In [ ]:

7- AKIM ALAN GRAFİĞİ- STREAMGRAPH

In [ ]:

Tanım:

Stream Graph akım grafiği yığım alan grafiği türüdür. Tıpkı çizgi grafiği ve alan grafiğinde olduğu gibi bir nümerik değerin değişimini gösterir. Birden fazla grubun verisi farklı renkler kullanılarak gösterilebilir.

Stacked Area Graph dan farklı olarak bunda köşeler veya kenarlar yuvarlatılmış ve daha güzel bir görsel sağlanmıştır. Merkezi bir eksen etrafında çizim yapıldığından daha akıcı bir görünüm sağlanmış olur.

In [ ]:

Ne için Kullanılır:

Akım grafiği bir bütünün kısımlarını göstermek için kullanımı güzel bir grafik türüdür. Fakat üst üste yığım yağıldığından tek tek her bir değişkenin değişimini göstermek için kötü bir yöntemdir. bu tür bir sorun için çzigi veya alan grafiği kullanmak daha dordu olur.

Çeşitleri:

Akım grafiklerinde en alta bir X ekseni kulanıla bileceği gibi eksen ortada olacak biçimde kullanımı da mümkündür. Ayrıca sürekli bir bütünün içerisindeki oranı göstermek adına tüm alan içierisindeki oran değişimi de gösterilebilir.

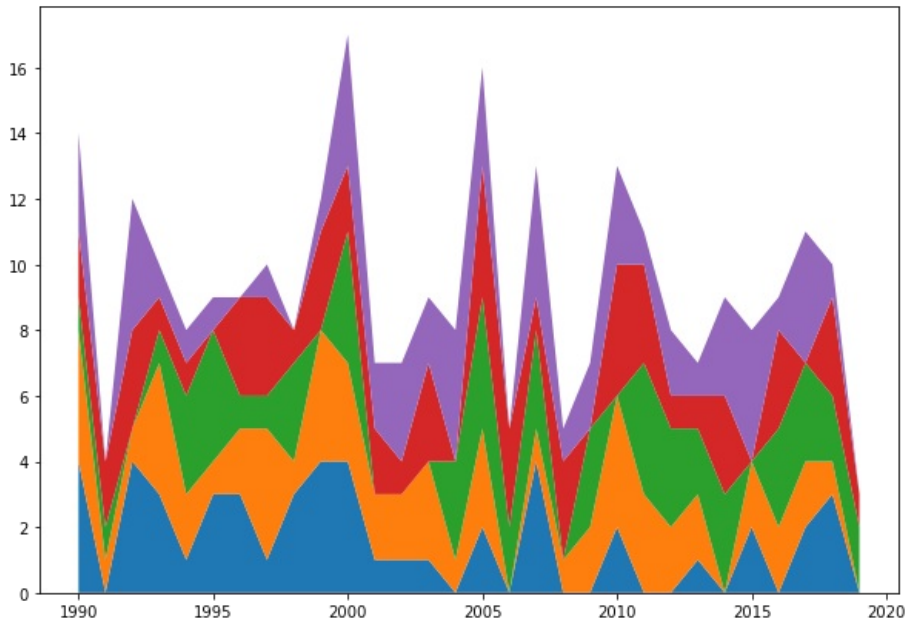
Kullanım Hataları:

Bu garfik türünde da yığım grafiğinde olduğu gibi her grubun verisinin okunması sorunludur. Ayrıca patern bariz değil ise çok küçük değişimlerde okunması biraz zor bir grafiktir.

