Exposure Module Documentation

1 Overview

This module provides functionality for calculating exposure measures for assets based on various hazard types. It includes classes for categorizing exposure levels, defining exposure measures, and calculating exposures for a list of assets.

2 Enums and Data Classes

2.1 `Category` (Enum)

Represents different levels of exposure:

- `LOWEST`, `LOW`, `MEDIUM`, `HIGH`, `HIGHEST`, `NODATA`

2.2 `Bounds` (dataclass)

Defines the bounds for a category:

- `category` (str): The category name
- `lower` (float): Lower bound (inclusive)
- `upper` (float): Upper bound (exclusive)

2.3 `AssetExposureResult` (dataclass)

Stores the exposure results for an asset:

- `hazard_categories` (Dict[type, Tuple[Category, float, str]]): Maps hazard types to their category, value, and data path

3 Abstract Base Classes

3.1 `ExposureMeasure` (ABC)

Base class for exposure measures:

- Abstract method: `get_exposures(asset: Asset, data_responses: Iterable[HazardDataRespons -> Dict[type, Tuple[Category, float, str]]`

4 Classes

4.1 `JupterExposureMeasure` (ExposureMeasure)

Implements exposure measures based on Jupiter data:
Methods:

- `__init__()`: Initializes the exposure bins
- `get_data_requests(asset: Asset, *, scenario: str, year: int) -> Iterable[HazardDataRequest]`: Returns data requests for the asset
- `get_exposures(asset: Asset, data_responses: Iterable[HazardDataResponse]) -> Dict[type, Tuple[Category, float, str]]`: Calculates exposures for the asset
- `get_exposure_bins() -> Dict`: Defines exposure bins for various hazard types
- `bounds_to_lookup(bounds: Iterable[Bounds]) -> Tuple[np.ndarray, np.ndarray]`: Converts bounds to lookup arrays

5 Functions

5.1 `calculate_exposures(assets: List[Asset], hazard_model: HazardModel, exposure_measure: ExposureMeasure, scenario: str, year: int) -> Dict[Asset, AssetExposureResult]`

Calculates exposures for a list of assets:

- Parameters:
  - `assets`: List of assets to calculate exposures for
  - `hazard_model`: Model providing hazard data
  - `exposure_measure`: Measure to use for exposure calculation
  - `scenario`: Climate scenario
  - `year`: Year for which to calculate exposures
- Returns: Dictionary mapping assets to their exposure results

6 Key Concepts

1. **Exposure Categories**: Exposures are categorized into levels (LOWEST to HIGHEST) based on predefined thresholds for each hazard type.
2. **Hazard Types**: The module considers various hazard types including CombinedInundation, ChronicHeat, Wind, Drought, Hail, and Fire.

3. **Data Requests**: The `JupterExposureMeasure` class generates data requests for each asset and hazard type.

4. **Exposure Calculation**: Exposures are calculated by comparing hazard data to predefined thresholds and assigning appropriate categories.

7 Usage Notes

1. The `JupterExposureMeasure` class is designed to work with specific hazard data sources and indicators. Ensure that the hazard model can provide the required data.

2. When using `calculate_exposures`, make sure that the `hazard_model` is compatible with the data requests generated by the `exposure_measure`.

3. The exposure bins in `JupterExposureMeasure` are hardcoded. Consider making these configurable if flexibility is needed.

4. The module uses numpy for efficient array operations, particularly in the `bounds_to_lookup` method.

5. Error handling is minimal. Consider adding more robust error checking and handling, especially for edge cases in data responses.

6. The module is designed to be extensible. New exposure measures can be implemented by subclassing `ExposureMeasure`.