Before we bring Yonggang in perhaps the 2 of us need to have a brainstorm session to fine tune the sampler UI design. Here are my initial thoughts:

* Much like regular output, we need a section where we specify geometries.
* We need another section to define collections on those geometries. Let's take 'stations' as an example. I can see a situation in which we write standard M21C collections on a single flight path. There is 1 station geometry, multiple collections. I do this consistently with my offlinre samplers.
* Right now, I see these geometries associated with different types of geolocation specifications:
  + **Station**, a collection of fixed points
  + **Trajectory**, a sequence of (time,lat,lon) with possibly 3 schemas for now:
    - schema = 'IODA', assumes that geolocation files are standard IODA files, so no need to to specify where the coordinate variables can be found.
    - schema = 'MACIE',  a particular convention for airborne data, where the variable names are established
    - schema = 'Generic', in this case you need to specify the index dimension, and the coordinate variable names. Think of IODA and MACIE as short-hands, nothing more. Simple things should be simple, complicated things possible.
    - There is no need for time to be monotonic.
  + **Swath.** Here again we could introduce schemas as short hands. **Swath** is a generalization of trajectories where for a given time you have 1D array array of coordinates, to the right and left of the sub point. For a **Swath**, the index is typically the time at the subpoint. So, it is better to have **Swath** as a separate class.
  + **Patch** or **Grid.** This is typically a logically rectangular grid. It is to **Station** what Swath is to **Trajectory.** It is not that dissimilar to a GEOS regional output (is this even implemented?), but the coordinate system is externally specified.
    - Instead of reading the coordinates from a file, one could instead specify the coordinates by means of standard geodetic Coordinate Reference System ("[geodetic CRS](https://ext.eurocontrol.int/aixm_confluence/display/ACG/Coordinate+Reference+System#:~:text=A%20geodetic%20CRS%20(e.g.%20EPSG,provided%20for%20each%20individual%20position.)"); there are well established conventions for this.
    - This device could be used to generate regional GEOS output, in whatever map projection one chooses
    - For now, read it from a file.
* Now once a geometry is specified, **collections** can be specified on these geometries
  + Given a geometry one has a choice of how to sample the model variables:
    - **Interpolate** model variables to the geometry, either linearly or nearest neighbors. This is the only option for **Station** and **Trajectory**, but could be be used for **Swath** and **Patch** as well.
    - **Regrid** the model fields to the geometry. This only makes sense for **Swath** or **Patch**.
    - **Mask.** In this case one masks out those grid points outside the geolocations, given a halo width. It would be sufficient to have cubed-sphere output in this case. This could be implemented for ALL geometries, but **Swath and Patch** are the priorities.
  + The time frequency of the sampled output depends on the geometry:
    - **Station** and **Patch**. The user specifies a time frequency like regular gridded output, usually a multiple of the heartbeat.
    - **Trajectory** and **Swath**: in this case the time sampling is dictated by the geometry. One could specify a time stride, but that stride could also be specified when one defines the geometry. So, no time frequency specification for the collection, it would be simpler to document.
  + As for the regular collections, one specifies a list of variables. Retaining symmetry with the regular collections would make sense. For example, "\_\_\_\_.fields:" instead of "\_\_\_\_\_.geoval\_fields".
  + Likewise, retain consistency with "regrid\_method", etc.

Let's brainstorm.  For the immediate future we will make due with what he has, but a gradual refactoring is in order. Do you agree?