

Anemoi graphs

anemoi-graphs.readthedocs.io

Mario Santa Cruz

ECMWF

mario.santacruz@ecwmf.int

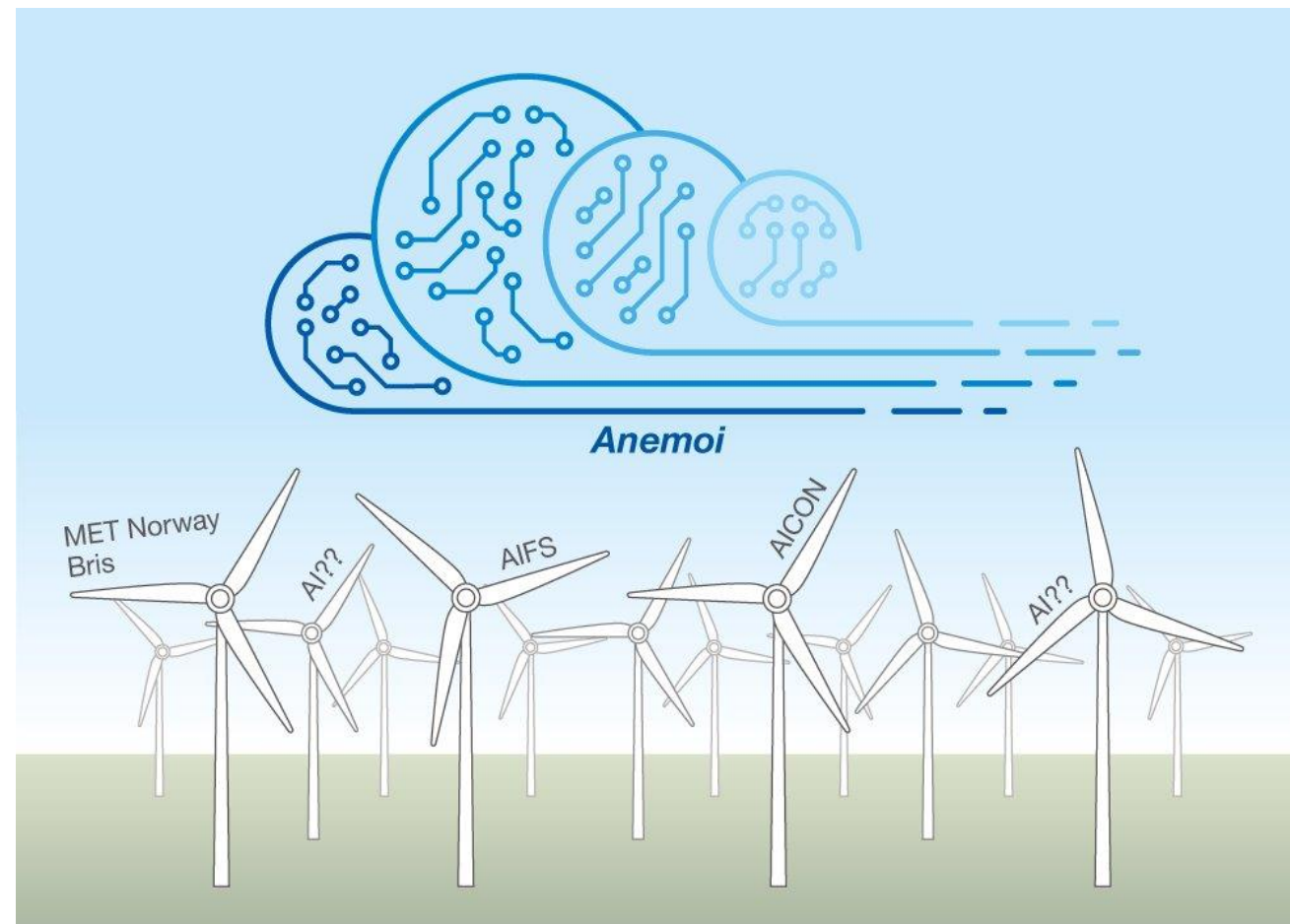
Anemoi framework

- Develop machine learning (ML) weather forecasting models.

- Dataset preparation tools.
- Graph creation tools.
- ML model training support.
- Inference tools integrated with verification software.
- Registry for datasets and trained models.

- **Benefits:**

- Simplifies shared meteorological challenges.
- Enables training of models using existing recipes and custom data.



Anemoi framework

- Develop machine learning (ML) weather forecasting models.
 - Dataset preparation tools.
 - Graph creation tools.
 - ML model training support.
 - Inference tools integrated with verification software.
 - Registry for datasets and trained models.
- **Benefits:**
 - Simplifies shared meteorological challenges.
 - Enables training of models using existing recipes and custom data.

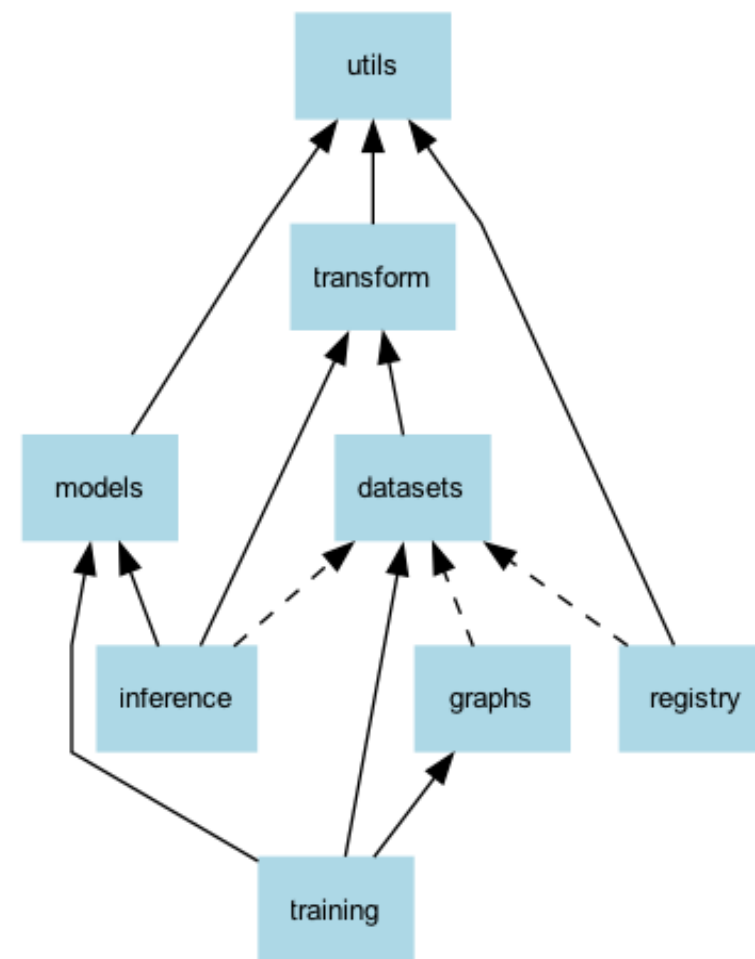
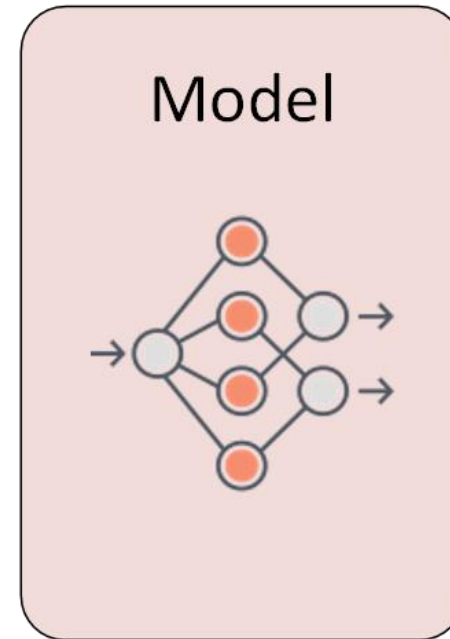
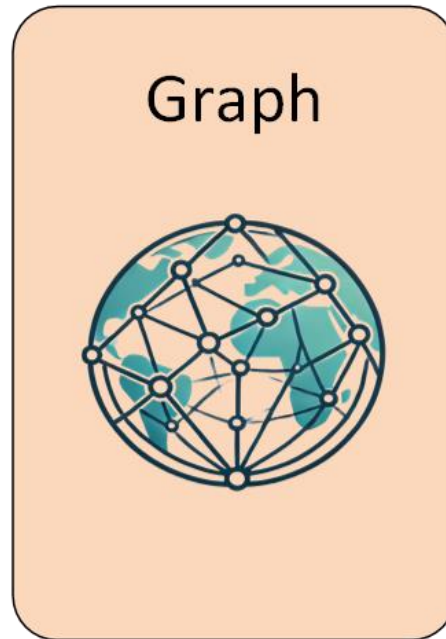
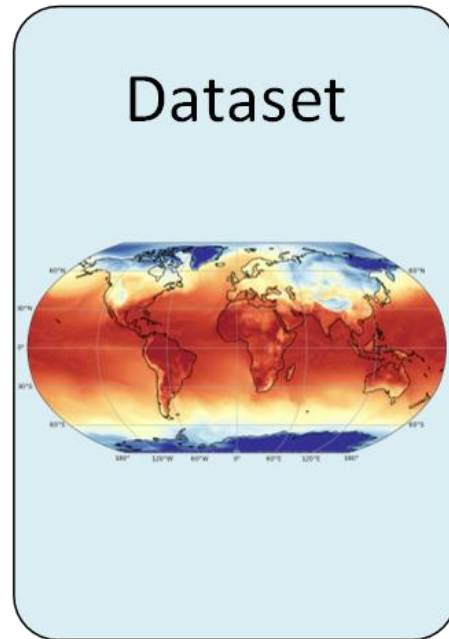


