations, and more robust codes. Complexity provides computational capacity. **Why do cells exhibit apparently redundant regulatory mechanisms?** Redundancy provides robustness in HDC: distributing information across many dimensions means no single component is critical.

The HDC lens also connects biological computation to emerging frameworks in neuroscience and cognitive science. **Predictive processing** posits that brains implement approximate Bayesian inference through hierarchical prediction and error correction—naturally formulated in high-dimensional latent spaces. **Active inference** extends this to action selection, viewing behavior as inference in joint perception-action spaces. Both frameworks align with HDC's emphasis on high-dimensional representations and similarity-based operations.

6.2 Design Principles for Synthetic Systems

The framework suggests several design principles for engineering synthetic biocomputing systems:

Embrace high dimensionality: Rather than minimizing system complexity, exploit high-dimensional encoding for robustness and capacity. Design proteins with many modifiable sites, circuits with many parallel pathways, and sensors with many detection modalities.

Use distributed representations: Avoid designs where single components are critical. Distribute information across multiple elements so that partial failures degrade gracefully rather than causing catastrophic collapse.

Exploit noise tolerance: Design systems assuming noise will be present rather than attempting perfect noise elimination. HDC's inherent noise tolerance reduces engineering burden and enables operation in variable environments.

Implement similarity-based operations: Biological pattern recognition naturally operates through similarity measurement (binding affinity, transcription factor cooperativity, neural population tuning). Design computational tasks to leverage this rather than fighting against it.

Hierarchical organization: Combine local operations (protein conformational changes, transcription factor binding) with global coordination (hormonal signals, chromatin state) to create multi-scale computational architectures matching biological organization.

6.3 Challenges and Limitations

Despite the framework's promise, significant challenges remain:

Measurement difficulties: Current technologies struggle to read out high-dimensional biological states in real-time. Single-cell RNA sequencing provides snapshots of gene ex-

pression space but is destructive and slow. Calcium imaging reads neural population activity but is limited to accessible tissues with fluorescent labels. Developing non-invasive, high-throughput, real-time measurement technologies is critical for practical biocomputing applications.

Readout latency: Many biological measurements (sequencing, mass spectrometry, immunoassays) require minutes to hours, precluding real-time control. This limits biocomputing to applications tolerating slow feedback. Future work should develop faster readout methods, possibly combining biological computation with electronic readout for hybrid systems.

Integration complexity: Combining multiple biological substrates (proteins, DNA, cells) into unified systems requires orchestrating interactions across disparate timescales, spatial scales, and chemical environments. Current synthetic biology tools provide good control over individual substrate types but limited integration capabilities. Standardized interfaces and modular design frameworks would accelerate progress.

Theoretical gaps: While HDC provides a useful framework, quantitative predictions remain difficult. How much noise can a given biological HDC system tolerate? What dimensionality is sufficient for a specific task? How do biological learning rules relate to optimal HDC learning? Addressing these questions requires tighter integration of theory and experiment.

Ethical considerations: As biological computing systems become more sophisticated, particularly those incorporating neural tissue, ethical questions arise about moral status, consciousness, and appropriate uses. Developing ethical frameworks concurrently with technical capabilities is essential for responsible innovation.

7 Conclusion

We have presented a comprehensive framework for understanding biological computation through the lens of hyperdimensional computing. The core insight is that biological systems naturally encode information in very high-dimensional spaces—protein conformational space, gene expression space, neural population space—and that this high-dimensionality is not a limitation but rather a computational resource. The mathematical properties of high-dimensional spaces (approximate orthogonality of random vectors, noise tolerance through distributed representation, efficient similarity operations) align remarkably well with observed characteristics of biological computation (robustness to noise, graceful degradation, associative memory).

Key contributions include:

Formalization of biological HDC: We have characterized the effective dimensionality of proteins (10^2-10^3 dimensions), nucleic acids (10^3-10^6 dimensions), and neural populations (10^6-10^9 dimensions), and shown how fundamental biological mechanisms—allosteric regulation, strand displacement, synaptic integration—implement core HDC operations of binding, bundling, and permutation.

