Summary of Python Modules in Forward-Equivalent Simulation

1 Overview

The Python files define the implementation of the Forward-Equivalent (FE) simulation framework, including:

- **Precomputing FE-adjusted birth, death, and mutation rates** to avoid simulating unobserved lineages.
- **Modeling evolutionary state transitions** using Poisson processes.
- **Applying mutations based on FE parameters** in a phylogenetic tree.
- **Providing utility functions** for interpolation, numerical computations, and tree statistics.

2 File Summaries

2.1 `modulators.py` - Core Computation of FE Parameters

This file defines the `FEModulator` class, which precomputes and stores all forward-equivalent model parameters for efficient simulation.

Key Features:

- Computes **survival probabilities** $s_a(\tau)$.
- Computes **integral survival values** $S_a(\tau)$ used for birth, death, and mutation rate adjustments.
- Computes **FE-adjusted mutation rates** $m_a(\tau)$.

Important Methods:

- `_s(t, phenotype)`: Returns survival probability at time t.
- `_S(t, phenotype)`: Computes integral of survival probability.
- `_m(t, phenotype)`: Returns FE-adjusted mutation rate.

Why is this file important?

- `FEModulator` **precomputes all essential FE parameters**, making simulations significantly more efficient.
- Without it, the simulation would need to recompute rates dynamically, making it much slower.

2.2 `my_bdms.py` - Evolutionary Processes and Mutations

This file defines how birth, death, and mutation events occur within the FE simulation framework. It includes:

Key Classes:

- `DiscreteProcess`: A Poisson process for constant-rate (birth, death, or mutation) transitions.
- `CustomProcess`: A more flexible Poisson process allowing time- and state-dependent transition rates.
- `CustomMutator`: Handles FE-based mutations and applies them to a phylogenetic tree.

Key Functionalities:

- `DiscreteProcess` simulates **fixed-rate evolutionary events**.
- `CustomProcess` allows **time-dependent** birth, death, and mutation rate modifications.
- `CustomMutator` uses `FEModulator` to apply **FE-adjusted mutations** to tree nodes.

Why is this file important?

- It allows the FE model to dynamically adjust transition rates instead of assuming static values.
- `CustomMutator` ensures that mutations follow FE-adjusted probabilities, maintaining statistical equivalence to FullSim.

2.3 `utils.py` - Helper Functions for Computation and Analysis

This file provides utility functions for:

- **Interpolating precomputed values** (for fast retrieval of FE parameters).
- **Computing tree statistics** (e.g., comparing FullSim vs. FE).
- **Ensuring numerical stability** in simulations.

Key Functions:

- `grid_interp(t, array, t_min, dt)`: Interpolates values from a precomputed time grid.
- `compute_tree_stats(tree)`: Computes tree statistics for validation and comparison.
- `clamp(value, min_val, max_val)`: Ensures values remain within valid ranges.

Why is this file important?

- Supports efficient lookup of FE-adjusted parameters.
- Provides statistical analysis tools to validate FE simulation accuracy.

3 How These Files Work Together

1. `modulators.py` **precomputes FE-adjusted rates**.
2. `my_bdms.py` **uses these rates** to simulate birth, death, and mutation processes.
3. `utils.py` **provides numerical tools** for computation and analysis.

4 Comparison to FullSim

- FullSim computes and prunes all lineages, while FE **skips unobserved lineages**.
- This reduces computational complexity from $O(M)$ (FullSim) to $O(N)$ (FE), where $N \ll M$.
- **FE is 2-6× faster** than FullSim while preserving statistical accuracy.

5 Potential Questions from the Professor

1. What is the most important file? *Answer:* `modulators.py` because it computes all FE-adjusted rates, making the simulation efficient.

2. How do these files interact? *Answer:* `modulators.py` computes rates, `my_bdms.py` applies them, and `utils.py` supports numerical operations.

3. Why is FE faster than FullSim? *Answer:* FE avoids computing unobserved lineages, making it 2-6× faster than FullSim.