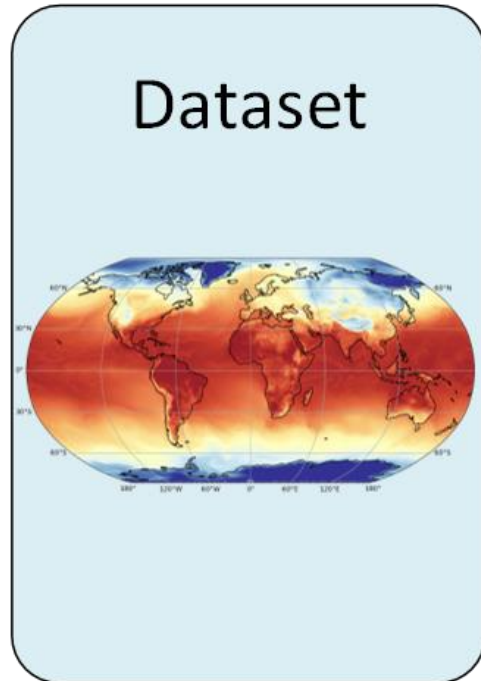
Diagram of dependencies within anemoi packages.

Getting started

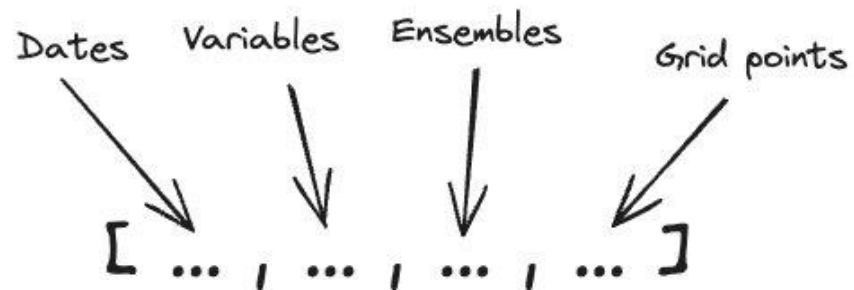
To train a data-driven weather forecasting model in **Anemoi**, three main components are needed:



anemoi-datasets

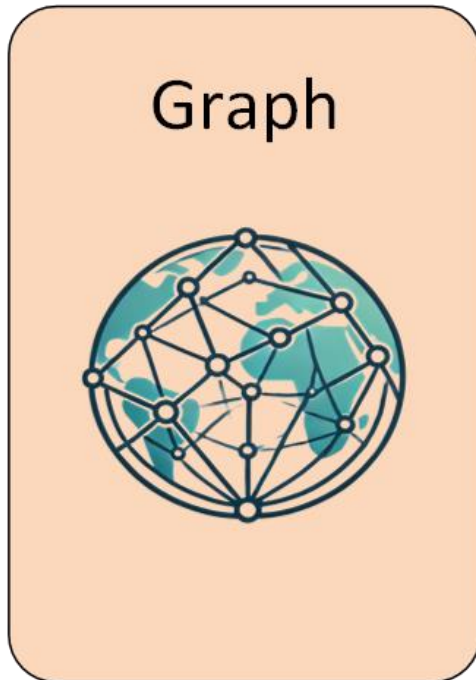


anemoi-datasets.readthedocs.io



- Create "machine-learning ready" datasets for training data-driven weather forecasts.
- Make the loading of data samples as efficient as possible
 - I/O operations are minimised
- Zarr
 - Offers an array-like view on chunks
 - Each file is a single date
- Using datasets
 - Subsetting datasets (time, variables, member, ...)
 - Combining datasets (join, concat, cutout, ensemble, ...)

anemoui-graphs



anemoui-graphs.readthedocs.io

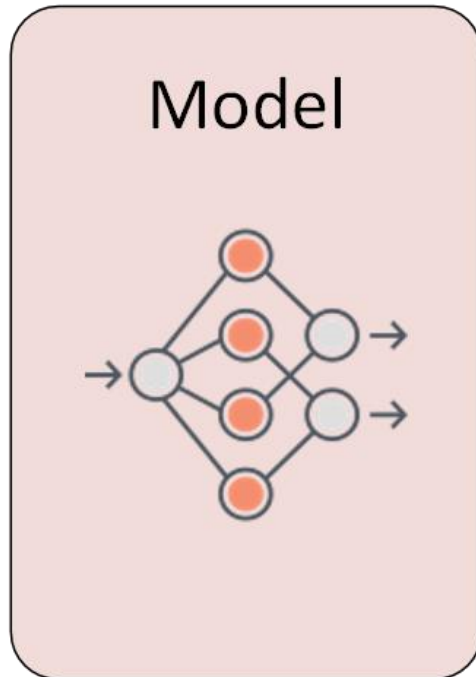
In Anemoui, it is represented by a `torch_geometric.data.HeteroData` object, and stored in a .PT file.

- A graph is a structure composed of:
 - **Nodes:** Represent locations in space.
 - **Edges:** Define how information flows between nodes.

```
HeteroData(  
  data={  
    x=[40320, 2], # coordinates in radians (lat in [-pi/2, pi/2], lon in [0, 2pi])  
    node_type='ZarrDatasetNodes',  
    area_weight=[40320, 1],  
  },  
  hidden={  
    x=[10242, 2], # coordinates in radians (lat in [-pi/2, pi/2], lon in [0, 2pi])  
    node_type='TriNodes',  
    area_weight=[10242, 1],  
  },  
  (data, to, hidden)={  
    edge_index=[2, 62980],  
    edge_type='CutOffEdges',  
    edge_length=[62980, 1],  
    edge_dirs=[62980, 2],  
  },  
  (hidden, to, hidden)={  
    edge_index=[2, 81900],  
    edge_type='MultiScaleEdges',  
    edge_length=[81900, 1],  
    edge_dirs=[81900, 2],  
  },  
  (hidden, to, data)={  
    edge_index=[2, 120960],  
    edge_type='KNNEdges',  
    edge_length=[120960, 1],  
    edge_dirs=[120960, 2],  
  }  
})
```

Console log of a graph created with anemoui-graphs.

anemoi-models



anemoi-models.readthedocs.io

- Graph Neural Networks (GNNs) are a type of neural network designed to operate on **graph-structured** data.
 - $GNN(G) = G'$, where G, G' are graphs
- Node features are updated based on message passing:
 - Message creation (from node and edge features)
 - Message aggregation
 - Node update
- Permutation-invariance

anemoi-graphs

Tools for creating graphs used in data-driven, deep learning weather forecasting models.

