Towards the first CTAO Science Data Challenge

Fabio Pintore
INAF/IASF Palermo

On behalf of the technical task team (Giacomo Principe, Kazuma Ishio, Sabrina Einecke, Dario Gasparrini, Roberta Zanin) and much more…

The first CTAO SDC - nutshell and goals

SDC#1 in short:

- Simulations of VHE sky as seen by the CTAO Alpha Configuration over a seven-year temporal period, from Jan 1st 2028 to Dec 31st, 2034
- 3 zenith bins (20-40-60 deg) and 5 times the night sky background (NSB)
- for the azimuth, average of two directions (south + north).
- Bad weather: already included in the duty-cycle (about 10% of the total exposure time)

- blind challenge

- simulated data and SATs will be publicly released

Observational Projects

The observational projects can be subdivided in the following main core projects.

Four of them are included in the **sky surveys**:

- Galactic plane
- extra-galactic sky
- Large Magellanic cloud
- Perseus cluster

while the remaining are mostly connected to **variable and exotic sources**, namely:

- AGN monitoring (15 AGNs)
- ToO dedicated to follow up of neutrino events, Galactic transients and MWL transient sources,..
- GRBs (rapid response)
- GW follow up (pointing scan)
- other dark matter sources

In addition, possible OFF observations may be provided for tests.

General SDC Status

- **Sky models** ✓
- **Analysis of the observational projects as input for the scheduling** ✓
- **Scheduling observations** ✓ **ON-GOING**
- **Preparation of simulations**
  - **Software-wise (gammapy)** ✓ **FEW ISSUES OPEN**
  - **Production (datacenter)** ✓ **CSCS IN SWITZERLAND**

Sky models: status

All models have been collected!

Sky-Models were provided in various formats:

- XML format (Ctools)
- YAML format (Gammapy)
- Other formats

→ All models will be converted in YAML as simulations will be performed with Gammapy (version 1.x, TBC).

Sky models: Gammapy side

- No known issues in Gammapy for simulating stationary sources

- Not easy simulations for time-dependent sources. Different formats for these data

Sky models: Gammapy side

1) e.g. AGN

fv: Binary Table of DC_Event7.fits[1] in /Users/fabiopintore/LAVORO/CTA/DATA_CHALLENGE/MODELS/Models/GRB-Models/

File Edit Tools Help

	time_start	time_stop	mod
Select	D	E	3961A
All	s	s	
Invert	Modify	Modify	Modify

1	0.000000000000E+00	1.240000E+00	components:\n- name: model_0\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
2	1.240000009537E+00	1.930000E+00	components:\n- name: model_1\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
3	1.929999947548E+00	3.020000E+00	components:\n- name: model_2\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
4	3.019999980927E+00	4.390000E+00	components:\n- name: model_3\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
5	4.389999866486E+00	5.810000E+00	components:\n- name: model_4\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
6	5.809999942780E+00	7.990000E+00	components:\n- name: model_5\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
7	7.989999771118E+00	1.020000E+01	components:\n- name: model_6\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
8	1.019999980927E+01	1.320000E+01	components:\n- name: model_7\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
9	1.319999980927E+01	1.690000E+01	components:\n- name: model_8\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
10	1.689999961853E+01	2.150000E+01	components:\n- name: model_9\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
11	2.150000000000E+01	2.750000E+01	components:\n- name: model_10\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
12	2.750000000000E+01	3.520000E+01	components:\n- name: model_11\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
13	3.520000076294E+01	4.430000E+01	components:\n- name: model_12\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
14	4.429999923706E+01	5.580000E+01	components:\n- name: model_13\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
15	5.579999923706E+01	7.030000E+01	components:\n- name: model_14\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
16	7.030000305176E+01	8.790000E+01	components:\n- name: model_15\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
17	8.790000152588E+01	1.100000E+02	components:\n- name: model_16\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
18	1.100000000000E+02	1.380000E+02	components:\n- name: model_17\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
19	1.380000000000E+02	1.720000E+02	components:\n- name: model_18\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
20	1.720000000000E+02	2.150000E+02	components:\n- name: model_19\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T

Go to: Edit cell:

2) e.g GW - GRB

fv: Summary of catO5_0.fits in /Users/fabiopintore/LAVORO/CTA/DATA_CHALLENGE/MODELS/Models/...

