Quantum networks dynamics and population analysis using QuNet

Computational Methods for Multi-Qubit Systems

Unnati Akhouri

3 June, 2025

Agenda

- Computational methods for analyzing quantum population dynamics
- Multi-qubit systems and their temporal evolution
- Data processing and statistical analysis techniques
- Visualization methods for quantum simulation results

Goal for today/tomorrow

By the end of this session, you will be able to:

- simulate quantum networks
- Extract and process quantum population data from simulation results
- Perform temporal averaging and ensemble statistics on quantum dynamics
- Analyze late-time behavior in quantum systems
- Visualize quantum state evolution using appropriate plotting techniques
- Understand the role of connectivity and update rules in quantum dynamics

What are Quantum Populations?

- **Population**: Probability of finding the system in a particular quantum state
- For multi-qubit systems:
  - Each qubit exists in superposition of $|0\rangle$ and $|1\rangle$ states
  - Population dynamics track probability evolution over time

Key Concepts

- **Unitary Evolution**: Quantum systems evolve via unitary operators that preserve probability
- **Ensemble Averaging**: Statistical analysis over multiple quantum experiments
- **Connectivity**: How qubits are connected in quantum circuits or lattices
- **Update Rules**: Cost function that the code computes to decide the sequence in which quantum operations are applied

Data Structure and Extraction

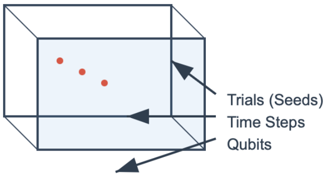
```
def get_pops(data, n_qubits, connectivity, update_rule):
    # Extracts population data from nested dictionary structure
    # Returns 3D array: [trial, time_step, qubit]
```

Key Features:

- **Hierarchical Data Access**: Navigate nested simulation results
- **Seed Management**: Handle multiple random seeds for statistics
- **Data Reshaping**: Convert dictionaries to NumPy arrays

Data Dimensions

- **Dimension 0**: Individual trials (different random seeds)
- **Dimension 1**: Time steps in the evolution
- **Dimension 2**: Individual qubits in the system



Questions you may wonder about:

1. Why run multiple trials with different seeds?
2. How does connectivity choice affect quantum dynamics?
3. What role do update rules play in quantum circuit execution?

Temporal Analysis Techniques

Three Main Approaches

1. **Late-Time Averaging**: Focus on equilibrium behavior
2. **Time-Cumulative Averaging**: Study approach to equilibrium
3. **Moving Average Analysis**: Smooth noisy data while preserving trends

Late-Time Averaging

```
def late_time_average_of_list(
    list_one_trial):
     # Removes first 201 and last 250 time
        steps
     # Computes mean over remaining "late-
        time" window
```

Analysis Window Selection:

- **Early-time removal**: Eliminate transient behavior and initialization effects
- **Late-time removal**: Avoid finite-size effects or numerical artifacts
- **Analysis window**: Focus on quasi-equilibrium behavior

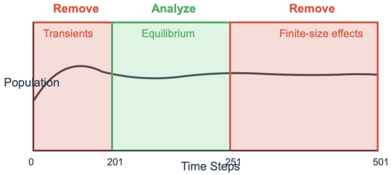
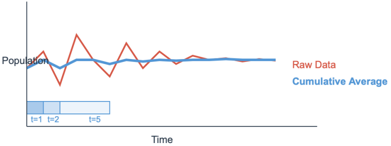


Figure: Caption

Time-Cumulative Averaging

```
def time_averaged_one_point_measures_at_t(
    dataset, t):
     # Computes cumulative average up to
        time t
     # Formula: (1/(t+1))      (data[0:t])
```



Applications:

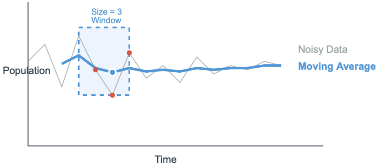
- Studying approach to equilibrium
- Identifying convergence behavior
- Analyzing long-time stability

Moving Average Analysis

```
def moving_average(data, window_size):
    # Applies sliding window average to
        smooth data
    # Uses convolution for efficient
        computation
```

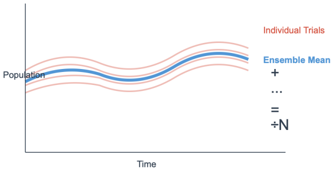
Benefits:

- Noise reduction in quantum measurements
- Trend identification in noisy data
- Maintaining temporal resolution while smoothing