- **High-Level Interface:**
 - YAML recipe file to define graph configuration.
- **Node Definition:**
 - Based on dataset coordinates (e.g., Zarr, NPZ, TXT, ...) or algorithmic methods (e.g., triangular refined icosahedron).
- **Edge Definition:**
 - Methods include cut-off radius or K nearest-neighbors.
- **Attributes:**
 - Supports node and edge attributes (weights, lengths, directions).

anemoi-graphs.readthedocs.io

The screenshot shows the documentation page for the 'Command line tool' of anemoi-graphs. The page title is 'anemoi-graphs.readthedocs.io' and the page path is '/ Command line tool'. The main heading is 'Command line tool'. The text explains that when the 'anemoi-graphs' package is installed, a command line tool is also installed, which can be used to build graphs based on YAML recipe files and inspect existing graphs. It provides help options: '% anemoi-graphs --help'. To create a graph, the 'create' command is used: '\$ anemoi-graphs create recipe.yaml graph.pt'. The documentation states that the '.yaml' recipe file consists of high-level specifications for generating the graphs at each layer, and an example of a simple recipe file is given in the 'following section'. The 'create' command reads specifications in the 'recipe.yaml' file and writes to a PyTorch '.pt' file. To describe an existing graph stored as a '.pt' file, the 'describe' command is used: '\$ anemoi-graphs describe graph.pt'. This will generate a text summary of the graph, including the number of nodes and edges at each layer, the geographic boundaries, and statistics about the edge lengths. The summary shows the path 'graph.pt', format version '0.0.1', and size '3.1 MiB (3,283,650)'. It includes two tables: one for nodes and one for edges.

Nodes name	Num. nodes	Attribute dim	Min. latitude	Max. latitude	Min. longitude	
data	10,840	4	-3.135	3.140	0.02	
hidden	6,200	4	-3.141	3.137	0.01	

Source	Destination	Num. edges	Attribute dim	Min. length	Max. length	Mean
data	hidden	13508	1	0.3116	25.79	11.
hidden	data	40910	1	0.2397	21.851	12.

Documentation of anemoi-graphs package.

Command line tool

- Create a new graph:

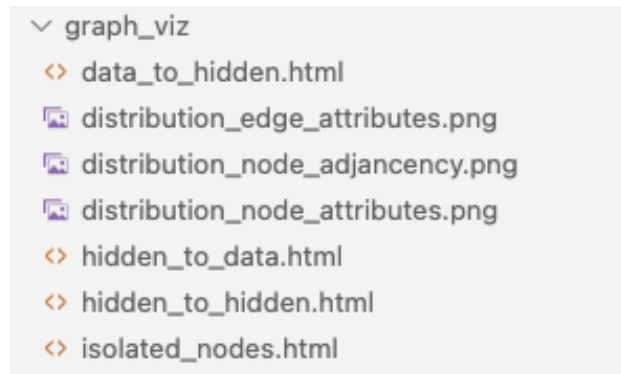
```
>>> anemoi-graphs create recipe.yaml graph.pt
```

- Describe an existing graph:

```
>>> anemoi-graphs describe graph.pt
```

- Inspect visually an existing graph:

```
>>> anemoi-graphs inspect graph.pt graph_viz/
```



Local files generated to inspect graphs.

```
Path      : graph.pt
Format version: 0.0.1

Size      : 3.1 MiB (3,283,650)
```

Nodes name	Num. nodes	Attribute dim	Min. latitude	Max. latitude	Min. longitude
data	10,840	4	-3.135	3.140	0.02
hidden	6,200	4	-3.141	3.137	0.01

Source	Destination	Num. edges	Attribute dim	Min. length	Max. length	Mean
data	hidden	13508	1	0.3116	25.79	11
hidden	data	40910	1	0.2397	21.851	12

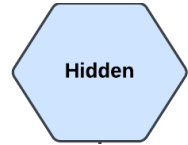
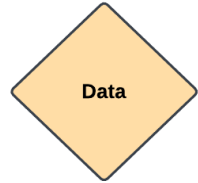
```
Graph ready, last update 17 seconds ago.
Statistics ready.
```

Console log when describing/inspecting a graph with anemoi-graphs.

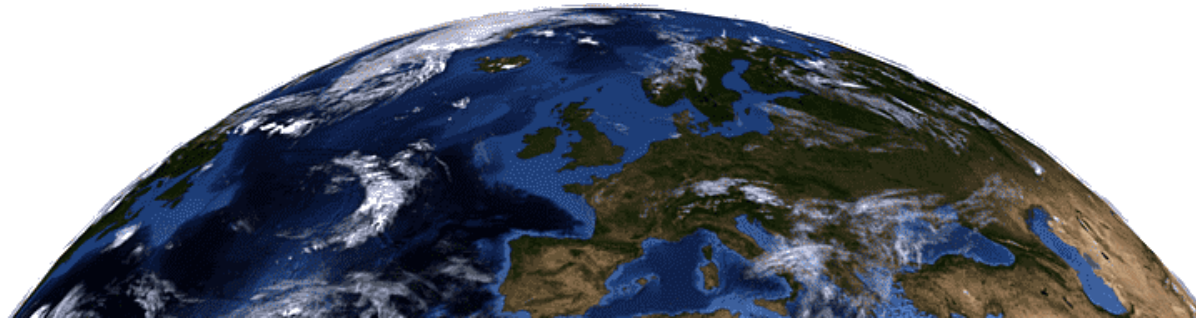
Note: The inspection tools provided are designed for testing different graph configuration but it is not recommended for high-resolution graphs with a high number of nodes/edges.

Encoder – Processor – Decoder

anemoi-training uses a specific **encoder-processor-decoder** graph structure:



- **Data Nodes:** Represent the locations of input/output data points (e.g., temperature, wind speed).
- **Hidden Nodes:**
 - Operate at a lower spatial resolution than the input/output nodes. ($\sim 1/2$)
 - Allow for feature extraction and dimensionality reduction, enabling the model to capture large-scale patterns.

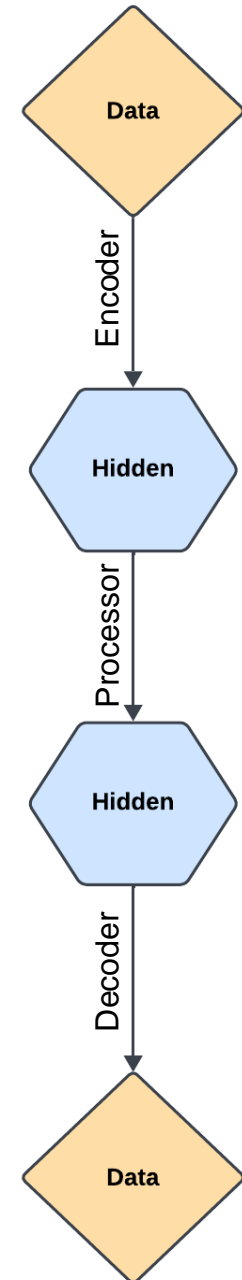


Note: For weather forecasting, the input and output nodes often correspond to the same physical locations. In other use cases (e.g., downscaling), input and output nodes may differ.

