```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import rcParams
from scipy import io
import os

rcParams.update({'font.size': 18})
plt.rcParams['figure.figsize'] = [8, 16]

X = pd.read_table("War13_X.csv",
delimiter=";").select_dtypes(include=[np.number]).to_numpy()
Xprime = pd.read_table("War13_Xprime.csv", delimiter=";",
decimal=",").select_dtypes(include=[np.number]).to_numpy()

def DMD(X,Xprime,r):
    U,Sigma,VT = np.linalg.svd(X,full_matrices=0) # Step 1
    Ur = U[:,:r]
    Sigmar = np.diag(Sigma[:r])
    VTr = VT[:r,:]
    Atilde = np.linalg.solve(Sigmar.T,(Ur.T @ Xprime @ VTr.T).T).T #
Step 2
    Lambda, W = np.linalg.eig(Atilde) # Step 3
    Lambda = np.diag(Lambda)

    Phi = Xprime @ np.linalg.solve(Sigmar.T,VTr).T @ W # Step 4
    alpha1 = Sigmar @ VTr[:,0]
    b = np.linalg.solve(W @ Lambda,alpha1)
    return Phi, Lambda, b

Phi, Lambda, b = DMD(X[:,:-1],X[:,1:],21)

print("Parametry kształtu Phi:", Phi.shape)
```

```
Parametry kształtu Phi: (22, 2)
```

```
## Plot Mode 2
vortmin = -5
vortmax = 5
V2 = np.copy(np.real(np.reshape(Phi[:,1],(11,2))))
V2 = V2.T

# normalize values... not symmetric
minval = np.min(V2)
maxval = np.max(V2)

if np.abs(minval) < 5 and np.abs(maxval) < 5:
    if np.abs(minval) > np.abs(maxval):
        vortmax = maxval
        vortmin = -maxval
    else:
```

```
        vortmin = minval
        vortmax = -minval

V2[V2 > vortmax] = vortmax
V2[V2 < vortmin] = vortmin

plt.imshow(V2,cmap='jet',vmin=vortmin,vmax=vortmax)

cvals = np.array([-4,-2,-1,-0.5,-0.25,-0.155])
plt.contour(V2,cvals*vortmax/5,colors='k',linestyles='dashed',linewidt
hs=1)
plt.contour(V2,np.flip(-cvals)*vortmax/5,colors='k',linestyles='solid'
,linewidths=0.4)

plt.scatter(22,2,5000,color='k') # draw cylinder


plt.show()
```

```
---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
<ipython-input-48-8fbb9260828d> in <cell line: 25>()
     23
     24 cvals = np.array([-4,-2,-1,-0.5,-0.25,-0.155])
---> 25
plt.contour(V2,cvals*vortmax/5,colors='k',linestyles='dashed',linewidt
hs=1)
     26
plt.contour(V2,np.flip(-cvals)*vortmax/5,colors='k',linestyles='solid'
,linewidths=0.4)
     27

/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py in
contour(data, *args, **kwargs)
   2525 @_copy_docstring_and_deprecators(Axes.contour)
   2526 def contour(*args, data=None, **kwargs):
-> 2527     __ret = gca().contour(
   2528         *args, **({"data": data} if data is not None else {}),
   2529         **kwargs)

/usr/local/lib/python3.10/dist-packages/matplotlib/__init__.py in
inner(ax, data, *args, **kwargs)
   1440     def inner(ax, *args, data=None, **kwargs):
   1441         if data is None:
-> 1442             return func(ax, *map(sanitize_sequence, args),
**kwargs)
   1443
   1444         bound = new_sig.bind(ax, *args, **kwargs)
```

```
/usr/local/lib/python3.10/dist-packages/matplotlib/axes/_axes.py in
contour(self, *args, **kwargs)
   6449         """
   6450         kwargs['filled'] = False
-> 6451         contours = mcontour.QuadContourSet(self, *args,
**kwargs)
   6452         self._request_autoscale_view()
   6453         return contours

/usr/local/lib/python3.10/dist-packages/matplotlib/contour.py in
__init__(self, ax, levels, filled, linewidths, linestyles, hatches,
alpha, origin, extent, cmap, colors, norm, vmin, vmax, extend,
antialiased, nchunk, locator, transform, negative_linestyles, *args,
**kwargs)
    767                 mpl.rcParams['contour.negative_linestyle']
    768
--> 769         kwargs = self._process_args(*args, **kwargs)
    770         self._process_levels()
    771

/usr/local/lib/python3.10/dist-packages/matplotlib/contour.py in
_process_args(self, corner_mask, algorithm, *args, **kwargs)
   1409             self._corner_mask = corner_mask
   1410
-> 1411             x, y, z = self._contour_args(args, kwargs)
   1412
   1413             contour_generator = contourpy.contour_generator(

/usr/local/lib/python3.10/dist-packages/matplotlib/contour.py in
_contour_args(self, args, kwargs)
   1458             _api.warn_external('Log scale: values of z <= 0
have been masked')
   1459             self.zmin = float(z.min())
-> 1460         self._process_contour_level_args(args, z.dtype)
   1461         return (x, y, z)
   1462

/usr/local/lib/python3.10/dist-packages/matplotlib/contour.py in
_process_contour_level_args(self, args, z_dtype)
   1141             raise ValueError("Filled contours require at least
2 levels.")