File Edit Tools Help

Index	Extension	Type	Dimension	View
0	Primary	Image	0	Header Image Table
1	ENERGIES	Binary	1 cols X 41 rows	Header Hist Plot All Select
2	TIMES	Binary	2 cols X 70 rows	Header Hist Plot All Select
3	SPECTRA	Binary	41 cols X 70 rows	Header Hist Plot All Select

Sky models: Gammapy side

1) e.g. AGN

fv: Binary Table of DC_Event7.fits[1] in /Users/fabiopintore/LAVORO/CTA/DATA_CHALLENGE/MODELS/Models/GRB-Models/

File Edit Tools Help

	time_start	time_stop	mod
Select	D	E	3961A
All	s	s	
Invert	Modify	Modify	Modify

1	0.000000000000E+00	1.240000E+00	components:\n- name: model_0\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
2	1.240000009537E+00	1.930000E+00	components:\n- name: model_1\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
3	1.929999947548E+00	3.020000E+00	components:\n- name: model_2\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
4	3.019999980927E+00	4.390000E+00	components:\n- name: model_3\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
5	4.389999866486E+00	5.810000E+00	components:\n- name: model_4\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
6	5.809999942780E+00	7.990000E+00	components:\n- name: model_5\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
7	7.989999771118E+00	1.020000E+01	components:\n- name: model_6\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
8	1.019999980927E+01	1.320000E+01	components:\n- name: model_7\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
9	1.319999980927E+01	1.690000E+01	components:\n- name: model_8\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
10	1.689999961853E+01	2.150000E+01	components:\n- name: model_9\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: Te
11	2.150000000000E+01	2.750000E+01	components:\n- name: model_10\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
12	2.750000000000E+01	3.520000E+01	components:\n- name: model_11\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
13	3.520000076294E+01	4.430000E+01	components:\n- name: model_12\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
14	4.429999923706E+01	5.580000E+01	components:\n- name: model_13\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
15	5.579999923706E+01	7.030000E+01	components:\n- name: model_14\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
16	7.030000305176E+01	8.790000E+01	components:\n- name: model_15\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
17	8.790000152588E+01	1.100000E+02	components:\n- name: model_16\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
18	1.100000000000E+02	1.380000E+02	components:\n- name: model_17\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
19	1.380000000000E+02	1.720000E+02	components:\n- name: model_18\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T
20	1.720000000000E+02	2.150000E+02	components:\n- name: model_19\n type: SkyModel\n spectral:\n type: CompoundSpectralModel\n model1:\n type: T

Go to: Edit cell:

2) e.g GW - GRB

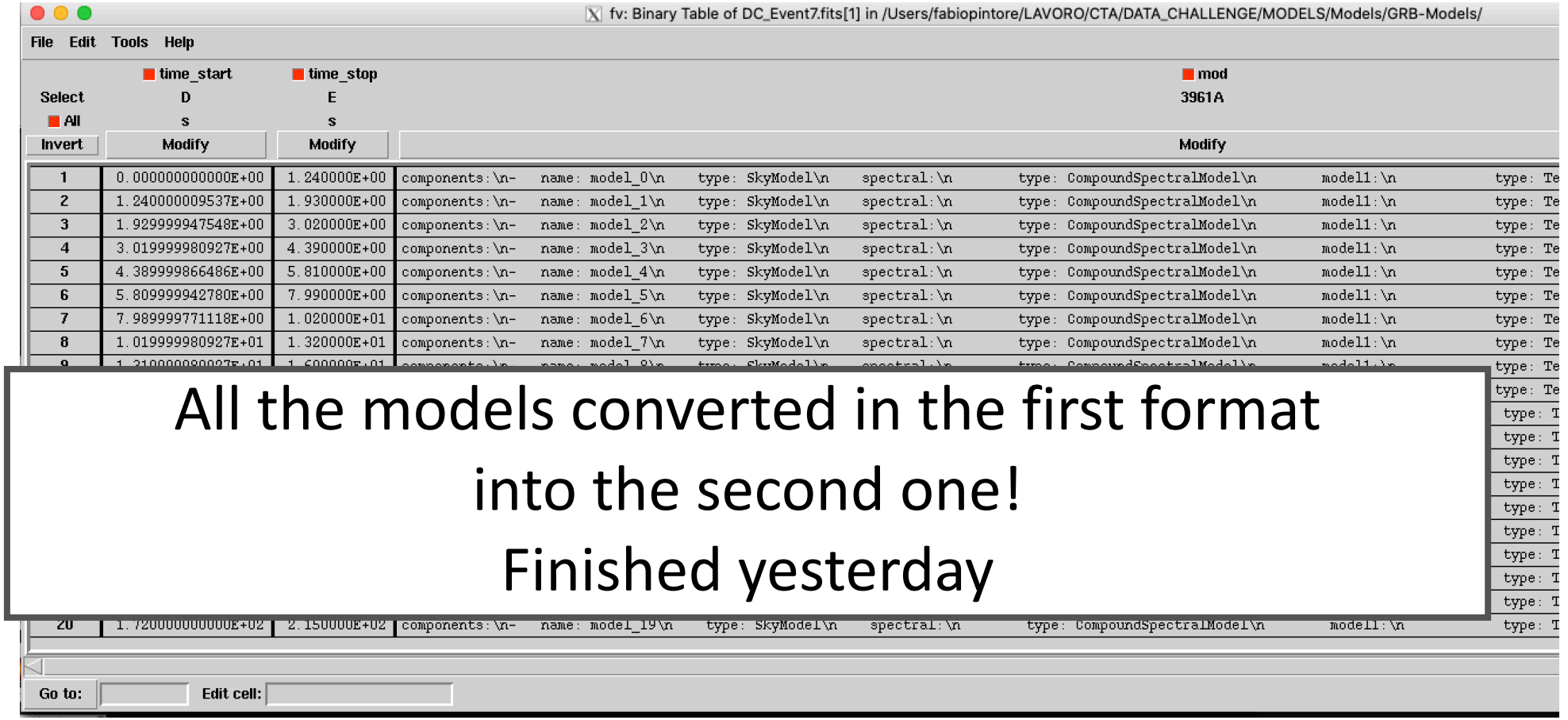
fv: Summary of catO5_0.fits in /Users/fabiopintore/LAVORO/CTA/DATA_CHALLENGE/MODELS/Models/...

File Edit Tools Help

Index	Extension	Type	Dimension	View
0	Primary	Image	0	Header Image Table
1	ENERGIES	Binary	1 cols X 41 rows	Header Hist Plot All Select
2	TIMES	Binary	2 cols X 70 rows	Header Hist Plot All Select
3	SPECTRA	Binary	41 cols X 70 rows	Header Hist Plot All Select

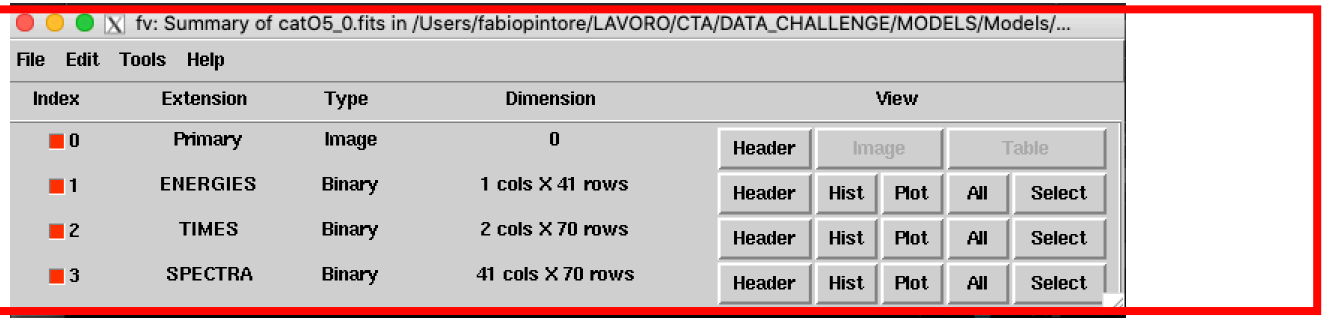
Sky models: Gammapy side

1) e.g. AGN



All the models converted in the first format into the second one!
Finished yesterday