Graph recipe

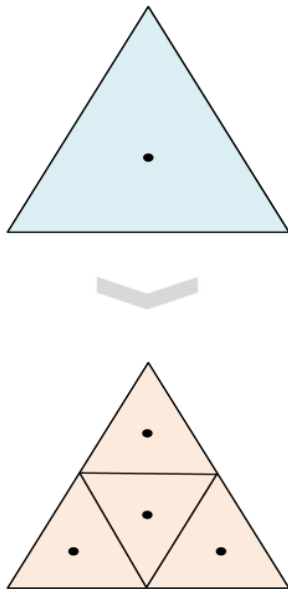
recipe.yaml

```
nodes:  
  data:  
    node_builder:  
      _target_: anemai.graphs.nodes.ZarrDatasetNodes  
      dataset: my_zarr_dataset  
  hidden:  
    node_builder:  
      _target_: anemai.graphs.nodes.TriNodes  
      resolution: 5 # num of refinements  
  
edges:  
  # Encoder configuration  
  - source_name: data  
    target_name: hidden  
    edge_builders:  
      - _target_: anemai.graphs.edges.CutOffEdges  
        cutoff_factor: 0.6  
  # Processor configuration  
  - source_name: hidden  
    target_name: hidden  
    edge_builders:  
      - _target_: anemai.graphs.edges.MultiScaleEdges  
        x_hops: 1  
  # Decoder configuration  
  - source_name: hidden  
    target_name: data  
    edge_builders:  
      - _target_: anemai.graphs.edges.KNNEdges  
        num_nearest_neighbours: 3
```

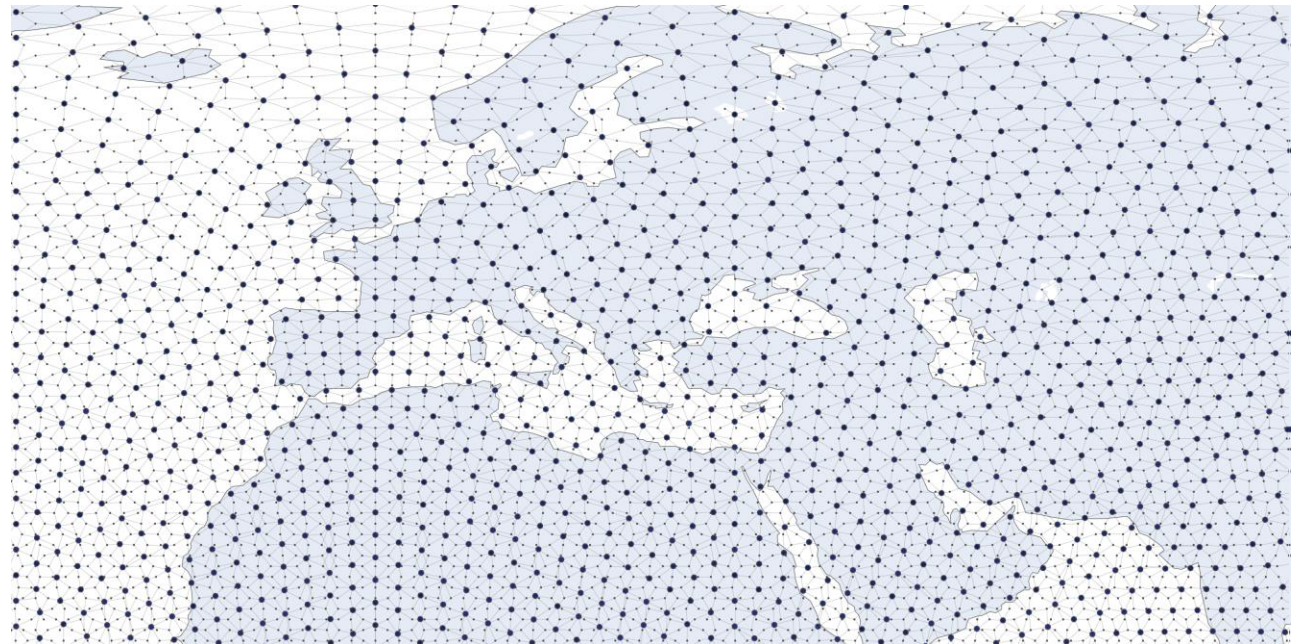


Triangular refined mesh

- It starts with an icosahedron (20 faces) projected to a sphere.
- Each refinement splits each face into 4 faces.



```
nodes:  
  hidden:  
    node_builder:  
      _target_: anemoi.graphs.nodes.TriNodes  
      resolution: 5 # num. of refinements
```



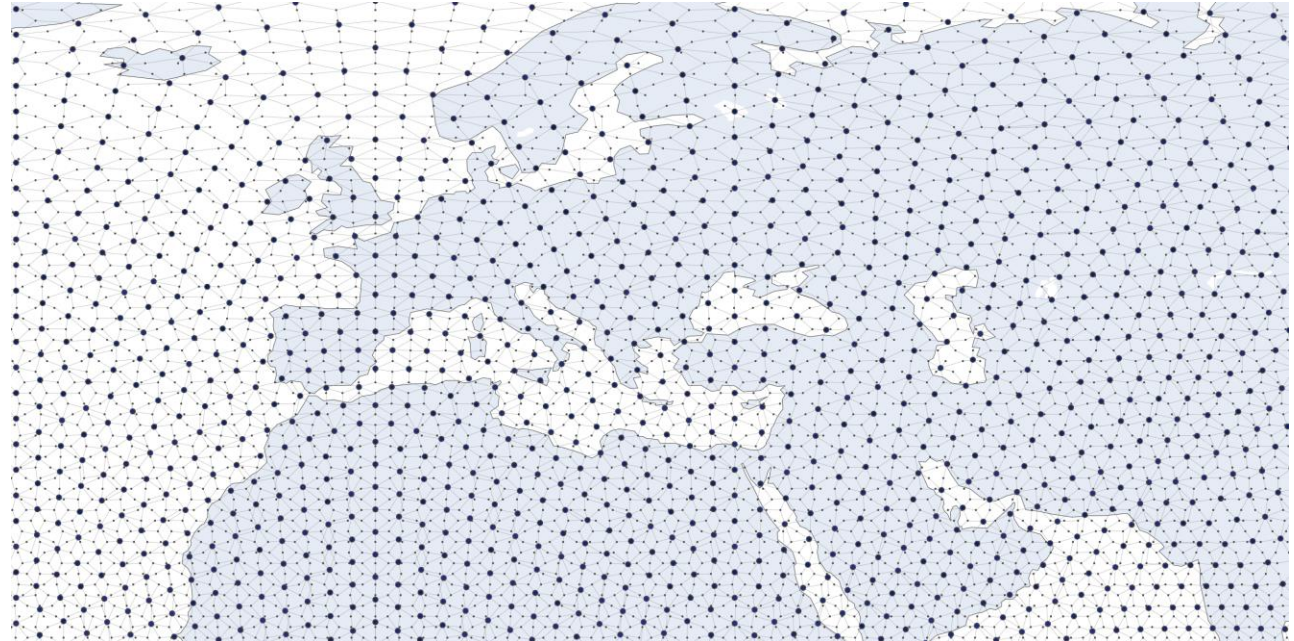
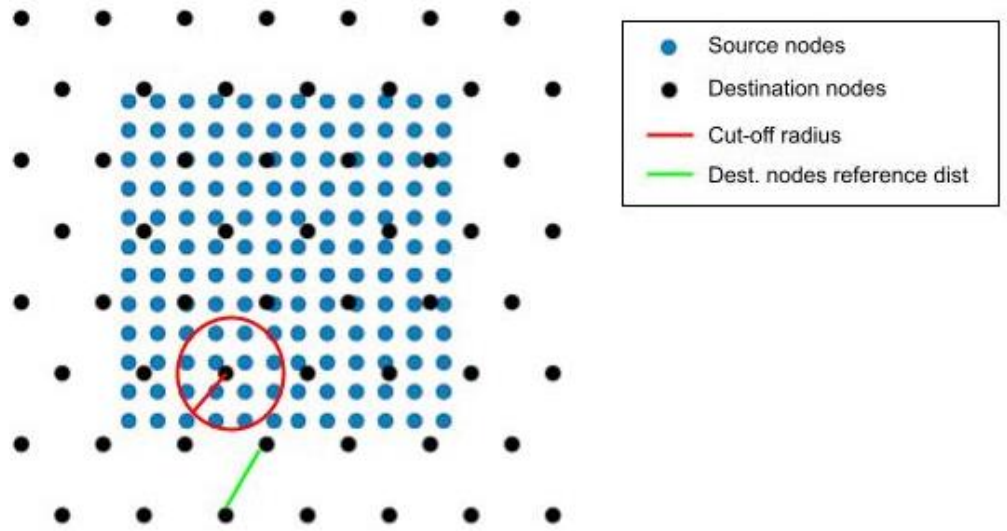
Encoder - CutOffEdges

For each target node, it connects all nodes within a given radius.