   1142         if len(self.levels) > 1 and
np.min(np.diff(self.levels)) <= 0.0:
-> 1143             raise ValueError("Contour levels must be
increasing")
   1144
   1145     def _process_levels(self):

ValueError: Contour levels must be increasing
```

```
Error in callback <function _draw_all_if_interactive at
0x7f1b0d3ea7a0> (for post_execute):

---------------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
/usr/local/lib/python3.10/dist-packages/matplotlib/pyplot.py in
_draw_all_if_interactive()
    118 def _draw_all_if_interactive():
    119     if matplotlib.is_interactive():
--> 120         draw_all()
    121
    122

/usr/local/lib/python3.10/dist-packages/matplotlib/_pylab_helpers.py
in draw_all(cls, force)
    130         for manager in cls.get_all_fig_managers():
    131             if force or manager.canvas.figure.stale:
--> 132                 manager.canvas.draw_idle()
    133
    134

/usr/local/lib/python3.10/dist-packages/matplotlib/backend_bases.py in
draw_idle(self, *args, **kwargs)
   2080         if not self._is_idle_drawing:
   2081             with self._idle_draw_cntx():
-> 2082                 self.draw(*args, **kwargs)
   2083
   2084     @property

/usr/local/lib/python3.10/dist-packages/matplotlib/backends/backend_ag
g.py in draw(self)
    398              (self.toolbar._wait_cursor_for_draw_cm() if
self.toolbar
    399               else nullcontext()):
--> 400             self.figure.draw(self.renderer)
    401             # A GUI class may be need to update a window using
this draw, so
    402             # don't forget to call the superclass.

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
draw_wrapper(artist, renderer, *args, **kwargs)
     93     @wraps(draw)
     94     def draw_wrapper(artist, renderer, *args, **kwargs):
---> 95         result = draw(artist, renderer, *args, **kwargs)
     96         if renderer._rasterizing:
     97             renderer.stop_rasterizing()

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
```

```
draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
     73         finally:
     74             if artist.get_agg_filter() is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/figure.py in
draw(self, renderer)
   3138
   3139             self.patch.draw(renderer)
-> 3140             mimage._draw_list_compositing_images(
   3141                 renderer, self, artists,
self.suppressComposite)
   3142

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists,
suppress_composite)
    129     if not_composite or not has_images:
    130         for a in artists:
--> 131             a.draw(renderer)
    132     else:
    133         # Composite any adjacent images together

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
     73         finally:
     74             if artist.get_agg_filter() is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/axes/_base.py in
draw(self, renderer)
   3062             _draw_rasterized(self.figure, artists_rasterized,
renderer)
   3063
-> 3064         mimage._draw_list_compositing_images(
   3065             renderer, self, artists,
self.figure.suppressComposite)
   3066

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists,
suppress_composite)
    129     if not_composite or not has_images:
    130         for a in artists:
--> 131             a.draw(renderer)
    132     else:
```

```
    133         # Composite any adjacent images together

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
     73         finally:
     74             if artist.get_agg_filter() is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
draw(self, renderer, *args, **kwargs)
    639                 renderer.draw_image(gc, l, b, im, trans)
    640         else:
--> 641             im, l, b, trans = self.make_image(
    642                 renderer, renderer.get_image_magnification())
    643             if im is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
make_image(self, renderer, magnification, unsampled)
    947         clip = ((self.get_clip_box() or self.axes.bbox) if
self.get_clip_on()
    948                 else self.figure.bbox)
--> 949         return self._make_image(self._A, bbox,
transformed_bbox, clip,
    950                                 magnification,
unsampled=unsampled)
    951

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_make_image(self, A, in_bbox, out_bbox, clip_bbox, magnification,
unsampled, round_to_pixel_border)
    543                 with self.norm.callbacks.blocked(), \
    544                      cbook._setattr_cm(self.norm, vmin=s_vmin,
vmax=s_vmax):
--> 545                     output = self.norm(resampled_masked)
    546             else:
    547                 if A.ndim == 2:  # _interpolation_stage ==
'rgba'

/usr/local/lib/python3.10/dist-packages/matplotlib/colors.py in
__call__(self, value, clip)
   1344             result.fill(0)  # Or should it be all masked?  Or
0.5?