2) e.g GW - GRB



Sky models: Gammapy side

Atreyee developed a function to read this format in her git branch in *gammapy.modeling.models.utils* (specific for this DC and not available for the rest of the “world”)

```
def _read_cta_sdc(filename):
    """To create a `LightCurveTemplateTemporalModel`
    from the cta-sdc files. This format is subject to change"""
    with fits.open(filename) as hdul:
        position = SkyCoord(
            ra=hdul[0].header["LONG"] * u.deg,
            dec=hdul[0].header["LAT"] * u.deg,
            frame="icrs",
        )

        energy_hdu = hdul["ENERGIES"]
        energy_axis = MapAxis.from_nodes(
            nodes=energy_hdu.data,
            unit=energy_hdu.header["TUNIT1"],
            name="energy",
            interp="log",
        )
        time_hdu = hdul["TIMES"]
        time_header = time_hdu.header
        time_header.setdefault("MJDREFF", 0.5)
        time_header.setdefault("MJDREFI", 55555)
        time_min = time_hdu.data["Initial Time"]
        time_max = time_hdu.data["Final Time"]
        edges = np.append(time_min, time_max[-1]) * u.Unit(time_header["TUNIT1"])
        time_ref = time_ref_from_dict(time_header)
        time_axis = MapAxis.from_edges(edges=edges, name="time", interp="log")
        data = hdul["SPECTRA"]
        return (
            RegionNDMap.create(
                region=PointSkyRegion(center=position),
                axes=[energy_axis, time_axis],
                data=np.array(list(data.data) * u.Unit(data.header["UNITS"])),
            ),
            time_ref,
        )
```

Sky models: Gammapy side

Atreyee developed a function to read this format in her git branch in *gammapy.modeling.models.utils* (specific for this DC and not available for the rest of the “world”)

```python
def _read_cta_sdc(filename):
    """To create a `LightCurveTemplateTemporalModel`
    from the cta-sdc files. This format is subject to change"""
    with fits.open(filename) as hdul:
        position = SkyCoord(
            ra=hdul[0].header["LONG"] * u.deg,
            dec=hdul[0].header["LAT"] * u.deg,
            frame="icrs",
        )

        energy_hdu = hdul["ENERGIES"]
        energy_axis = MapAxis.from_nodes(
            nodes=energy_hdu.data,
            unit=energy_hdu.header["TUNIT1"],
            name="energy",
            interp="log",
        )
        time_hdu = hdul["TIMES"]
        time_header = time_hdu.header
        time_header.setdefault("MJDREFF", 0.5)
        time_header.setdefault("MJDREFI", 55555)
        time_min = time_hdu.data["Initial Time"]
        time_max = time_hdu.data["Final Time"]
        edges = np.append(time_min, time_max[-1]) * u.Unit(time_header["TUNIT1"])
        time_ref = time_ref_from_dict(time_header)
        time_axis = MapAxis.from_edges(edges=edges, name="time", interp="log")
        data = hdul["SPECTRA"]
        return (
            RegionNDMap.create(
                region=PointSkyRegion(center=position),
                axes=[energy_axis, time_axis],
                data=np.array(list(data.data) * u.Unit(data.header["UNITS"])),
            ),
            time_ref,
        )
```

The function is then called by *LightCurveTemplateTemporalModel* model in *gammapy-modeling.models.temporal*

Sky models: Gammapy side

Atreyee developed a function to read this format in her git branch in *gammapy.modeling.models.utils* (specific for this DC and not available for the rest of the “world”)

```
def _read_cta_sdc(filename):
    """To c
    from th
    with fi
        posi
        )
        ener
        ener
        )
        time
        time
        time
        time
        time
        time
        edge
        time
        time
        data
        retu
            region=PointSkyRegion(center=position),
            axes=[energy_axis, time_axis],
            data=np.array(list(data.data) * u.Unit(data.header["UNITS"])),
        ),
        time_ref,
    )
```

How do we pass this model to *SkyModel*?
There is an issue with the *_check_units* in the *SkyModel* initialization as it tries to evaluate the model.