Why is this important?

- You want to maximise the information flowing to the hidden nodes.
- You want to minimise the number of edges (efficiency)

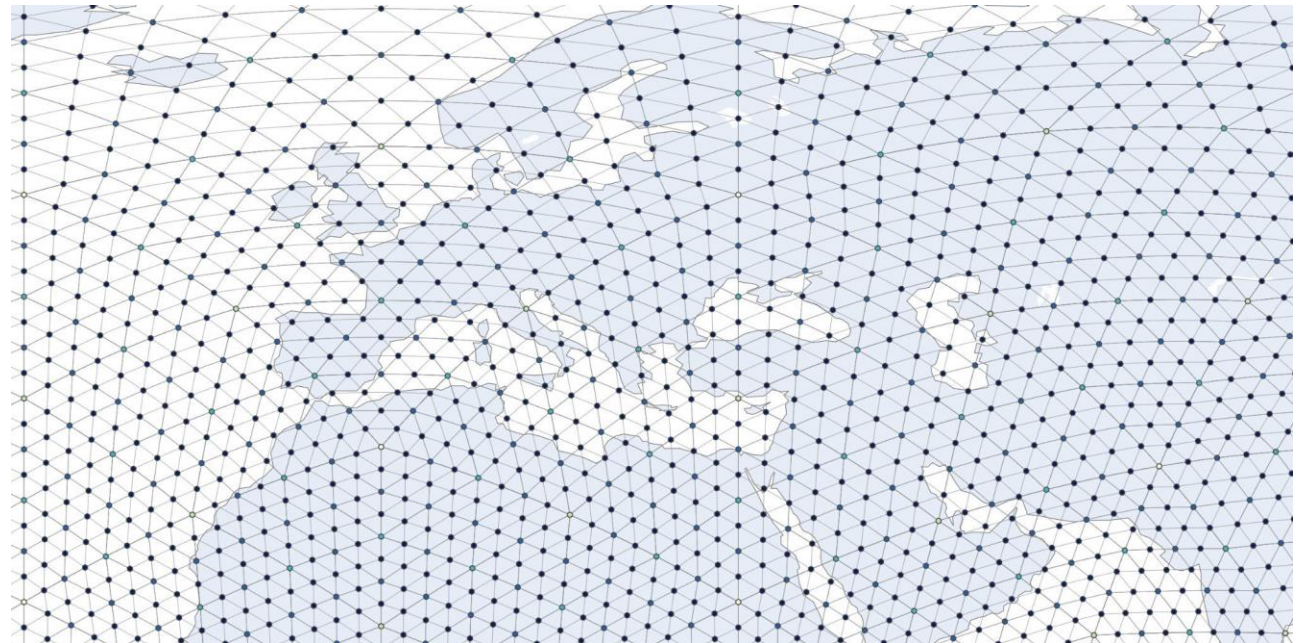
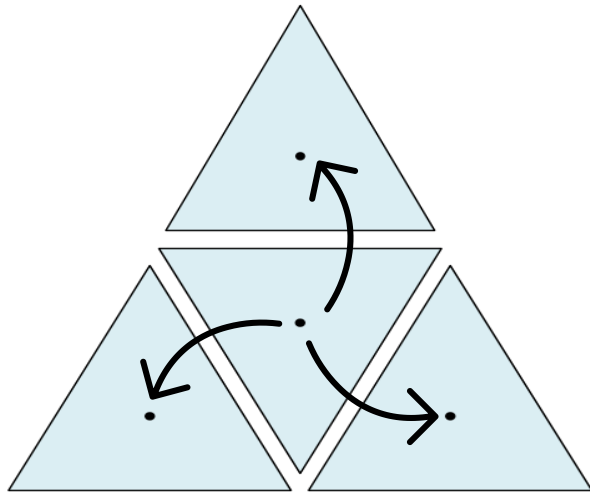
```
edges:  
- source_name: data  
  target_name: hidden  
  edge_builders:  
  - _target_: anemoi.graphs.edges.CutOffEdges  
    cutoff_factor: 0.6
```



Processor - MultiScaleEdges

- The processor connections define how far the information flows.
 - Connections are created for each level of refinement.

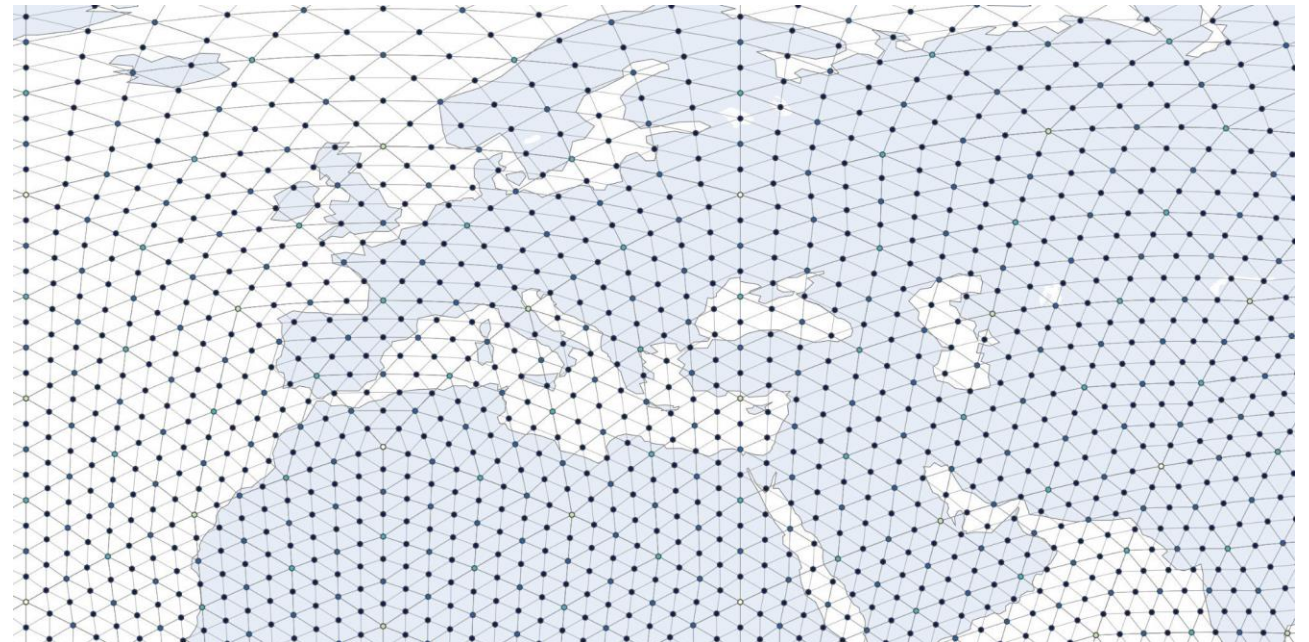
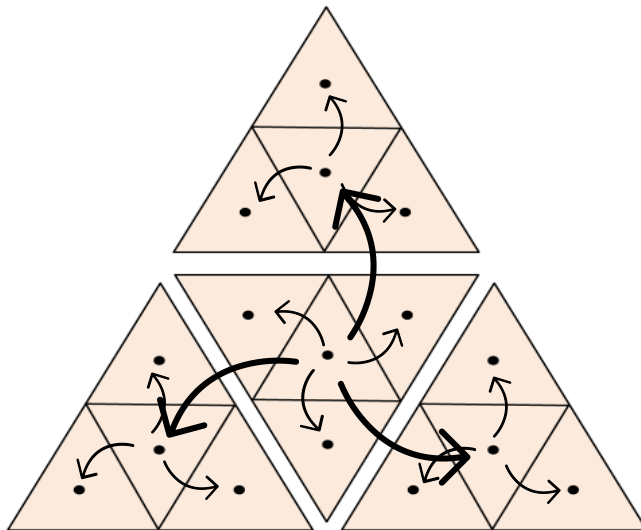
```
edges:  
- source_name: hidden  
  target_name: hidden  
  edge_builders:  
  - _target_: anemoi.graphs.edges.MultiScaleEdges  
    x_hops: 1
```



Processor - MultiScaleEdges

- The processor connections define how far the information flows.
 - Connections are created for each level of refinement.

