```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
from pylab import gca


def frame_tick(frame_width = 2, tick_width = 1.5):
    # reformat plotting code
    ax = gca()
    for axis in ['top','bottom','left','right']:
        ax.spines[axis].set_linewidth(frame_width)
    plt.tick_params(direction = 'in',
                    width = tick_width)

def set_axes_equal(ax):
    '''Make axes of 3D plot have equal scale so that spheres appear as spheres,
    cubes as cubes, etc..  This is one possible solution to Matplotlib's
    ax.set_aspect('equal') and ax.axis('equal') not working for 3D.

    Input
      ax: a matplotlib axis, e.g., as output from plt.gca().
    '''

    x_limits = ax.get_xlim3d()
    y_limits = ax.get_ylim3d()
    z_limits = ax.get_zlim3d()

    x_range = abs(x_limits[1] - x_limits[0])
    x_middle = np.mean(x_limits)
    y_range = abs(y_limits[1] - y_limits[0])
    y_middle = np.mean(y_limits)
    z_range = abs(z_limits[1] - z_limits[0])
    z_middle = np.mean(z_limits)

    # The plot bounding box is a sphere in the sense of the infinity
    # norm, hence I call half the max range the plot radius.
    plot_radius = 0.5*max([x_range, y_range, z_range])

    ax.set_xlim3d([x_middle - plot_radius, x_middle + plot_radius])
    ax.set_ylim3d([y_middle - plot_radius, y_middle + plot_radius])
    ax.set_zlim3d([z_middle - plot_radius, z_middle + plot_radius])
def animate_data(image):
    for i in range(image.shape[2]):
        plt.imshow(image[:,:,i])
        plt.pause(0.1)
        plt.clf()

def improve_pairplot(g, replacements):
    g.fig.set_dpi(50)#)#.figsize

#     g._legend.remove()

#     g._legend.set_bbox_to_anchor((0.50,-0.05))
```

```
#     g._legend.set_fontsize = 20
#     g._legend.set_ncol = 2#((fontsize=20, bbox_to_anchor=(0.50,-0.05), ncol=2).fig
#     g._legend.get_title().set_fontsize('20')
    for idx,i in enumerate(g.axes[0]):
        for idx_j,j in enumerate(g.axes):
            g.axes[idx_j][idx].spines['left'].set_linewidth(2)
            g.axes[idx_j][idx].spines['bottom'].set_linewidth(2)
            g.axes[idx_j][idx].tick_params(direction = 'in', width = 1.5)
            xlabel = g.axes[idx_j][idx].get_xlabel()
            ylabel = g.axes[idx_j][idx].get_ylabel()
            if xlabel in replacements.keys():
                g.axes[idx_j][idx].set_xlabel(replacements[xlabel], fontsize = 18)
            if ylabel in replacements.keys():
                g.axes[idx_j][idx].set_ylabel(replacements[ylabel], fontsize = 18)
    return g


```