In [ ]:

In [2]:

```
# Libraries
import matplotlib.pyplot as plt
import numpy as np
from scipy import stats

x = np.arange(1990, 2020) # (N,) array-like

y = [np.random.randint(0, 5, size=30) for _ in range(5)] # (M, N) array-like

fig, ax = plt.subplots(figsize=(10, 7))
ax.stackplot(x, y);
```

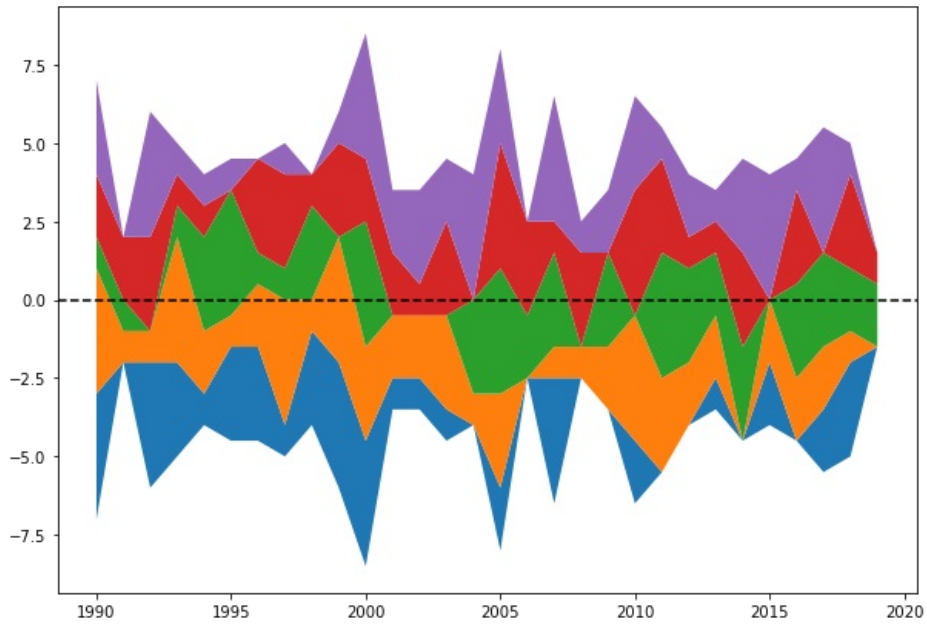


In [ ]:

```

```

In [3]:

```
fig, ax = plt.subplots(figsize=(10, 7))
ax.stackplot(x, y, baseline="sym")
ax.axhline(0, color="black", ls="--");
```



In [ ]:

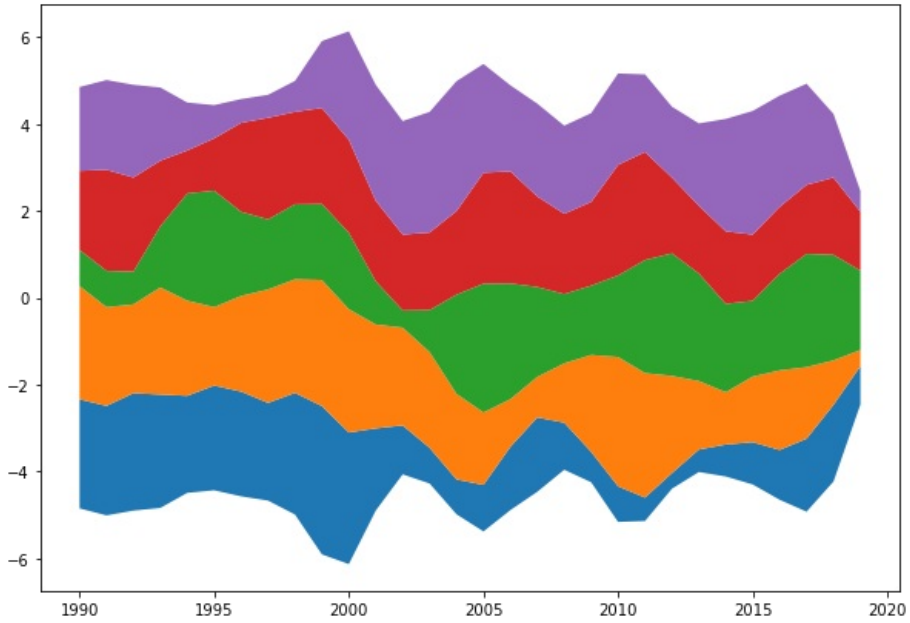
```

```

In [4]:

```
def gaussian_smooth(x, y, sd):
    weights = np.array([stats.norm.pdf(x, m, sd) for m in x])
    weights = weights / weights.sum(1)
    return (weights * y).sum(1)
```