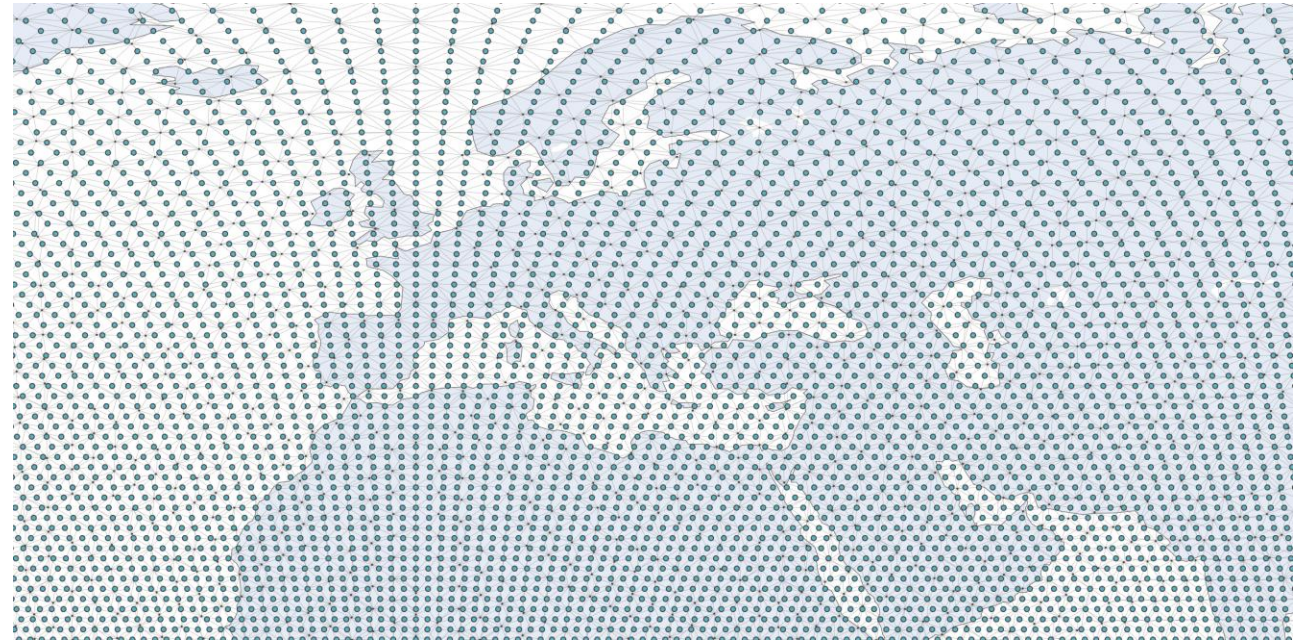
```
edges:  
- source_name: hidden  
  target_name: hidden  
  edge_builders:  
  - _target_: anemoi.graphs.edges.MultiScaleEdges  
    x_hops: 1
```



Decoder - KNNEdges

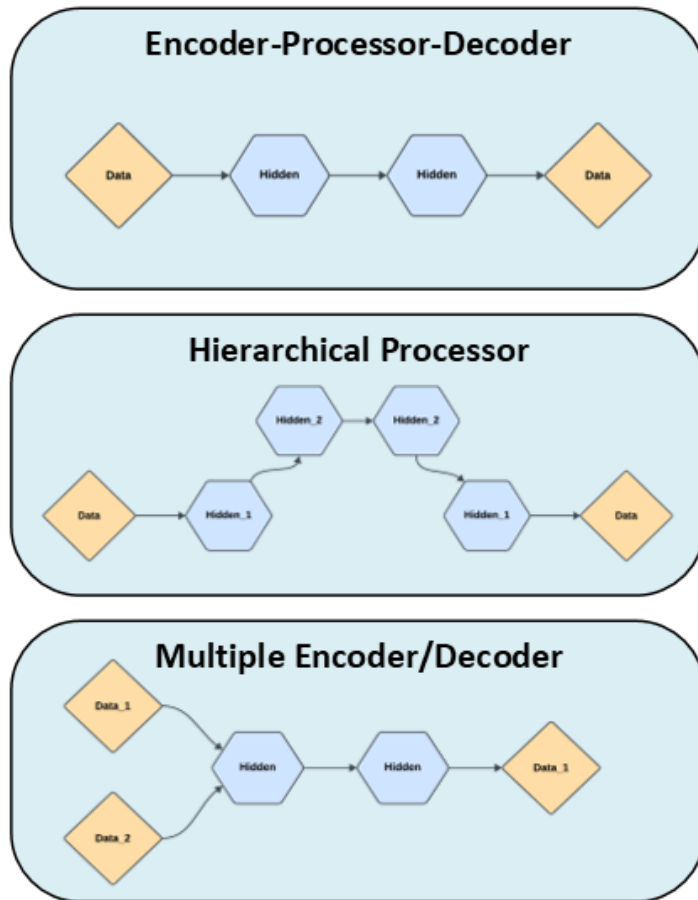
- For each target node, it connects the N nearest neighbours.

```
edges:  
- source_name: hidden  
  target_name: data  
  edge_builders:  
  - _target_: anemol.graphs.edges.KNNEdges  
    num_nearest_neighbours: 3
```



NOTE: The decoder is the part of the graph with more edges.

Graph configurations



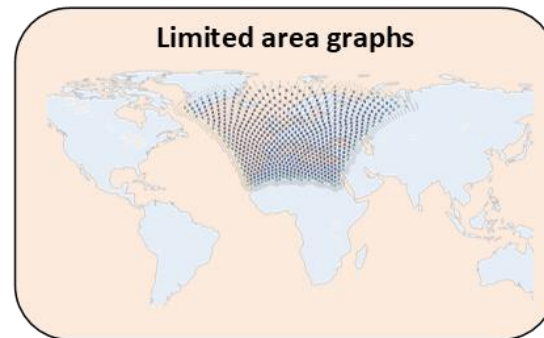
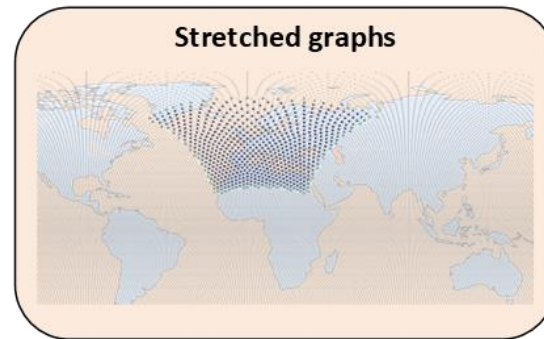
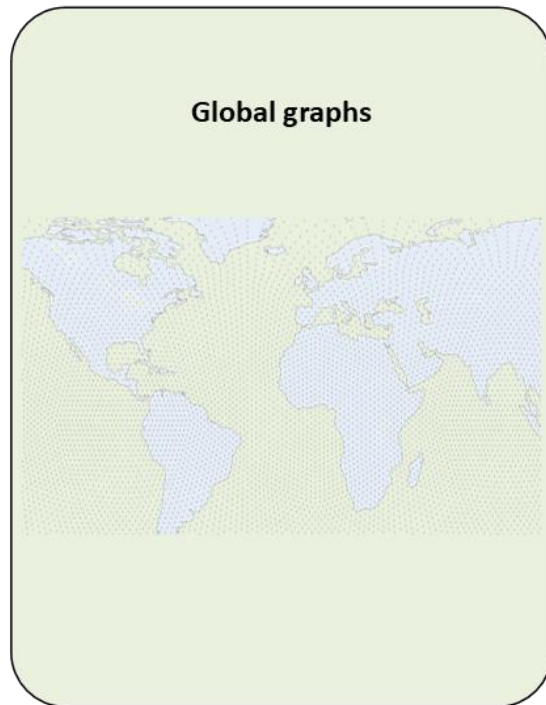
The screenshot displays the GitHub interface for the 'anemoi-core' repository, specifically the 'training' directory. The left sidebar shows the file tree with folders like 'docs', 'src', 'anemoi/training', 'commands', 'config', 'data', 'dataloader', 'diagnostics', 'graph', 'hardware', 'model', and 'training'. The main area shows a commit history table for the 'training' directory.

