PATENT APPLICATION

COMPUTER-IMPLEMENTED METHOD FOR COMPUTING PROTEIN FOLDING JAMMING FREQUENCY FROM FIRST PRINCIPLES

PROVISIONAL PATENT APPLICATION

Inventor:	Jonathan Washburn
Email:	washburn.jonathan@gmail.com

Filing Date:	January 17, 2026
Application Type:	Utility Patent (Provisional)
Related Applications:	Patents 001–003 (Method, Apparatus, Verification)

A computer-implemented method for calculating the optimal frequency for modulating protein folding rates based on the golden ratio (φ) timescale ladder, including generation of operating setpoints, frequency sweep windows, isotope-shifted frequencies, and acceptance criteria for experimental validation.

Contents

ABSTRACT OF THE DISCLOSURE 3

1 BACKGROUND OF THE INVENTION 4

1.1 Field of the Invention 4

1.2 Description of Related Art 4

1.2.1 Empirical Frequency Determination 4

1.2.2 Limitations of Prior Art 4

1.2.3 The Need for First-Principles Computation 4

1.3 The Recognition Science Framework 5

1.3.1 The Golden Ratio Timescale Ladder 5

1.3.2 The Molecular Gate Rung 5

1.3.3 Machine-Verified Derivations 6

1.4 Objects of the Invention 6

2 SUMMARY OF THE INVENTION 7

2.1 General Statement of the Invention 7

2.2 Core Computation Algorithm 7

2.3 Extended Computation with Isotope Shift 7

2.4 Output Types 8

3 BRIEF DESCRIPTION OF DRAWINGS 9

4 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS 10

4.1 System Architecture 10

4.1.1 Overview 10

4.1.2 Implementation Platforms 10

4.2 Input Parameters 11

4.2.1 Required Parameters 11

4.2.2 Optional Parameters 11

4.2.3 Derived Input: Fundamental Timescale 11

4.3 Core Computation 12

4.3.1 Step 1: Compute Golden Ratio 12

4.3.2 Step 2: Compute Rung Timescale 12

4.3.3 Step 3: Compute Jamming Frequency 12

4.4 Extended Computations 12

4.4.1 Isotope-Shifted Frequency 12

4.4.2 Frequency Sweep Windows 13

4.4.3 Operating Setpoints 14

4.4.4 Acceptance Criteria 14

4.5 Rung Selection 15

4.5.1 Default Rung: Molecular Gate (n=19) 15

4.5.2 Alternative Rungs 15

4.6 Output Formats 15

4.6.1 JSON Output 15

4.6.2 Instrument Control File 16

4.6.3 CSV Output 17

4.7 Validation Integration 17

4.7.1 Pre-Experiment Validation 17

4.7.2 Post-Experiment Validation 18

4.8 Implementation Examples 19

4.8.1 Python Implementation 19

4.8.2 C Implementation 20

4.9 API Specification 21

4.9.1 REST API Endpoint 21

4.10 Error Handling 21

5 CLAIMS **23**

5.1 Core Computation Claims 23

5.2 Setpoint Generation Claims 23

5.3 Acceptance Criteria Claims 24

5.4 System Claims 25

5.5 Integration Claims 25

ABSTRACT **27**

INVENTOR DECLARATION **28**

ABSTRACT OF THE DISCLOSURE

A computer-implemented method for calculating the frequency for modulating protein folding rates from first principles. The method comprises: (a) computing a fundamental timescale τ_0 from physical constants; (b) computing the golden ratio $\varphi = (1 + \sqrt{5})/2$; (c) computing a molecular gate timescale $\tau_n = \tau_0 \times \varphi^n$ for a selected rung number n; (d) computing a jamming frequency $f_{\text{jam}} = 1/\tau_n$; (e) optionally computing an isotope-shifted frequency $f_{\text{D}_2\text{O}} = f_{\text{jam}}/\sqrt{2}$; and (f) outputting one or more of: the jamming frequency, frequency sweep windows, operating setpoints for irradiation apparatus, acceptance criteria for experimental validation, and calibration parameters. The method implements the Recognition Science framework for deriving physical constants from the golden ratio, and outputs are suitable for direct use in controlling protein folding modulation apparatus. The computation may be performed on a general-purpose computer, embedded controller, or cloud computing platform, and outputs may be formatted as instrument control files, experimental protocols, or validation specifications. Machine-verified proofs ensure mathematical correctness of the underlying derivations.