Ensemble Averaging

```
def
    ensemble_averaged_one_point_measures_mean_std
    (datasets):
     # Computes mean and standard deviation
         across trials
     # Returns ensemble statistics for each
         time step and qubit
```



Statistical Measures:

- **Mean**: Central tendency of quantum populations
- **Standard Deviation**: Measure of quantum fluctuations and measurement uncertainty

Cross-Qubit Analysis

```
def late_time_averages_one_point_all_seeds_between_q(datasets):
    # Analyzes variation between different qubits
    # Returns mean population and inter-qubit standard deviation
```

Physical Interpretation:

- **Uniform populations**: Indicates thermalization or equilibration
- **Large inter-qubit variations**: Suggests localization or non-ergodic behavior
- **Standard deviation trends**: Measure of spatial correlations

Visualization Techniques

Multi-Panel Visualization Strategy

- **Line Plots**: Show detailed temporal evolution for each qubit
- **Heatmaps**: Reveal spatial patterns and correlations
- **Complementary Views**: Different aspects of the same data

Implementation Example

```
def plot_node_evolution(data):
    # Creates two complementary visualizations:
    # 1. Line plot: Individual qubit evolution
    # 2. Heatmap: Space-time visualization
```

Design Principles:

- **Line Plots**: Best for comparing individual qubit dynamics, identifying oscillations
- **Heatmaps**: Best for spatial correlation patterns, overall system behavior

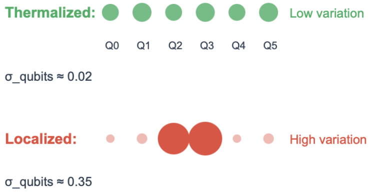
Physical Interpretation and Applications

Three Key Physical Phenomena

1. **Quantum Thermalization**
2. **Many-Body Localization**
3. **Quantum Algorithm Performance**

Quantum Thermalization

- **Early-time behavior**: Non-equilibrium dynamics, transient oscillations
- **Late-time behavior**: Approach to thermal equilibrium
- **Fluctuations**: Quantum vs. classical statistical behavior



Key Question: How do isolated quantum systems reach thermal equilibrium?

Many-Body Localization

- **Extended states**: Populations spread across qubits
- **Localized states**: Populations concentrated on few qubits
- **Mobility edge**: Transition between extended and localized behavior

Key Question: Under what conditions do quantum systems fail to thermalize?

Computational Best Practices

Code Organization:

- **Modular functions**: Each function has specific, well-defined purpose
- **Clear documentation**: Comments explain physical meaning
- **Consistent data structures**: Standardized array dimensions and indexing

Performance Considerations:

- **NumPy vectorization**: Efficient array operations
- **Memory management**: Appropriate data types and array sizes
- **Statistical sampling**: Balance between accuracy and computational cost

Error Handling and Validation

```
if window_size <= 0:
    raise ValueError("Window size must be greater than 0")
if data.shape[0] < window_size:
    raise ValueError("Number of time steps must be >= window size")
```

Why Error Handling Matters:

- Prevents silent failures in analysis pipelines
- Makes debugging easier for complex quantum simulations
- Ensures reproducible scientific results

Practical Exercises

Exercise 1: Data Extraction

- Extract population data for different connectivity types
- Compare data structure for different numbers of qubits
- Analyze how different update rules affect results

Exercise 2: Statistical Analysis

- Implement ensemble averaging for extracted data
- Calculate standard deviation and confidence intervals for population measurements
- Identify qubits with anomalous behavior

More Exercises

Exercise 3: Visualization

- Create line plots showing population evolution for selected qubits
- Generate heatmaps for different connectivity patterns
- Compare early-time vs. late-time behavior visually

Exercise 4: Physical Analysis

- Identify signatures of thermalization in your data
- Calculate effective temperatures from population distributions
- Analyze the role of system size on equilibration time

Summary and Key Takeaways

Core Concepts Covered:

1. **Data Management**: Efficient extraction and organization of quantum simulation data
2. **Temporal Analysis**: Methods for analyzing time-dependent quantum behavior
3. **Statistical Methods**: Ensemble averaging and uncertainty quantification
4. **Visualization**: Effective representation of multi-dimensional quantum data
5. **Physical Interpretation**: Connection between computational results and quantum physics

Next Steps

- Apply these methods to your own quantum simulation data
- Explore advanced statistical techniques for quantum data analysis
- Investigate connections between population dynamics and quantum information theory

Questions?