Name	Last commit message	Last commit date
..		
docs	Merge commit '38b75fadd5fcd9335547e8239180a92801...	last month
src	chore(training): Add default config files for 2 and 3 level hi...	19 hours ago
tests	build: fix isort and pre-commits	last month
.gitattributes	Add 'training/' from commit 'c99069ee00147e889c947f71...	last month
.gitignore	Add 'training/' from commit 'c99069ee00147e889c947f71...	last month
.readthedocs.yaml	docs: point RTD to right subfolder	last month
CHANGELOG.md	chore(training): Add default config files for 2 and 3 level hi...	19 hours ago
CONTRIBUTORS.md	Add 'training/' from commit 'c99069ee00147e889c947f71...	last month
LICENSE	Add 'training/' from commit 'c99069ee00147e889c947f71...	last month
README.md	docs: Tidy for core	last month
pyproject.toml	bump anemoi-datasets required version to 0.5.13 (#74)	yesterday
pytest.ini	Add 'training/' from commit 'c99069ee00147e889c947f71...	last month

Below the table, the 'anemoi-training' directory is highlighted, showing files like 'encoder_decoder_only.yaml', 'hierarchical_2level.yaml', 'hierarchical_3level.yaml', 'limited_area.yaml', 'multi_scale.yaml', 'stretched_grid.yaml', and 'config.yaml'.

Example [configuration files](#) for different use cases.

Use cases



Some user may focus in specific regions of interest. In this case, there are several options:

- **Stretched graphs**
 - Increases resolution over a region of interest while maintaining a coarser grid elsewhere.
- **Limited area graphs**
 - Nodes are restricted to a specific region.
 - A coarser dataset can be used as *boundary forcing*.

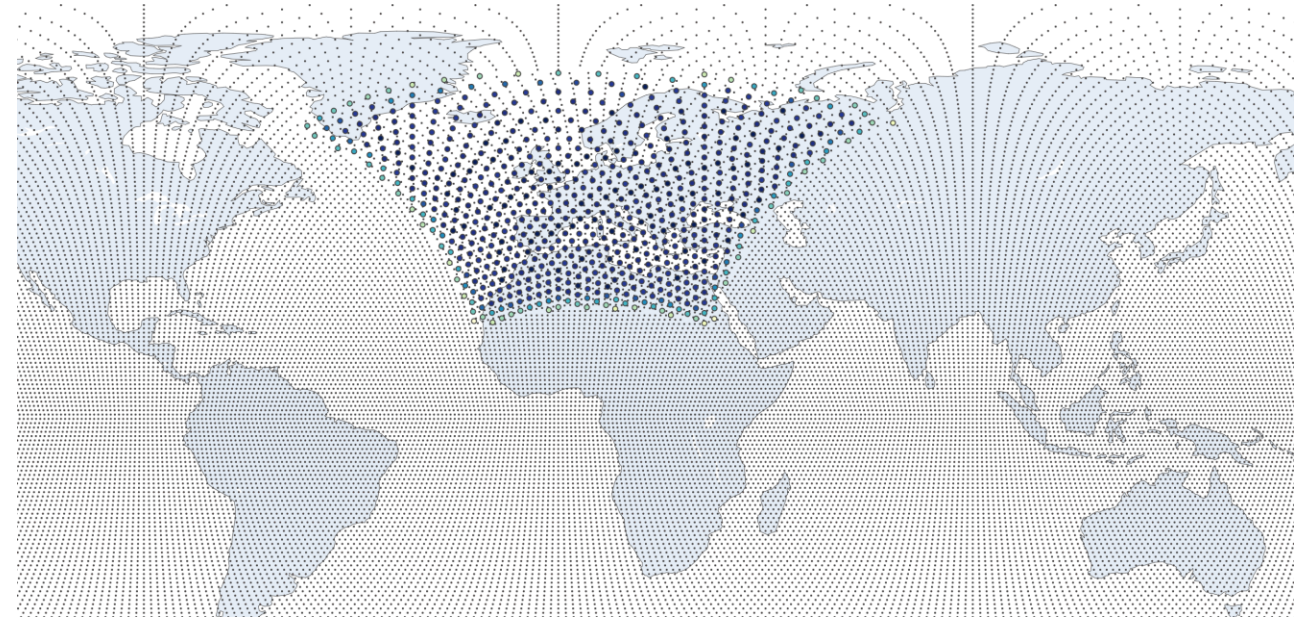
Regional modeling

```
nodes:  
  data:  
    node_builder:  
      _target_: anemoi.graphs.nodes.ZarrDatasetNodes  
    dataset:  
      cutout:  
        - my_local_zarr_dataset  
        - my_global_zarr_dataset  
      adjust: all  
    attributes:  
      area_weight:  
        _target_: anemoi.graphs.nodes.attributes.SphericalAreaWeights  
        norm: unit-max  
      cutout_mask:  
        _target_: anemoi.graphs.nodes.attributes.CutOutMask
```

Graph recipe file containing the data nodes configuration.

```
hidden:  
  node_builder:  
    _target_: anemoi.graphs.nodes.LimitedAreaTriNodes  
  resolution: 5  
  reference_node_name: data  
  node_attr_name: cutout_mask
```

Graph recipe file containing the nodes configuration for LAM.



Data and hidden nodes for a limited area graph.

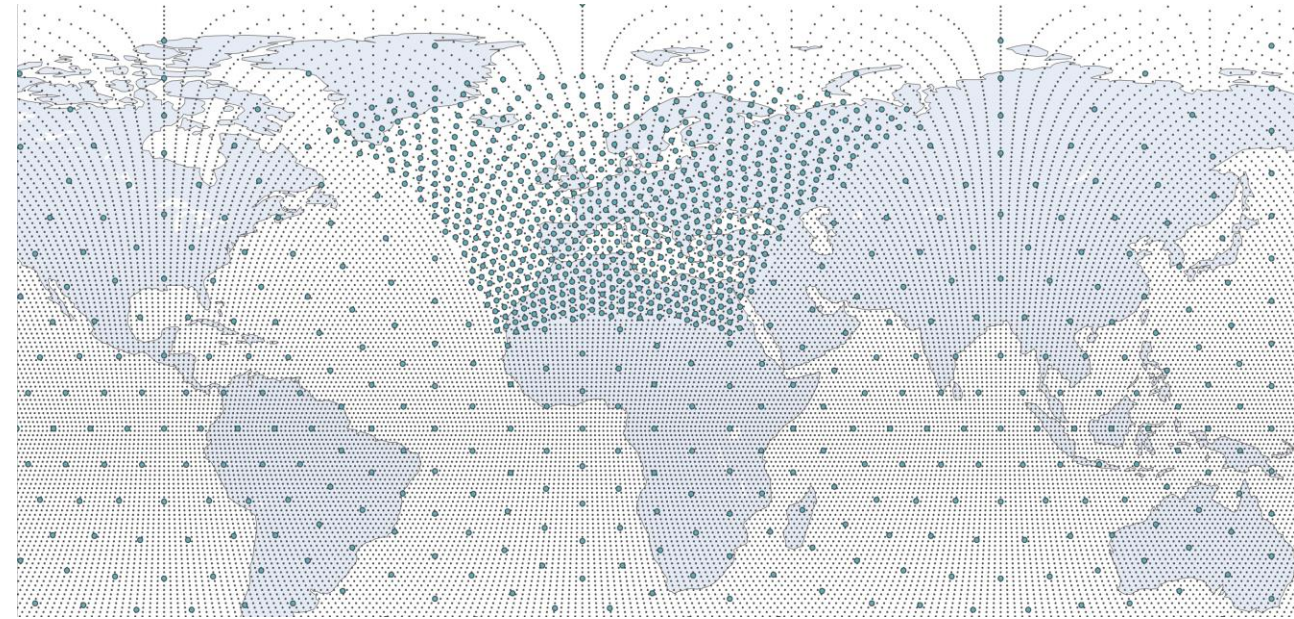
```
hidden:  
  node_builder:  
    _target_: anemoi.graphs.nodes.StretchedTriNodes  
  lam_resolution: 5  
  global_resolution: 3  
  reference_node_name: data  
  node_attr_name: cutout_mask
```