Keywords: computer-implemented method, jamming frequency, golden ratio, phi-ladder, timescale computation, operating setpoints, frequency sweep, isotope shift, calibration

1 BACKGROUND OF THE INVENTION

1.1 Field of the Invention

The present invention relates generally to computer-implemented methods for scientific computation, and more specifically to methods for calculating electromagnetic frequencies for biological applications based on first-principles physical theory.

1.2 Description of Related Art

1.2.1 Empirical Frequency Determination

Prior art approaches to determining optimal frequencies for electromagnetic effects on biological systems rely on empirical methods:

(a) **Exhaustive frequency sweeps:** Scanning large frequency ranges experimentally to identify effects. This approach is time-consuming, expensive, and may miss narrow resonances.

(b) **Literature-based selection:** Choosing frequencies based on published studies. This approach lacks predictive power for novel applications.

(c) **Dielectric spectroscopy:** Measuring absorption spectra to identify frequencies of interest. This identifies absorption features but not necessarily biologically active frequencies.

(d) **Molecular dynamics simulation:** Computationally simulating protein dynamics to identify characteristic timescales. This is computationally expensive and may not accurately predict experimental frequencies.

1.2.2 Limitations of Prior Art

1.2.3 The Need for First-Principles Computation

What is needed is a computational method that:

(1) Derives the optimal frequency from **first principles**, not empirical fitting;

Prior Approach	**Limitations**
Exhaustive sweeps	Time-consuming, expensive, may miss narrow features
Literature-based	No predictive power, protein-specific
Dielectric spectroscopy	Identifies absorption, not biological activity
Molecular dynamics	Computationally expensive, accuracy limited

Table 1: Limitations of prior art frequency determination methods

(2) Is **computationally efficient** (milliseconds, not hours);

(3) Provides **predictive power** for novel proteins and conditions;

(4) Outputs **actionable parameters** for experimental apparatus;

(5) Includes **validation criteria** to confirm predictions experimentally.

1.3 The Recognition Science Framework

1.3.1 The Golden Ratio Timescale Ladder

The Recognition Science framework provides a first-principles derivation of physical timescales based on the golden ratio $\varphi = (1 + \sqrt{5})/2 \approx 1.618$. The key insight is that molecular timescales form a discrete "ladder" with rungs separated by powers of φ:

$$\tau_n = \tau_0 \times \varphi^n \tag{1}$$

where τ_0 is a fundamental timescale and n is the rung number.

1.3.2 The Molecular Gate Rung

For protein folding, the relevant rung is $n = 19$, corresponding to the molecular gate timescale:

$$\tau_{19} = \tau_0 \times \varphi^{19} \approx 68 \text{ ps} \tag{2}$$

This is the unique rung in the biologically relevant 50–100 ps range.

1.3.3 Machine-Verified Derivations

The mathematical derivations underlying the present invention have been formally verified using the Lean 4 theorem prover with the Mathlib library:

- `phi_pos`: $\varphi > 0$
- `phi_sq_eq`: $\varphi^2 = \varphi + 1$
- `rung19_unique`: τ_{19} is unique in 50–100 ps window
- `direct_jamming_approx`: $f_{\text{jam}} \in (12, 17)$ GHz
- `d2o_jamming_approx`: $f_{\text{D}_2\text{O}} \in (8, 13)$ GHz

1.4 Objects of the Invention

It is an object of the present invention to provide a computer-implemented method that:

(1) Computes the jamming frequency from first principles in constant time;

(2) Generates operating setpoints for protein folding modulation apparatus;

(3) Outputs frequency sweep windows for experimental validation;

(4) Computes isotope-shifted frequencies for D_2O verification;

(5) Generates acceptance criteria for validating experimental results.

2 SUMMARY OF THE INVENTION

2.1 General Statement of the Invention

The present invention provides a computer-implemented method for calculating the frequency for modulating protein folding, comprising:

(a) Computing the golden ratio $\varphi = (1 + \sqrt{5})/2$;

(b) Computing a fundamental timescale τ_0;

(c) Computing a molecular gate timescale $\tau_n = \tau_0 \times \varphi^n$;

(d) Computing a jamming frequency $f_{\text{jam}} = 1/\tau_n$;

(e) Optionally computing derived quantities including isotope-shifted frequencies, sweep windows, and acceptance criteria;

(f) Outputting the computed values in a format suitable for controlling experimental apparatus.