In [5]:

```
fig, ax = plt.subplots(figsize=(10, 7))
y_smoothed = [gaussian_smooth(x, y_, 1) for y_ in y]
ax.stackplot(x, y_smoothed, baseline="sym");
```



In [ ]:

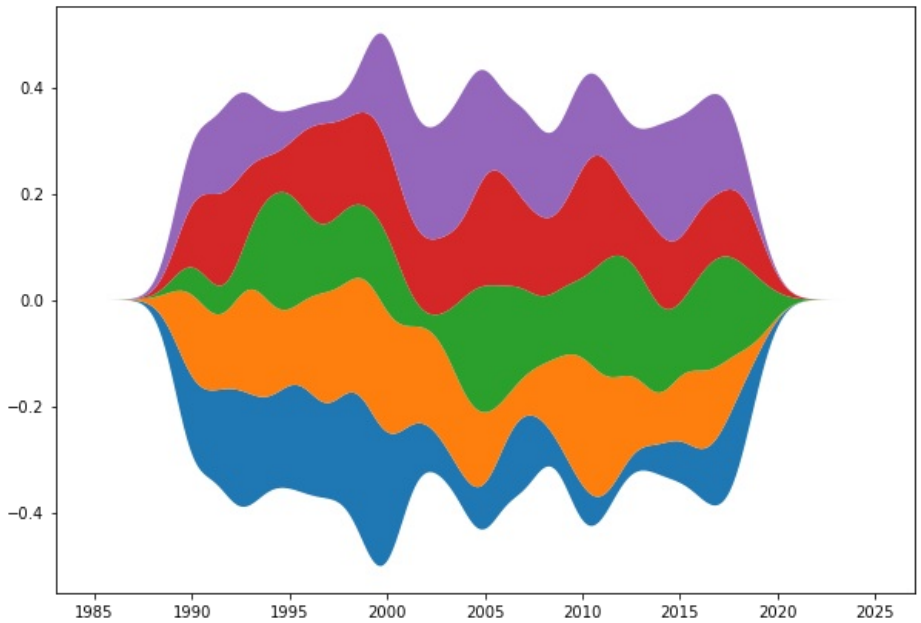
```

```

In [6]:

```
def gaussian_smooth(x, y, grid, sd):
    weights = np.transpose([stats.norm.pdf(grid, m, sd) for m in x])
    weights = weights / weights.sum(0)
    return (weights * y).sum(1)
```

In [7]:

```
fig, ax = plt.subplots(figsize=(10, 7))
grid = np.linspace(1985, 2025, num=500)
y_smoothed = [gaussian_smooth(x, y_, grid, 1) for y_ in y]
ax.stackplot(grid, y_smoothed, baseline="sym");
```

