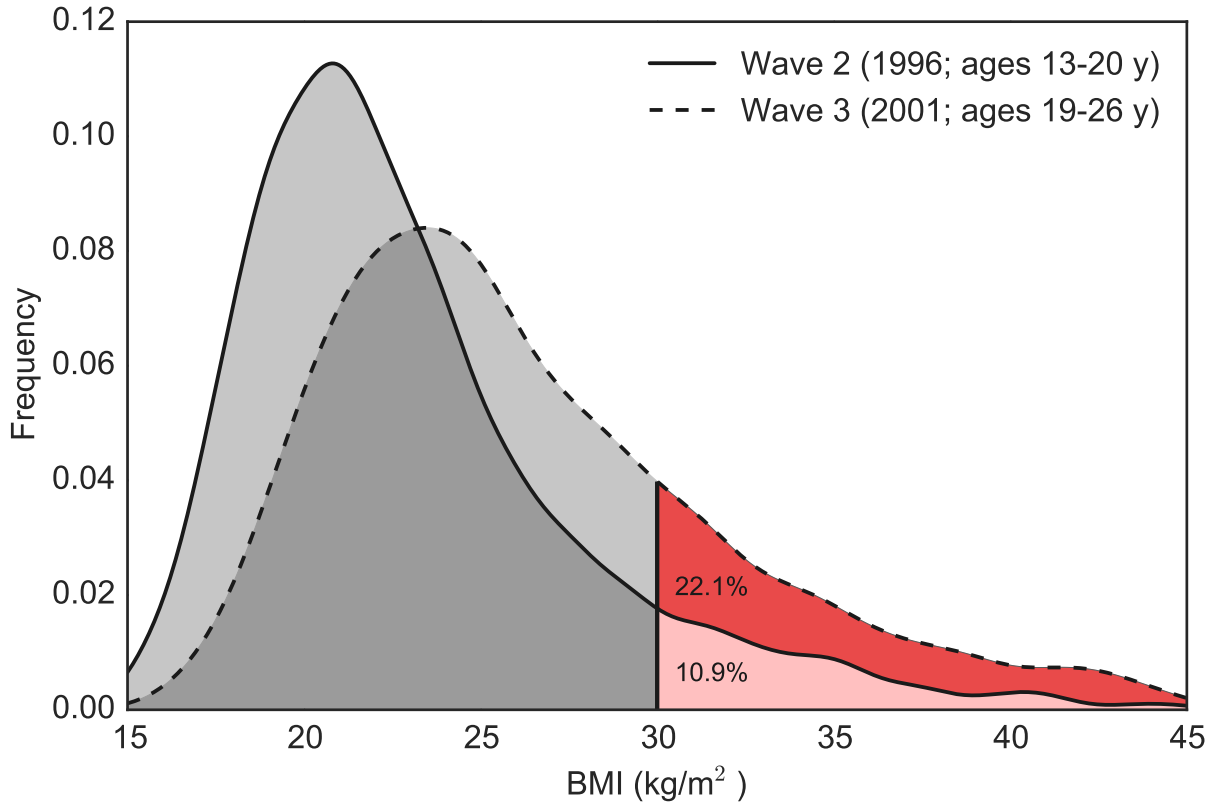

Figure 1: Distribution frequency of BMI from the National Longitudinal Study of Adolescent Health (Add Health) waves II (1996; ages 13—20y) and III (2001; ages 19—26y). The transition between adolescence and young adulthood is a high risk period for weight gain. 10.9% of young adults in wave II had a BMI > 30 on the basis of adult cutoffs, a number which increased to 22.1% in wave III.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt


cols2 = {'Gender': 'BIO_SEX2',
         'Age': 'CALCAGE2',
         'Weight': 'H2WS16W',
         'Height (ft)': 'H2WS16HF',
         'Height (in)': 'H2WS16HI'}

cols3 = {'Gender': 'BIO_SEX3',
         'Age': 'CALCAGE3',
         'Weight': 'H3WGT',
         'Height (ft)': 'H3HGT_F',
         'Height (in)': 'H3HGT_I'}


def calc_bmi(df, d=2):
    if d == 2:
        cols = cols2
    elif d == 3:
        cols = cols3

    df = df[list(cols.values())].convert_objects(convert_numeric=True)
    df['weight_lbs'] = df[cols['Weight']]
    df['height_in'] = df[cols['Height (in)']]
    df['height_in'] += 12 * df[cols['Height (ft)']]
    df['bmi'] = df.apply(lambda x: (x['weight_lbs']/x['height_in']**2) * 703, axis=1)
    df = df[1 < df.bmi]
    df = df[df.bmi < 100].dropna()

    return df

def load_data():
    df2 = pd.read_csv('Data_Wave02.tsv', sep='\t')
    df3 = pd.read_csv('Data_Wave03.tsv', sep='\t')

    df2 = calc_bmi(df2)
    df3 = calc_bmi(df3, d=3)
    return df2, df3

def plot(df2, df3):
    sns.set(style="white", color_codes=True)
    f, ax = sns.plt.subplots()
    sns.kdeplot(df2.bmi, ax=ax, shade=True, color='k', gridsize=10000, clip=(15, 45))
    sns.kdeplot(df3.bmi, ax=ax, shade=True, color='k', ls='dashed', gridsize=10000, clip
    =(15, 45))

    plt.legend(['Wave 2 (1996; ages 13-20 y)', 'Wave 3 (2001; ages 19-26 y)'], fontsize
    =15)
    plt.xlim(15, 45)
```

```
52
53      ax.annotate('10.9%', xy=(25, .02), xytext=(30.5, .005), color='k')
54      ax.annotate('22.1%', xy=(25, .04), xytext=(30.5, .02), color='k')
55
56      y1 = ax.lines[0].get_ydata()
57      x1 = ax.lines[0].get_xdata()
58      x_mask1 = np.ma.masked_less_equal(x1, 30).mask
59      y_masked1 = np.ma.masked_array(y1, x_mask1)
60
61      y2 = ax.lines[1].get_ydata()
62      x2 = ax.lines[1].get_xdata()
63      x_mask2 = np.ma.masked_less_equal(x2, 30).mask
64      y_masked2 = np.ma.masked_array(y2, x_mask2)
65
66      ax.fill_between(x2, np.zeros_like(y2), y_masked2, facecolor='red', interpolate=True,
        alpha=0.5)
67      ax.fill_between(x1, np.zeros_like(y1), y_masked1, facecolor='white', interpolate=True
        )
68      ax.fill_between(x2, np.zeros_like(y2), y_masked2, facecolor='red', interpolate=True,
        alpha=0.25)
69
70      plt.vlines(x=30, ymin=0, ymax=0.0398, color='k', linewidth=2, alpha=1)#, ls='dashed')
71
72      plt.xticks(size=15)
73      plt.yticks(size=15)
74
75      plt.ylabel('Frequency', fontsize=15)
76      plt.xlabel('BMI (kg/m$^2$)', fontsize=15)
77      plt.tight_layout()
78      plt.show()
79
80
81  if __name__ == "__main__":
82      df2, df3 = load_data()
83      plot(df2, df3)
```