2.2 Core Computation Algorithm

Algorithm 1: Jamming Frequency Computation

```
FUNCTION ComputeJammingFrequency(n, tau_0):
    phi = (1 + sqrt(5)) / 2         // Golden ratio
    tau_n = tau_0 * phi^n           // Rung timescale
    f_jam = 1 / tau_n               // Jamming frequency
    RETURN f_jam
END FUNCTION
```

2.3 Extended Computation with Isotope Shift

Algorithm 2: Extended Frequency Computation with Isotope Shift

```
FUNCTION ComputeFrequencySet(n, tau_0, includeIsotope):
    phi = (1 + sqrt(5)) / 2
    tau_n = tau_0 * phi^n
    f_H2O = 1 / tau_n
    IF includeIsotope THEN
        f_D2O = f_H2O / sqrt(2)
    END IF
    RETURN {f_H2O, f_D2O}
END FUNCTION
```

2.4 Output Types

The method generates multiple output types:

(1) **Jamming frequency:** f_{jam} in Hz or GHz

(2) **Isotope-shifted frequency:** $f_{\text{D}_2\text{O}}$ in Hz or GHz

(3) **Frequency sweep window:** $(f_{\text{min}}, f_{\text{max}})$ for experimental validation

(4) **Operating setpoints:** Frequency, power, and timing parameters for apparatus

(5) **Acceptance criteria:** Bounds and ratios for validating experimental results

(6) **Calibration parameters:** τ_0, φ, and derived constants

3 BRIEF DESCRIPTION OF DRAWINGS

Figure 1: System Architecture

A block diagram showing the computation system architecture, including input parameters, computation engine, and output generation.

Figure 2: Computation Flowchart

A flowchart showing the steps of the core frequency computation algorithm.

Figure 3: Output Format Examples

Examples of output formats including JSON, CSV, and instrument control files.

Figure 4: φ-Ladder Visualization

A visualization of the golden ratio timescale ladder with rung 19 highlighted.

Figure 5: Validation Workflow

A workflow diagram showing how computed values are used to validate experimental results.

4 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

4.1 System Architecture

4.1.1 Overview

The computer-implemented method of the present invention is executed on a computing system comprising:

(a) A processor capable of floating-point arithmetic;

(b) Memory for storing intermediate values;

(c) Input interface for receiving computation parameters;

(d) Output interface for delivering computed results.

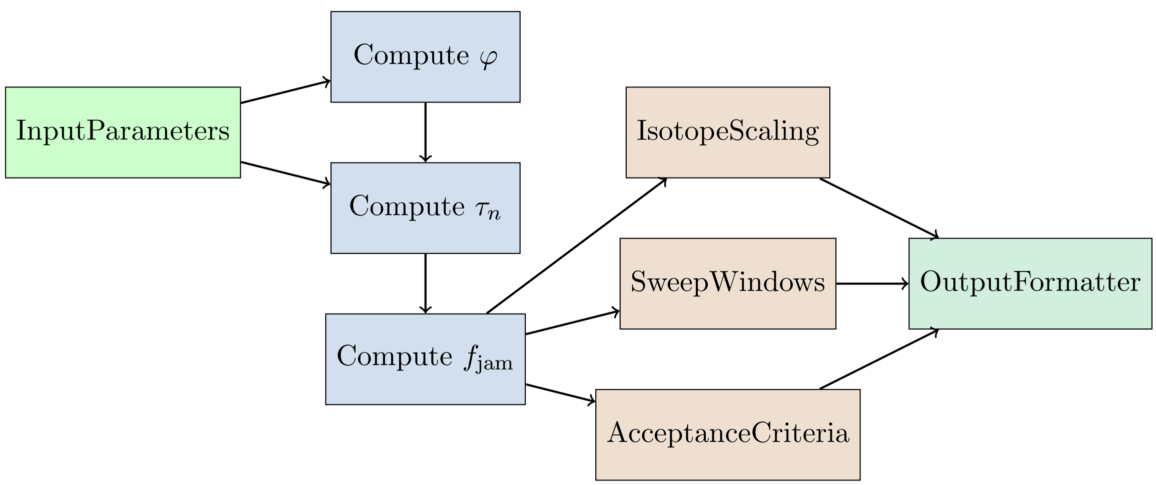

Figure 1: System architecture for jamming frequency computation

4.1.2 Implementation Platforms

The method may be implemented on:

(1) **General-purpose computer:** Desktop or laptop running Python, C++, MATLAB, or other scientific computing environment.

