Import relevant packages here.

In [7]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

Load the data and verify it is loaded correctly.

- Print it (head, tail, or specific rows, choose a sensible number of rows).
- Compare it to the source file.

In [6]:
```
file = 'C:/Users/alvor/OneDrive/Desktop/Masters Work/Winter 2024/TIL 6022/Week 3/cf_data.csv'
data = pd.read_csv(file)
data.head(10)
```

Out[6]:

	dv	s	a
0	-0.743240	53.5427	1.242570
1	-0.557230	53.6120	1.777920
2	-0.454769	53.6541	0.544107
3	-0.525396	53.7030	-0.294755
4	-0.601285	53.7592	-0.290961
5	-0.682448	53.8232	-0.283414
6	-0.768859	53.8957	-0.271604
7	-0.860452	53.9770	-0.133532
8	-0.832777	54.0678	0.243356
9	-0.576125	54.1436	0.406759

In the ensuing, you will use `numpy`.

Let's create a grid for the values to plot. But first create **two arrays named `dv` and `s`** using `numpy.linspace` that hold the grid values at the relevant indices in their respective dimension of the grid.

Create a **grid named `a`** with zeros using `numpy.zeros` in to which calculated acceleration values can be stored.

Let the grid span:

- Speed difference `dv` [m/s]
  - From -10 till 10
  - With 41 evenly spaced values
- Headway `s` [m]
  - From 0 till 200
  - With 21 evenly spaced values

In [22]:
```
dv = np.linspace(-10, 10, num=41)
s = np.linspace(0, 200, 21)
a = np.zeros([21, 41])
```

Create from the imported data 3 separate `numpy` arrays for each column `dv`, `s` and `a`. (We do this for speed reasons later.)

- Make sure to name them differently from the arrays that belong to the grid as above.
- You can access the data of each column in a `DataFrame` using `data.xxx` where `xxx` is the column name (not as a string).
- Use the method `to_numpy()` to convert a column to a `numpy` array.

In [24]:
```
DV = data.dv.to_numpy()
S = data.s.to_numpy()
A = data.a.to_numpy()

DV, S, A
```

Out[24]:
```
(array([-0.74324 , -0.55723 , -0.454769, ...,  5.13764 ,  5.15348 ,
         5.25868 ]),
 array([ 53.5427,  53.612 ,  53.6541, ..., 115.118 , 114.599 , 113.112 ]),
 array([ 1.24257 ,  1.77792 ,  0.544107, ...,  0.232283,  0.262078,
        -0.61244 ]))
```

Create an algorithm that calculates all the acceleration values and stores them in the grid. The algorithm is described visually in the last part of the lecture. At each grid point, it calculates a weighted mean of all measurements. The weights are given by an exponential function, based on the 'distance' between the grid point, and the measurement values of `dv` and `s`. To get you started, how many `for`-loops do you need?

For this you will need `math`.
Use an *upsilon* of 1.5m/s and a *sigma* of 30m.

Warning: This calculation may take some time. So:

- Print a line for each iteration of the outer-most `for`-loop that shows you the progress.
- Test you code by running it only on the first 50 measurements of the data.

In [30]:
```
import math
upsilon = 1.5 # m/s
sigma = 30 # meters
rowCounter = 0

for i_dv in range(len(dv)):

    for i_s in range(len(s)):
        sum_weighted_acc = 0
        sum_weight = 0

        for i_data in range(len(A)):
            weight = math.exp(-abs(DV[i_data] - dv[i_dv]) / upsilon - abs(S[i_data] - s[i_s] / sigma))
            sum_weighted_acc += (weight * A[i_data])
            sum_weight += weight

        a[i_s, i_dv] = sum_weighted_acc / sum_weight

    print(f'Row number {rowCounter} is completed')
    rowCounter += 1
```

```
Row number 0 is completed
Row number 1 is completed
Row number 2 is completed
Row number 3 is completed
Row number 4 is completed
Row number 5 is completed
Row number 6 is completed
Row number 7 is completed
Row number 8 is completed
Row number 9 is completed
Row number 10 is completed
Row number 11 is completed
Row number 12 is completed
Row number 13 is completed
Row number 14 is completed
Row number 15 is completed
Row number 16 is completed
Row number 17 is completed
Row number 18 is completed
Row number 19 is completed
Row number 20 is completed
Row number 21 is completed
Row number 22 is completed
Row number 23 is completed
Row number 24 is completed
Row number 25 is completed
Row number 26 is completed
Row number 27 is completed
Row number 28 is completed
Row number 29 is completed
Row number 30 is completed
Row number 31 is completed
Row number 32 is completed
Row number 33 is completed
Row number 34 is completed
Row number 35 is completed
Row number 36 is completed
Row number 37 is completed
Row number 38 is completed
Row number 39 is completed
Row number 40 is completed
```

The following code will plot the data for you. Does it make sense when considering:

- Negative (slower than leader) and positive (faster than leader) speed differences?
- Small and large headways?

In [31]:
```
X, Y = np.meshgrid(dv, s)
axs = plt.axes()
p = axs.pcolor(X, Y, a, shading='nearest')
axs.set_title('Acceleration [m/s/s]')
axs.set_xlabel('Speed difference [m/s]')
axs.set_ylabel('Headway [m]')
axs.figure.colorbar(p);
axs.figure.set_size_inches(10, 7)
```