Saved

April 17, 2018

1 Installing the packages

We install the Holoviews package first:

- CONDA: conda install holoviews conda install geoviews
- PIP: pip install 'holoviews[all]'
- RAW: git clone git://github.com/ioam/holoviews.git cd holoviews pip install -e .

2 The basics

```
In [1]: #%% Importing packages

        # Holoviews is loaded
        import holoviews as hv

        # Geoviews needs to be loaded as well (quite heavy to load if not used)
        import geoviews as gv

        # Features are often used, shortcut is handy
        import geoviews.feature as gf

        # The packages is based on Xarray, so this is kinda handy
        import xarray as xr

        # Cartopy is used to make the maps
        from cartopy import crs


        # There is a tab-completion issue, but that can be fixed
        hv.extension(case_sensitive_completion=True)
        # Holoviews uses an ~/.holoviews.rc file at start-up, this line can be placed there
```

```
WARNING: param.Version now supports PEP440 and a new tag based workflow. See param/version.py fo
WARNING: param.Version now supports PEP440 and a new tag based workflow. See param/version.py fo
```

```
<IPython.core.display.HTML object>
```

```
In [2]: #%% Loading some data (Multi-File data loading is used here as an example)
        ds_psl = xr.open_mfdataset('/nobackup_3/users/stoop/7A_weather_data/psl_d_ECEarth_2C_s16
        ds_rsds = xr.open_mfdataset('/nobackup_3/users/stoop/7A_weather_data/rsds_d_ECEarth_2C_s
        ds_wind = xr.open_mfdataset('/nobackup_3/users/stoop/7A_weather_data/sfcwind_d_ECEarth_2
        ds_temp = xr.open_mfdataset('/nobackup_3/users/stoop/7A_weather_data/tas_d_ECEarth_2C_s1
```

```
In [3]: #%% Asignment of dimensions (GeoViews can use Iris cubes, Xarray data or just Numpy data

        # Set the key dims (coordinates), the ones that are not part of the image will be slider
        kdims = ['time', 'lat', 'lon']

        # Load the GeoViews dataset (data, key dimensions, variable dimensions, projection)
        gv_psl= gv.Dataset(ds_psl, kdims=kdims, vdims='psl', crs=crs.PlateCarree())
        gv_rsds= gv.Dataset(ds_rsds, kdims=kdims, vdims='rsds', crs=crs.PlateCarree())
        gv_wind= gv.Dataset(ds_wind, kdims=kdims, vdims='sfcwind', crs=crs.PlateCarree())
        gv_temp= gv.Dataset(ds_temp, kdims=kdims, vdims='tas', crs=crs.PlateCarree())
```

```
In [4]: # Make an image of the data
        gv_temp.to.image(['lon','lat'])
```

```
Out[4]: :HoloMap   [time]
           :Image   [lon,lat]   (tas)
```

3 Changing figure characteristics

```
In [5]: # We want a colorbar
        %opts Image {+framewise} [colorbar=True] Curve [xrotation=60]

        # Try again
        gv_temp.to.image(['lon','lat'])
```

```
Out[5]: :HoloMap   [time]
           :Image   [lon,lat]   (tas)
```

```
In [6]: #%% Add a point on th map to track how temperature changes

        # Set the options
        %opts Curve [aspect=2 xticks=4 xrotation=15] Points (color='k')

        # Select the temperature curve (utrecht)
        temp_curve = hv.Curve(ds_temp.sel(lon=5.1, lat=52, method='nearest'), kdims=['time'])

        # Add a point on the temperature map
        temp_map = gv_temp.to(gv.Image,['lon', 'lat']) * gv.Points([(5.1,52)], crs=crs.PlateCarr

        # Plot them both
        temp_map + temp_curve
```

```
Out[6]: :Layout
           .HoloMap.I :HoloMap   [time]
              :Overlay
                 .Image.I  :Image   [lon,lat]   (tas)
                 .Points.I :Points   [Longitude,Latitude]
           .Curve.I   :Curve   [time]   (tas)
```

4 Adding features and different maps

```
In [ ]: #%% Adding contours

        temp_map * gf.coastline * gf.borders
```

```
In [ ]: #%% Changing the projection
        %opts Feature [projection=crs.Geostationary()]

        temp_map * gf.coastline
```

Simmilair features can be made with Bokeh