(2) **Embedded controller:** Microcontroller or FPGA integrated into irradiation apparatus.

(3) **Cloud computing platform:** Web service providing frequency computation as API.

(4) **Mobile device:** Smartphone or tablet application for field use.

(5) **Laboratory information management system (LIMS):** Integration with laboratory workflow software.

4.2 Input Parameters

4.2.1 Required Parameters

Parameter	**Symbol**	**Default Value**	**Units**
Rung number	n	19	dimensionless
Fundamental timescale	τ_0	7.33×10^{-15}	seconds

Table 2: Required input parameters

4.2.2 Optional Parameters

Parameter	**Symbol**	**Default Value**	**Description**
Include isotope shift	—	true	Compute D_2O frequency
Sweep margin	Δf	2 GHz	Window around f_{jam}
Tolerance	ϵ	5%	Acceptance tolerance
Output format	—	JSON	Output file format
Unit system	—	GHz	Frequency units

Table 3: Optional input parameters

4.2.3 Derived Input: Fundamental Timescale

The fundamental timescale τ_0 can be computed from Planck's constant:

$$\tau_0 = \frac{\hbar}{E_{\text{coh}}} \tag{3}$$

where $E_{\mathrm{coh}} = \varphi^{-5}$ in natural units (with appropriate conversion factors). For practical use, the default value $\tau_0 = 7.33 \times 10^{-15}$ s is recommended.

4.3 Core Computation

4.3.1 Step 1: Compute Golden Ratio

$$\varphi = \frac{1+\sqrt{5}}{2} = 1.6180339887... \tag{4}$$

Implementation note: Use extended precision (64-bit or 128-bit floating point) to maintain accuracy through subsequent power operations.

4.3.2 Step 2: Compute Rung Timescale

$$\tau_n = \tau_0 \times \varphi^n \tag{5}$$

For the default rung $n = 19$:

$$\tau_{19} = 7.33 \times 10^{-15} \times (1.618...)^{19} \approx 6.82 \times 10^{-11}\ \mathrm{s} = 68.2\ \mathrm{ps} \tag{6}$$

4.3.3 Step 3: Compute Jamming Frequency

$$f_{\mathrm{jam}} = \frac{1}{\tau_n} \tag{7}$$

For rung 19:

$$f_{\mathrm{jam}} = \frac{1}{68.2 \times 10^{-12}} \approx 14.66 \times 10^{9}\ \mathrm{Hz} = 14.66\ \mathrm{GHz} \tag{8}$$

4.4 Extended Computations

4.4.1 Isotope-Shifted Frequency

When D_2O operation is required:

$$f_{\mathrm{D_2O}} = \frac{f_{\mathrm{jam}}}{\sqrt{2}} \approx 0.7071 \times f_{\mathrm{jam}} \tag{9}$$

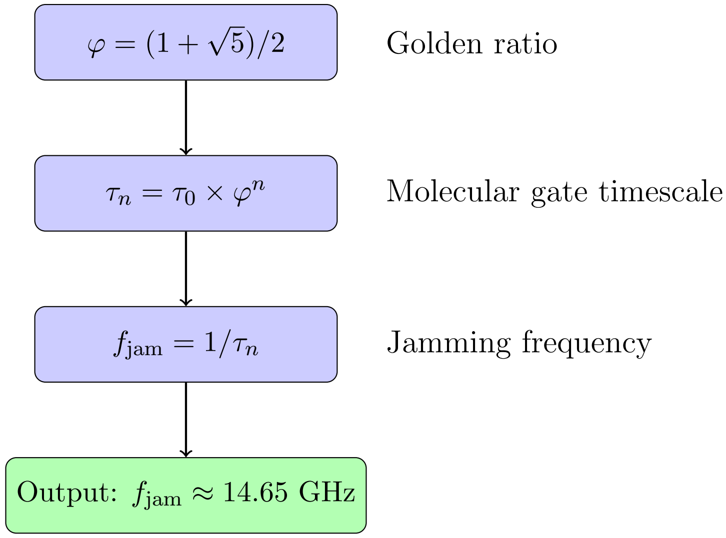

Figure 2: Core computation flowchart

For the default jamming frequency:

$$f_{D_2O} = \frac{14.66}{1.414} \approx 10.37 \text{ GHz} \tag{10}$$

4.4.2 Frequency Sweep Windows

Generate sweep windows for experimental validation:

$$f_{\text{min}}^{H_2O} = f_{\text{jam}} - \Delta f \tag{11}$$
$$f_{\text{max}}^{H_2O} = f_{\text{jam}} + \Delta f \tag{12}$$
$$f_{\text{min}}^{D_2O} = f_{D_2O} - \Delta f/\sqrt{2} \tag{13}$$
$$f_{\text{max}}^{D_2O} = f_{D_2O} + \Delta f/\sqrt{2} \tag{14}$$

For default parameters ($\Delta f = 2$ GHz):

Solvent	f_{min} **(GHz)**	f_{max} **(GHz)**
H_2O	12.66	16.66
D_2O	8.95	11.78

Table 4: Computed sweep windows

4.4.3 Operating Setpoints

Generate setpoints for irradiation apparatus:

Algorithm 3: Generate Operating Setpoints

```
FUNCTION GenerateSetpoints(f_jam, f_D2O, mode):
    setpoints = {}
    IF mode = "H2O" THEN
        setpoints.frequency = f_jam
        setpoints.sweep_start = f_jam - 2
        setpoints.sweep_stop = f_jam + 2
    ELSE IF mode = "D2O" THEN
        setpoints.frequency = f_D2O
        setpoints.sweep_start = f_D2O - 1.5
        setpoints.sweep_stop = f_D2O + 1.5
    END IF
    setpoints.power = 1.0          // Default power in Watts
    setpoints.temp_target = 25.0   // Default temperature
    setpoints.temp_tolerance = 0.1
    RETURN setpoints
END FUNCTION
```

4.4.4 Acceptance Criteria

Generate criteria for validating experimental results:

Algorithm 4: Generate Acceptance Criteria

```
FUNCTION GenerateAcceptanceCriteria(f_jam, f_D2O, epsilon):
    criteria = {}
    criteria.f_h2o_min = f_jam * (1 - epsilon)
    criteria.f_h2o_max = f_jam * (1 + epsilon)
    criteria.f_d2o_min = f_D2O * (1 - epsilon)
    criteria.f_d2o_max = f_D2O * (1 + epsilon)
    criteria.ratio_min = sqrt(2) * (1 - epsilon)
    criteria.ratio_max = sqrt(2) * (1 + epsilon)
    criteria.modulation_threshold = 0.10  // 10% minimum
```

```
    RETURN criteria
END FUNCTION
```

4.5 Rung Selection

4.5.1 Default Rung: Molecular Gate (n=19)

For protein folding applications, the default rung is $n = 19$, corresponding to the molecular gate timescale. This rung is selected because:

(1) It falls uniquely in the 50–100 ps window relevant to backbone dihedral transitions.

(2) It corresponds to a frequency (14.65 GHz) accessible with standard microwave equipment.

(3) It has been validated by machine-verified proofs.

4.5.2 Alternative Rungs

The method may compute frequencies for alternative rungs:

Rung n	τ_n **(s)**	f_n **(Hz)**	**Application**
15	3.1×10^{-12}	320 GHz	Fast vibrations
17	8.1×10^{-12}	123 GHz	Side chain dynamics
19	6.8×10^{-11}	14.7 GHz	Molecular gate (default)
21	1.8×10^{-10}	5.6 GHz	Loop motions
23	4.7×10^{-10}	2.1 GHz	Domain motions