Other issue with the expected *spectral_model,* that it is mandatory

Sky models: Gammapy side

```python
tag = ["SkyModel", "sky-model"]
_apply_irf_default = {"exposure": True, "psf": True, "edisp": True}

def __init__(
    self,
    spectral_model,
    spatial_model=None,
    temporal_model=None,
    name=None,
    apply_irf=None,
    datasets_names=None,
):
    self.spatial_model = spatial_model
    self.spectral_model = spectral_model
    self.temporal_model = temporal_model
    self._name = make_name(name)

    if apply_irf is None:
        apply_irf = self._apply_irf_default.copy()

    self.apply_irf = apply_irf
    self.datasets_names = datasets_names
    self._check_unit()
```

gammapy.modeling.models.SkyModel

```python
def _check_unit(self):
    from gammapy.data.gti import GTI

    # evaluate over a test geom to check output unit
    # TODO simpler way to test this ?
    axis = MapAxis.from_energy_bounds(
        "0.1 TeV", "10 TeV", nbin=1, name="energy_true"
    )

    geom = WcsGeom.create(skydir=self.position, npix=(2, 2), axes=[axis])

    gti = GTI.create(1 * u.day, 2 * u.day)
    value = self.evaluate_geom(geom, gti)

    if self.apply_irf["exposure"]:
        ref_unit = u.Unit("cm-2 s-1 MeV-1 sr-1")
    else:
        ref_unit = u.Unit("sr-1")

    if self.spatial_model is None:
        ref_unit = ref_unit / u.Unit("sr-1")

    if not value.unit.is_equivalent(ref_unit):
        raise ValueError(
            f"SkyModel unit {value.unit} is not equivalent to {ref_unit}"
        )
```

It raises a *ValueError*. Add a condition to pass when the model is from the cta-sdc?

Sky models: Gammapy side

The next step is to implement the functionality to sample the events (*gammapy.dataset.MapDatasetEventSampler*)

```python
def _evaluate_timevar_source(self, dataset, evaluator, time_axis=None):
    """Calculate Npred for a given `dataset.model` by evaluating
    it in region geometry.

    Parameters
    ----------
    dataset : `~gammapy.datasets.MapDataset`
        Map dataset.
    evaluator : `~gammapy.datasets.evaluators.MapEvaluator`
        Map evaluator.
    time_axis : `~gammapy.Maps.MapAxis`
        Axis of the time.

    Returns
    -------
    npred : `~gammapy.maps.Map`
        Npred map.
    """
    energy_axis = MapAxis.from_edges(
        dataset.geoms["geom"].axes["energy"].edges, name="energy_true"
    )

    target = evaluator.model.spatial_model.position
    on_region = PointSkyRegion(center=target)
    region_geom = RegionGeom.create(on_region, axes=[energy_axis])

    flux = evaluator.model.evaluate_geom(region_geom.to_wcs_geom(),
        gti=dataset.gti)
    region_exposure =
        dataset.exposure.to_region_nd_map(region_geom.center_skydir)

    npred = flux * region_exposure * dataset.geoms["geom"].bin_volume()

    return npred
```

don’t work…

```python
def _sample_coord_time_energy(self, dataset, evaluator, t_delta="1 s"):
    """Sample model components of a source with time-dependent spectrum.

    Parameters
    ----------
    dataset : `~gammapy.datasets.MapDataset`
        Map dataset.
    evaluator : `~gammapy.datasets.evaluators.MapEvaluator`
        Map evaluator.
    t_delta : `~astropy.units.Quantity`
        Minimum step time.

    Returns
    -------
    table : `~astropy.table.Table`
        Table of sampled events.
    """
    if not isinstance(evaluator.model.spatial_model, PointSpatialModel):
        raise TypeError(
            f"Event sampler expects PointSpatialModel for a time varying
                source. Got {evaluator.model.spatial_model} instead."
        )

    else:
        raise NotImplementedError("The functionality is not yet implemented")

    return
```