   1345         elif vmin > vmax:
-> 1346             raise ValueError("minvalue must be less than or
equal to maxvalue")
   1347         else:
   1348             if clip:
```

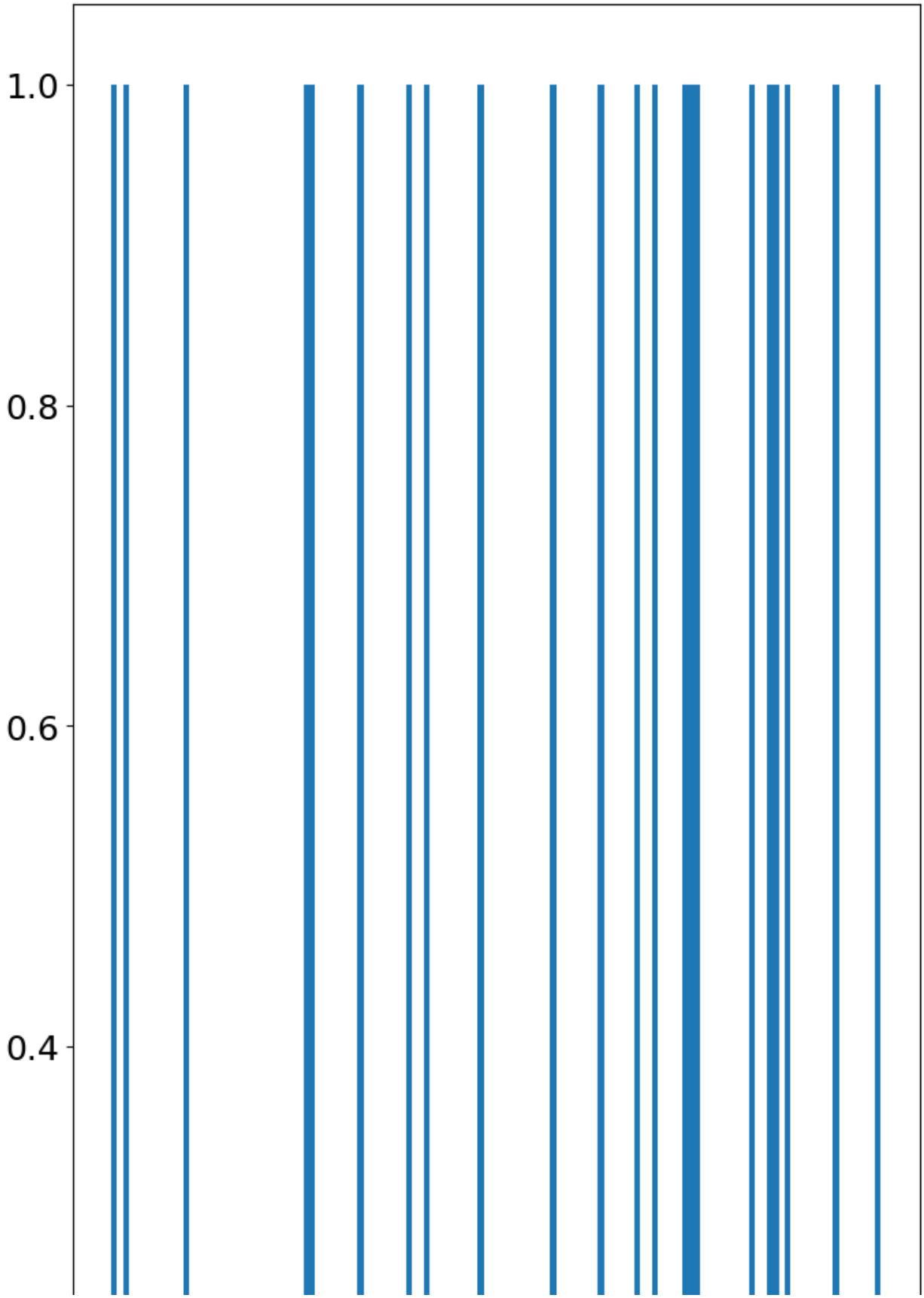
```
ValueError: minvalue must be less than or equal to maxvalue

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/usr/local/lib/python3.10/dist-packages/IPython/core/formatters.py in __call__(self, obj)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)

/usr/local/lib/python3.10/dist-packages/IPython/core/pylabtools.py in print_figure(fig, fmt, bbox_inches, base64, **kwargs)
    149         FigureCanvasBase(fig)
    150
--> 151     fig.canvas.print_figure(bytes_io, **kw)
    152     data = bytes_io.getvalue()
    153     if fmt == 'svg':

/usr/local/lib/python3.10/dist-packages/matplotlib/backend_bases.py in print_figure(self, filename, dpi, facecolor, edgecolor, orientation, format, bbox_inches, pad_inches, bbox_extra_artists, backend, **kwargs)
   2340                 )
   2341                 with getattr(renderer, "_draw_disabled", nullcontext)():
-> 2342                     self.figure.draw(renderer)
   2343
   2344             if bbox_inches:

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in draw_wrapper(artist, renderer, *args, **kwargs)
     93     @wraps(draw)
     94     def draw_wrapper(artist, renderer, *args, **kwargs):
---> 95         result = draw(artist, renderer, *args, **kwargs)
     96         if renderer._rasterizing:
     97             renderer.stop_rasterizing()

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
     73         finally:
     74             if artist.get_agg_filter() is not None:
```

```
/usr/local/lib/python3.10/dist-packages/matplotlib/figure.py in
draw(self, renderer)
   3138
   3139             self.patch.draw(renderer)
-> 3140             mimage._draw_list_compositing_images(
   3141                 renderer, self, artists,
self.suppressComposite)
   3142

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists,
suppress_composite)
    129     if not_composite or not has_images:
    130         for a in artists:
--> 131             a.draw(renderer)
    132     else:
    133         # Composite any adjacent images together

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
     73         finally:
     74             if artist.get_agg_filter() is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/axes/_base.py in
draw(self, renderer)
   3062             _draw_rasterized(self.figure, artists_rasterized,
renderer)