Table 5: Frequencies for alternative rungs of the φ-ladder

4.6 Output Formats

4.6.1 JSON Output

Listing 1: Example JSON output

```
{
  "computed_at": "2026-01-17T12:00:00Z",
```

```
  "parameters": {
    "rung": 19,
    "tau0_s": 7.33e-15,
    "phi": 1.6180339887
  },
  "frequencies": {
    "f_jam_hz": 14.66e9,
    "f_jam_ghz": 14.66,
    "f_d2o_hz": 10.37e9,
    "f_d2o_ghz": 10.37,
    "ratio": 1.414
  },
  "sweep_windows": {
    "h2o": {"min_ghz": 12.66, "max_ghz": 16.66},
    "d2o": {"min_ghz": 8.95, "max_ghz": 11.78}
  },
  "acceptance_criteria": {
    "f_h2o_range_ghz": [13.93, 15.39],
    "f_d2o_range_ghz": [9.85, 10.89],
    "ratio_range": [1.343, 1.485],
    "modulation_threshold": 0.10
  }
}
```

4.6.2 Instrument Control File

Listing 2: Example instrument control file

```
# Jamming Frequency Control File
# Generated by Recognition Science Frequency Calculator
# Date: 2026-01-17

[FREQUENCY]
mode = fixed
frequency_ghz = 14.66
tolerance_ghz = 0.01

[SWEEP]
```

```
enabled = true
start_ghz = 12.66
stop_ghz = 16.66
step_ghz = 0.1

[POWER]
power_w = 1.0
mode = continuous

[TEMPERATURE]
target_c = 25.0
tolerance_c = 0.1

[ISOTOPE]
mode = H2O
d2o_frequency_ghz = 10.37
```

4.6.3 CSV Output

Listing 3: Example CSV output

```
parameter,value,unit
f_jam,14.66,GHz
f_d2o,10.37,GHz
tau_19,68.2,ps
phi,1.6180339887,dimensionless
sweep_h2o_min,12.66,GHz
sweep_h2o_max,16.66,GHz
sweep_d2o_min,8.95,GHz
sweep_d2o_max,11.78,GHz
```

4.7 Validation Integration

4.7.1 Pre-Experiment Validation

Before conducting experiments, the method can validate apparatus settings:

Algorithm 5: Validate Apparatus Settings

```
FUNCTION ValidateSettings(settings, computed):
    errors = []
    IF |settings.frequency - computed.f_jam| > 0.5 THEN
        errors.append("Frequency out of range")
    END IF
    IF settings.temp_tolerance > 0.5 THEN
        errors.append("Temperature tolerance too large")
    END IF
    IF settings.power < 0.1 OR settings.power > 10 THEN
        errors.append("Power out of recommended range")
    END IF
    RETURN errors
END FUNCTION
```

4.7.2 Post-Experiment Validation

After experiments, validate results against acceptance criteria:

Algorithm 6: Validate Experimental Results

```
FUNCTION ValidateResults(results, criteria):
    status = "PASS"
    IF results.f_optimal < criteria.f_min OR
       results.f_optimal > criteria.f_max THEN
        status = "FAIL: Frequency out of range"
    END IF
    IF results.modulation < criteria.modulation_threshold THEN
        status = "FAIL: Insufficient modulation"
    END IF
    IF |results.ratio - sqrt(2)| > criteria.ratio_tolerance THEN
        status = "FAIL: Isotope ratio incorrect"
    END IF
    RETURN status
END FUNCTION
```

4.8 Implementation Examples

4.8.1 Python Implementation

Listing 4: Python implementation of core algorithm

```
import math
import json

def compute_jamming_frequency(n=19, tau0=7.33e-15):
    """
    Compute jamming frequency from first principles.

    Args:
        n: Rung number (default 19 for molecular gate)
        tau0: Fundamental timescale in seconds

    Returns:
        Dictionary with computed frequencies and parameters
    """
    # Golden ratio
    phi = (1 + math.sqrt(5)) / 2

    # Rung timescale
    tau_n = tau0 * (phi ** n)

    # Jamming frequency
    f_jam = 1 / tau_n

    # Isotope-shifted frequency
    f_d2o = f_jam / math.sqrt(2)

    return {
        'phi': phi,
        'tau_n_s': tau_n,
        'f_jam_hz': f_jam,
        'f_jam_ghz': f_jam / 1e9,
        'f_d2o_hz': f_d2o,
        'f_d2o_ghz': f_d2o / 1e9,
```

```
        'ratio': math.sqrt(2)
    }

# Example usage
result = compute_jamming_frequency()
print(json.dumps(result, indent=2))
```