Graph recipe file containing the nodes configuration for stretched graphs.

Regional modeling

```
nodes:  
  data:  
    node_builder:  
      _target_: anemoi.graphs.nodes.ZarrDatasetNodes  
    dataset:  
      cutout:  
        - my_local_zarr_dataset  
        - my_global_zarr_dataset  
      adjust: all  
    attributes:  
      area_weight:  
        _target_: anemoi.graphs.nodes.attributes.SphericalAreaWeights  
        norm: unit-max  
      cutout_mask:  
        _target_: anemoi.graphs.nodes.attributes.CutOutMask
```

Graph recipe file containing the data nodes configuration.



Data and hidden nodes for a stretched graph.

```
hidden:  
  node_builder:  
    _target_: anemoi.graphs.nodes.LimitedAreaTriNodes  
  resolution: 5  
  reference_node_name: data  
  node_attr_name: cutout_mask
```

Graph recipe file containing the nodes configuration for LAM.

```
hidden:  
  node_builder:  
    _target_: anemoi.graphs.nodes.StretchedTriNodes  
  lam_resolution: 5  
  global_resolution: 3  
  reference_node_name: data  
  node_attr_name: cutout_mask
```

Graph recipe file containing the nodes configuration for stretched graphs.

Limited Area Modeling

```
edges:  
# Encoder configuration  
- source_name: data  
  target_name: hidden  
  edge_builders:  
    - _target_: anemoi.graphs.edges.CutOffEdges  
      cutoff_factor: 0.6  
  attributes:  
    edge_length:  
      _target_: anemoi.graphs.edges.attributes.EdgeLength  
# Processor configuration  
- source_name: hidden  
  target_name: hidden  
  edge_builders:  
    - _target_: anemoi.graphs.edges.MultiScaleEdges  
      x_hops: 1  
  attributes:  
    edge_length:  
      _target_: anemoi.graphs.edges.attributes.EdgeLength  
# Decoder configuration  
- source_name: hidden  
  target_name: data  
  edge_builders:  
    - _target_: anemoi.graphs.edges.KNNEdges  
      target_mask_attr_name: cutout_mask  
      num_nearest_neighbours: 3  
  attributes:  
    edge_length:  
      _target_: anemoi.graphs.edges.attributes.EdgeLength
```

Cont.: Graph recipe file containing the edge configuration for LAM.

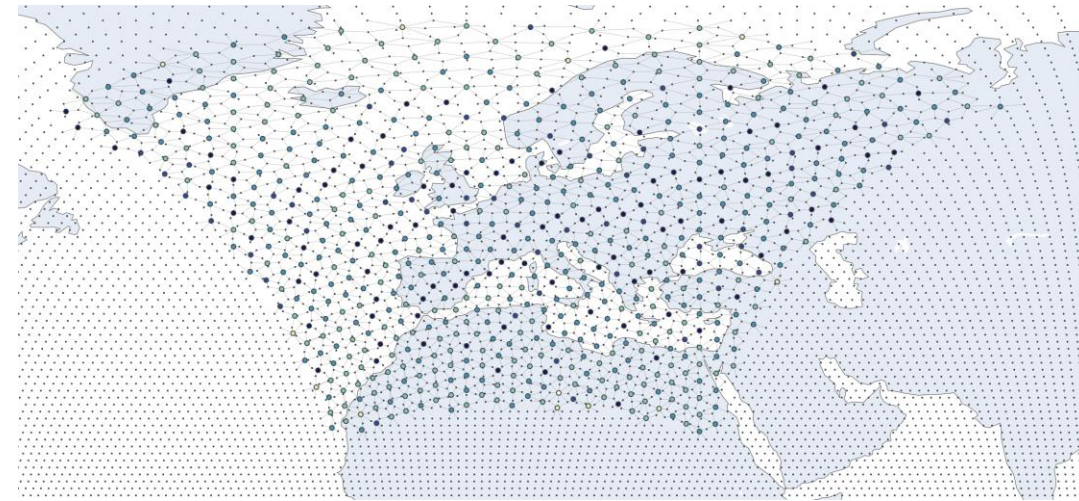


Diagram of encoder connections.

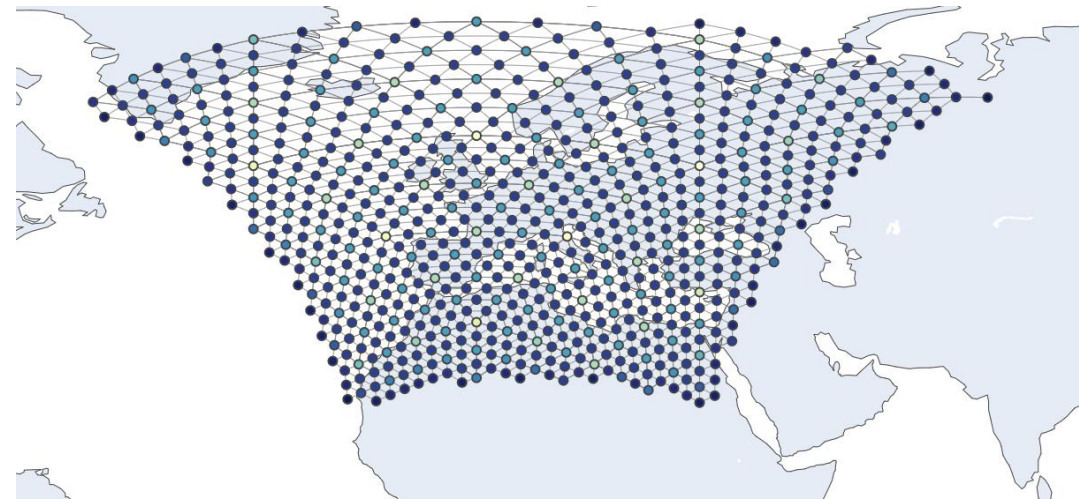


Diagram of processor connections.

Wrap up

- *anemoi-graphs* is used by *anemoi-training* to create the graphs on the fly.
- *anemoi-training* contains recipes for all use cases
- *anemoi-graphs* has a command line tool to create and inspect graphs
- *anemoi-graphs* is designed to be extended with new node, edge and attribute builders.
- *anemoi-graphs* supports different use cases:
 - Global graphs
 - Limited area graphs
 - Stretched graphs
- More complex setups (multi-encoder/decoder, dynamic graphs, ...) are not supported outside *anemoi-graphs*.
- Graph configuration plays a key role in the flow of information and the efficiency of the model.

Questions?

anemoui-graphs.readthedocs.io