Vertical Discretization

Core Components and Initialization

The `VerticalDiscretization` abstract class, implemented in C#, provides a framework for defining vertical coordinate systems in estuarine circulation modeling. It supports the creation of computational grids for 2D or 3D hydrodynamic simulations, with derived classes implementing specific coordinate systems (sigma and z-level). The class is initialized with the following parameters:

- Number of grid points in x-direction: n_x (horizontal)
- Number of grid points in z-direction: n_z (vertical)
- Spatial step in x-direction: $\Delta\xi = \frac{L}{n_x-1}$, where L is the estuary length (m)
- Spatial step in z-direction: $\Delta\eta$ (dimensionless for sigma, meters for z-level)
- Estuary length: L (m)
- Estuary depth: h (m)
- Horizontal coordinates: $x(i,j)$ (m, 2D array)

The class maintains arrays for:

- Vertical coordinates: $z(i,j)$ (m)
- Metric term: $\frac{\partial z}{\partial \xi}$
- Metric term: $\frac{\partial z}{\partial \eta}$

These arrays are accessible via properties `Z`, `Z_xi`, and `Z_eta`.

Functioning Logic

The `VerticalDiscretization` abstract class defines two abstract methods implemented by derived classes:

1. `InitializeGrid`: Sets up the vertical coordinate grid based on the chosen system.
2. `UpdateDepth`: Updates the grid when the estuary depth changes.

Two derived classes, `SigmaCoordinates` and `ZLevelCoordinates`, implement these methods for sigma and z-level coordinate systems, respectively.

Sigma Coordinates

The `SigmaCoordinates` class implements a terrain-following sigma coordinate system, where the vertical coordinate scales with the local water depth. It is initialized with the same parameters as the base class.

Grid Initialization

The `InitializeGrid` method defines the vertical coordinate:

$$z(i,j) = \eta_j \cdot h, \quad \eta_j = j \cdot \Delta\eta, \quad j = 0, 1, \ldots, n_z - 1 \tag{1}$$

where $\eta_j \in [0,1]$ is the sigma coordinate (0 at the bed, 1 at the surface), and h is the estuary depth. This ensures that grid layers follow the bathymetry.

Metric Terms

The method `ComputeMetricTerms` calculates spatial derivatives:

$$\frac{\partial z}{\partial \xi}(i,j) = \begin{cases} \frac{z(i+1,j)-z(i-1,j)}{2\Delta\xi} & \text{if } 0 < i < n_x - 1 \\ \frac{z(1,j)-z(0,j)}{\Delta\xi} & \text{if } i = 0 \\ \frac{z(n_x-1,j)-z(n_x-2,j)}{\Delta\xi} & \text{if } i = n_x - 1 \end{cases} \tag{2}$$

$$\frac{\partial z}{\partial \eta}(i,j) = \begin{cases} \frac{z(i,j+1)-z(i,j-1)}{2\Delta\eta} & \text{if } 0 < j < n_z - 1 \\ \frac{z(i,1)-z(i,0)}{\Delta\eta} & \text{if } j = 0 \\ \frac{z(i,n_z-1)-z(i,n_z-2)}{\Delta\eta} & \text{if } j = n_z - 1 \end{cases} \tag{3}$$

For sigma coordinates, since $z = \eta \cdot h$, and assuming constant depth across x, $\frac{\partial z}{\partial \xi} \approx 0$, while $\frac{\partial z}{\partial \eta} \approx h$.

Depth Update

The `UpdateDepth` method updates the estuary depth h and reinitializes the grid using the same sigma coordinate formulation.

Z-Level Coordinates

The `ZLevelCoordinates` class implements a z-level coordinate system with fixed horizontal layers, independent of bathymetry. It is initialized with the same parameters as the base class.

Grid Initialization

The `InitializeGrid` method defines the vertical coordinate:

$$z(i,j) = j \cdot \Delta z, \quad \Delta z = \frac{h}{n_z - 1}, \quad j = 0, 1, \ldots, n_z - 1 \tag{4}$$

where Δz is the uniform vertical spacing, and z ranges from 0 (bed) to h (surface).

Metric Terms

The `ComputeMetricTerms` method calculates the same derivatives as in sigma coordinates:

$$\frac{\partial z}{\partial \xi}(i,j) = \begin{cases} \frac{z(i+1,j)-z(i-1,j)}{2\Delta\xi} & \text{if } 0 < i < n_x - 1 \\ \frac{z(1,j)-z(0,j)}{\Delta\xi} & \text{if } i = 0 \\ \frac{z(n_x-1,j)-z(n_x-2,j)}{\Delta\xi} & \text{if } i = n_x - 1 \end{cases} \tag{5}$$

$$\frac{\partial z}{\partial \eta}(i,j) = \begin{cases} \frac{z(i,j+1)-z(i,j-1)}{2\Delta\eta} & \text{if } 0 < j < n_z - 1 \\ \frac{z(i,1)-z(i,0)}{\Delta\eta} & \text{if } j = 0 \\ \frac{z(i,n_z-1)-z(i,n_z-2)}{\Delta\eta} & \text{if } j = n_z - 1 \end{cases} \tag{6}$$

For z-level coordinates, since z is constant across x, $\frac{\partial z}{\partial \xi} = 0$, and $\frac{\partial z}{\partial \eta} = \frac{\Delta z}{\Delta \eta}$.

Depth Update

The `UpdateDepth` method updates the estuary depth h and reinitializes the grid with updated Δz.

Physical and Mathematical Models

The `VerticalDiscretization` framework supports two coordinate systems:

- **Sigma Coordinates**:

$$z(i,j) = \eta_j \cdot h, \quad \eta_j = j \cdot \Delta\eta, \quad \Delta\eta = \frac{1}{n_z - 1} \tag{7}$$

$$\frac{\partial z}{\partial \xi} \approx 0 \quad (\text{assuming constant depth}) \tag{8}$$

$$\frac{\partial z}{\partial \eta} \approx h \tag{9}$$

- **Z-Level Coordinates**:

$$z(i,j) = j \cdot \Delta z, \quad \Delta z = \frac{h}{n_z - 1} \tag{10}$$

$$\frac{\partial z}{\partial \xi} = 0 \tag{11}$$

$$\frac{\partial z}{\partial \eta} = \frac{\Delta z}{\Delta \eta} \tag{12}$$

These models provide flexible vertical discretization for estuarine modeling, with sigma coordinates suitable for terrain-following simulations and z-level coordinates for fixed-layer applications, ensuring accurate representation of vertical gradients and bathymetric variations.