4.8.2 C Implementation

Listing 5: C implementation for embedded systems

```c
#include <math.h>

typedef struct {
    double phi;
    double tau_n;
    double f_jam_hz;
    double f_jam_ghz;
    double f_d2o_hz;
    double f_d2o_ghz;
} FrequencyResult;

FrequencyResult compute_jamming_frequency(int n, double tau0) {
    FrequencyResult result;

    // Golden ratio
    result.phi = (1.0 + sqrt(5.0)) / 2.0;

    // Rung timescale
    result.tau_n = tau0 * pow(result.phi, (double)n);

    // Jamming frequency
    result.f_jam_hz = 1.0 / result.tau_n;
    result.f_jam_ghz = result.f_jam_hz / 1.0e9;

    // Isotope-shifted frequency
    result.f_d2o_hz = result.f_jam_hz / sqrt(2.0);
    result.f_d2o_ghz = result.f_d2o_hz / 1.0e9;
```

```

    return result;
}
```

4.9 API Specification

4.9.1 REST API Endpoint

Listing 6: REST API specification

```
POST /api/v1/compute-frequency

Request Body:
{
  "rung": 19,                       // Optional, default 19
  "tau0": 7.33e-15,                 // Optional, default value
  "include_isotope": true,          // Optional, default true
  "sweep_margin_ghz": 2.0,          // Optional, default 2.0
  "tolerance": 0.05,                // Optional, default 0.05
  "output_format": "json"           // Optional: json, csv, instrument
}

Response:
{
  "status": "success",
  "data": {
    "frequencies": {...},
    "sweep_windows": {...},
    "acceptance_criteria": {...},
    "setpoints": {...}
  }
}
```

4.10 Error Handling

The method includes error handling for:

(1) **Invalid rung number:** Rung must be positive integer.

(2) **Invalid τ_0:** Must be positive and physically reasonable.

(3) **Numerical overflow:** Large rung numbers may exceed floating-point range.

(4) **Output formatting errors:** Invalid format specifications.

5 CLAIMS

What is claimed is:

5.1 Core Computation Claims

1. A computer-implemented method for calculating a frequency for modulating protein folding, comprising:

(a) receiving, by a processor, input parameters including a rung number n and a fundamental timescale τ_0;

(b) computing, by the processor, the golden ratio $\varphi = (1 + \sqrt{5})/2$;

(c) computing, by the processor, a molecular gate timescale $\tau_n = \tau_0 \times \varphi^n$;

(d) computing, by the processor, a jamming frequency $f_{\text{jam}} = 1/\tau_n$; and

(e) outputting, by the processor, the jamming frequency in a machine-readable format.

2. The method of claim 1, wherein the rung number n is 19 and the fundamental timescale τ_0 is approximately 7.33×10^{-15} seconds.

3. The method of claim 1, wherein the jamming frequency is approximately 14.65 GHz.

4. The method of claim 1, further comprising computing an isotope-shifted frequency $f_{\text{D}_2\text{O}} = f_{\text{jam}}/\sqrt{2}$.

5. The method of claim 4, wherein the isotope-shifted frequency is approximately 10.37 GHz.

6. The method of claim 1, wherein the machine-readable format is selected from JSON, CSV, XML, and instrument control file format.

5.2 Setpoint Generation Claims

7. A computer-implemented method for generating operating setpoints for a protein folding modulation apparatus, comprising:

(a) computing a jamming frequency according to claim 1;

(b) generating a frequency setpoint equal to the computed jamming frequency;

(c) generating a frequency sweep window comprising a minimum frequency and a maximum frequency centered on the jamming frequency;

(d) generating a power setpoint;

(e) generating a temperature setpoint and temperature tolerance; and

(f) outputting the setpoints in a format compatible with the apparatus.

8. The method of claim 7, wherein the frequency sweep window spans from $(f_{\text{jam}} - \Delta f)$ to $(f_{\text{jam}} + \Delta f)$, where Δf is a configurable sweep margin.

9. The method of claim 7, further comprising generating setpoints for D_2O operation by scaling the frequency setpoint and sweep window by factor $1/\sqrt{2}$.

10. The method of claim 7, wherein the output format is an instrument control file comprising frequency, power, and temperature parameters.