Synthetic amino acid expansion: Analysis of how genetic code expansion extends protein encoding capacity, providing explicit engineering handles for designing protein-based HDC systems. Increasing the amino acid alphabet from 20 to 30 increases encoding density

by 14% while introducing novel chemical functionalities unavailable to natural proteins.

Multi-substrate integration: Framework for integrating diverse biological substrates into unified computational architectures, exploiting complementary strengths (proteins: fast dynamics, complex chemistry; nucleic acids: stable memory, high density; neural tissue: massive parallelism, adaptive learning).

Design principles: General principles for engineering synthetic biological computing systems that exploit HDC properties, including distributed representation for robustness, similarity-based operations leveraging natural biological mechanisms, and hierarchical organization matching biological structure.

Synthetic biology applications: Demonstration of framework utility for gene circuit design (distributed sensing, noise-robust detection), cellular programming (stem cell differentiation as trajectory optimization in high-dimensional epigenetic space), and adaptive behaviors (cells that learn from experience through epigenetic memory).

The applications envisioned—distributed biosensing, context-dependent therapeutics, adaptive cellular behaviors, and engineered tissue—leverage biology's unique computational strengths rather than attempting to replicate silicon-based architectures. Biology offers massive parallelism, energy efficiency, molecular specificity, adaptive self-organization, and natural operation in physiological contexts—advantages complementary to electronic computation.

Fundamental questions remain about the computational complexity classes efficiently addressed by biological HDC substrates, scaling laws governing capacity versus system size, and information-theoretic limits on biological operations. Addressing these requires continued interplay between theoretical analysis, computational modeling, and experimental validation. The synthesis of molecular biology, synthetic biology, computer science, and applied mathematics represented by biological HDC exemplifies the interdisciplinary integration necessary for 21st-century biotechnology.

If successfully developed, biological HDC systems would represent a genuinely new information processing paradigm—one that complements rather than replaces electronic computation, drawing on billions of years of evolutionary optimization while introducing synthetic innovations. Beyond practical applications, this framework may reshape our understanding of computation itself, revealing it as a phenomenon not limited to engineered devices but emerging naturally from the complex dynamics of living systems. The question is not whether biology computes, but rather how we can better understand and harness the computational principles that life has already discovered.

Acknowledgments

I thank the open-source scientific community for tools and resources that enabled this work. Computational resources were provided through publicly available platforms. This work received no specific grant from funding agencies.

References

Adamatzky, A. (Ed.). (2016). *Advances in Unconventional Computing: Volume 1: Theory.* Springer.

Carlsson, G. (2009). Topology and data. *Bulletin of the American Mathematical Society*, 46(2), 255-308.

Chen, L., Kyng, R., Liu, Y. P., Peng, R., Gutenberg, M. P., & Sachdeva, S. (2022). Maximum flow and minimum-cost flow in almost-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 612-623).

Cherry, K. M., & Qian, L. (2018). Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks. *Nature*, 559(7714), 370-376.

Chin, J. W. (2017). Expanding and reprogramming the genetic code. *Nature*, 550(7674), 53-60.

Cohen, M. B., Kyng, R., Miller, G. L., Pachocki, J. W., Peng, R., Rao, A. B., & Xu, S. C. (2014). Solving SDD linear systems in nearly $m \log^{1/2} n$ time. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing* (pp. 343-352).

Huang, P. S., Boyken, S. E., & Baker, D. (2016). The coming of age of de novo protein design. *Nature*, 537(7620), 320-327.

Kanerva, P. (2009). Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2), 139-159.

Nielsen, A. A. K., Der, B. S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E. A., ... & Voigt, C. A. (2016). Genetic circuit design automation. *Science*, 352(6281), aac7341.

Qian, L., & Winfree, E. (2011). Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034), 1196-1201.

Rahimi, A., Benatti, S., Kanerva, P., Gupta, R., & Benini, L. (2016). Hyperdimensional biosignal processing: A case study for EMG-based hand gesture recognition. In *2016 IEEE International Conference on Rebooting Computing (ICRC)* (pp. 1-8).

Stringer, C., Pachitariu, M., Steinmetz, N., Reddy, C. B., Carandini, M., & Harris, K. D. (2019). Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437), eaav7893.