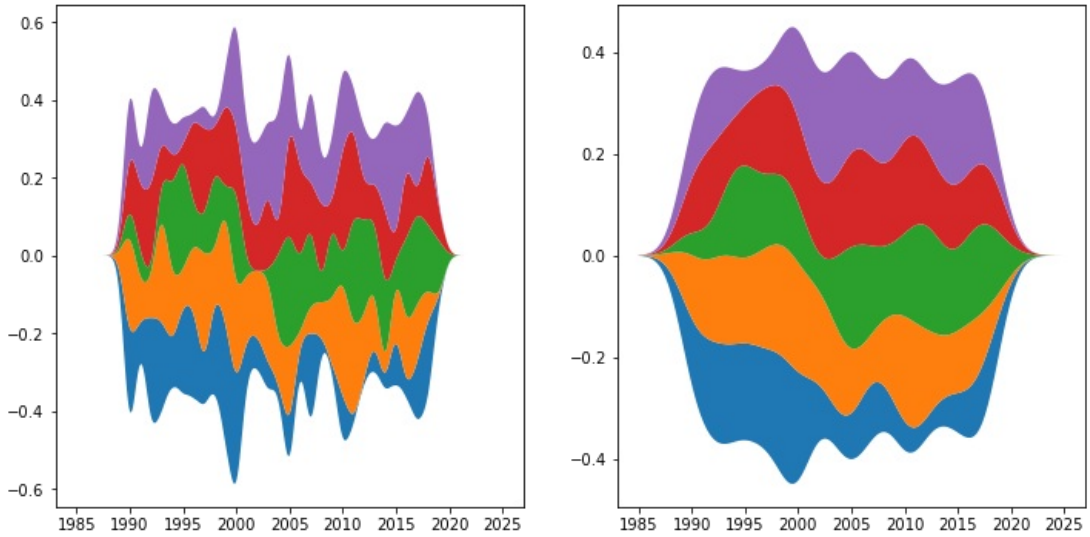


In [ ]:

```
```

In [8]:

```
fig, ax = plt.subplots(1, 2, figsize=(12, 6))
# sd of 0.6
y_smoothed_1 = [gaussian_smooth(x, y_, grid, 0.6) for y_ in y]
# sd of 1.5
y_smoothed_2 = [gaussian_smooth(x, y_, grid, 1.5) for y_ in y]

ax[0].stackplot(grid, y_smoothed_1, baseline="sym")
ax[1].stackplot(grid, y_smoothed_2, baseline="sym");
```

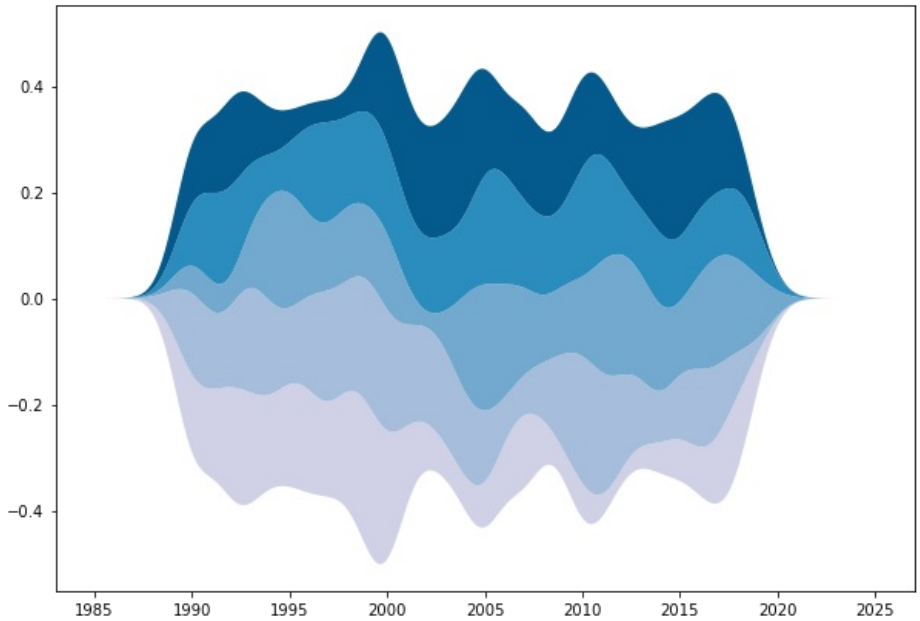


In [ ]:

```
```

In [9]:

```
COLORS = ["#D0D1E6", "#A6BDDB", "#74A9CF", "#2B8CBE", "#045A8D"]
fig, ax = plt.subplots(figsize=(10, 7))
# Colors in the `COLORS` list are assigned to individual areas from bottom to top.
ax.stackplot(grid, y_smoothed, colors=COLORS, baseline="sym");
```



In [ ]:

```
```