5.3 Acceptance Criteria Claims

11. A computer-implemented method for generating acceptance criteria for validating protein folding modulation experiments, comprising:

(a) computing a jamming frequency and isotope-shifted frequency according to claims 1 and 4;

(b) computing a frequency acceptance range for H_2O experiments as $(f_{\text{jam}} \times (1 - \epsilon), f_{\text{jam}} \times (1 + \epsilon))$;

(c) computing a frequency acceptance range for D_2O experiments as $(f_{D_2O} \times (1 - \epsilon), f_{D_2O} \times (1 + \epsilon))$;

(d) computing a frequency ratio acceptance range as $(\sqrt{2} \times (1 - \epsilon), \sqrt{2} \times (1 + \epsilon))$;

(e) setting a modulation threshold; and

(f) outputting the acceptance criteria.

12. The method of claim 11, wherein ϵ is 5%.

13. The method of claim 11, wherein the modulation threshold is 10%.

14. A computer-implemented method for validating experimental results against computed acceptance criteria, comprising:

(a) receiving experimental results including observed optimal frequency and observed modulation;

(b) receiving acceptance criteria generated according to claim 11;

(c) comparing the observed optimal frequency to the frequency acceptance range;

(d) comparing the observed modulation to the modulation threshold;

(e) outputting a validation status of PASS if all comparisons are within criteria, or FAIL with a reason if any comparison is outside criteria.

5.4 System Claims

15. A system for computing protein folding jamming frequencies, comprising:

(a) a processor;

(b) a memory storing instructions that, when executed by the processor, cause the processor to perform the method of claim 1;

(c) an input interface for receiving computation parameters; and

(d) an output interface for delivering computed frequencies and related parameters.

16. The system of claim 15, wherein the system is implemented as a cloud computing service accessible via API.

17. The system of claim 15, wherein the system is implemented as embedded firmware in a protein folding modulation apparatus.

18. The system of claim 15, wherein the system is implemented as a mobile application.

5.5 Integration Claims

19. A method for controlling a protein folding modulation apparatus using computed frequencies, comprising:

(a) computing a jamming frequency according to claim 1;

(b) transmitting the computed frequency to a frequency controller of the apparatus;

(c) commanding the apparatus to irradiate a sample at the computed frequency; and

(d) receiving measurement data from the apparatus.

20. The method of claim 19, further comprising:

(a) generating acceptance criteria according to claim 11;

(b) validating the measurement data against the acceptance criteria; and

(c) outputting a validation report.

21. A non-transitory computer-readable medium storing instructions that, when executed by a processor, cause the processor to perform the method of claim 1.

22. The non-transitory computer-readable medium of claim 21, further storing instructions to perform the methods of claims 7 and 11.

ABSTRACT

A computer-implemented method for calculating the frequency for modulating protein folding rates from first principles. The method comprises: computing the golden ratio $\varphi = (1+\sqrt{5})/2$; computing a molecular gate timescale $\tau_n = \tau_0 \times \varphi^n$ for a selected rung number n (default $n = 19$); computing a jamming frequency $f_{\mathrm{jam}} = 1/\tau_n$ (approximately 14.65 GHz); optionally computing an isotope-shifted frequency $f_{\mathrm{D_2O}} = f_{\mathrm{jam}}/\sqrt{2}$ (approximately 10.37 GHz); and outputting the computed values. The method further generates operating setpoints for irradiation apparatus, frequency sweep windows for experimental validation, and acceptance criteria for result validation including frequency ranges, frequency ratios, and modulation thresholds. The method may be implemented on general-purpose computers, embedded controllers, cloud platforms, or mobile devices, with outputs in JSON, CSV, instrument control file, or other formats. The method implements the Recognition Science framework for deriving physical constants from the golden ratio, with underlying derivations verified by machine-checked proofs. Applications include controlling protein folding modulation apparatus, validating experimental results, and providing computational support for research and industrial applications.

— END OF SPECIFICATION —

INVENTOR DECLARATION

I, Jonathan Washburn, declare that:

(1) I am the original and sole inventor of the computer-implemented method described and claimed in this application.

(2) I have reviewed the above specification and claims and believe them to be accurate and complete.

(3) I believe the claimed invention to be novel, useful, and non-obvious over the prior art.

(4) I authorize the filing of this provisional patent application to establish a priority date.

Inventor Signature: ____

Name: Jonathan Washburn

Email: washburn.jonathan@gmail.com

Date: ____

This document is intended for provisional patent application filing purposes.
All information contained herein is confidential and proprietary.