   3063
-> 3064         mimage._draw_list_compositing_images(
   3065             renderer, self, artists,
self.figure.suppressComposite)
   3066

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_draw_list_compositing_images(renderer, parent, artists,
suppress_composite)
    129     if not_composite or not has_images:
    130         for a in artists:
--> 131             a.draw(renderer)
    132     else:
    133         # Composite any adjacent images together

/usr/local/lib/python3.10/dist-packages/matplotlib/artist.py in
draw_wrapper(artist, renderer)
     70                 renderer.start_filter()
     71
---> 72             return draw(artist, renderer)
```

```
     73         finally:
     74             if artist.get_agg_filter() is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
draw(self, renderer, *args, **kwargs)
    639                 renderer.draw_image(gc, l, b, im, trans)
    640         else:
--> 641             im, l, b, trans = self.make_image(
    642                 renderer, renderer.get_image_magnification())
    643             if im is not None:

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
make_image(self, renderer, magnification, unsampled)
    947         clip = ((self.get_clip_box() or self.axes.bbox) if
self.get_clip_on()
    948                 else self.figure.bbox)
--> 949         return self._make_image(self._A, bbox,
transformed_bbox, clip,
    950                                 magnification,
unsampled=unsampled)
    951

/usr/local/lib/python3.10/dist-packages/matplotlib/image.py in
_make_image(self, A, in_bbox, out_bbox, clip_bbox, magnification,
unsampled, round_to_pixel_border)
    543                 with self.norm.callbacks.blocked(), \
    544                      cbook._setattr_cm(self.norm, vmin=s_vmin,
vmax=s_vmax):
--> 545                     output = self.norm(resampled_masked)
    546             else:
    547                 if A.ndim == 2:  # _interpolation_stage ==
'rgba'

/usr/local/lib/python3.10/dist-packages/matplotlib/colors.py in
__call__(self, value, clip)
   1344             result.fill(0)  # Or should it be all masked?  Or
0.5?
   1345         elif vmin > vmax:
-> 1346             raise ValueError("minvalue must be less than or
equal to maxvalue")
   1347         else:
   1348             if clip:

ValueError: minvalue must be less than or equal to maxvalue

<Figure size 800x1600 with 1 Axes>
```

```
V2 = np.real(np.reshape(Phi[:,1],(2,11)))
```

```
plt.hist(V2.reshape(-1),128)
plt.show()
```


1.0
0.8
0.6
0.4