HW03

Jun Zhu, SA20007073

March 28, 2021

1

r_v is used for geophone, while r_p is used for hydrophone. Geophone records the velocity of the particle vibration.

The design of effective hydrophones must take into account the acoustic resistance of water, which is 3750 times that of air; hence the pressure exerted by a wave of the same intensity in air is increased by a factor of 3750 in water. If any problem recording the pressure, maybe removing the acoustic response of different mediums is a feasible solution.

2 Multiplicity Analysis

2.1 Some figures

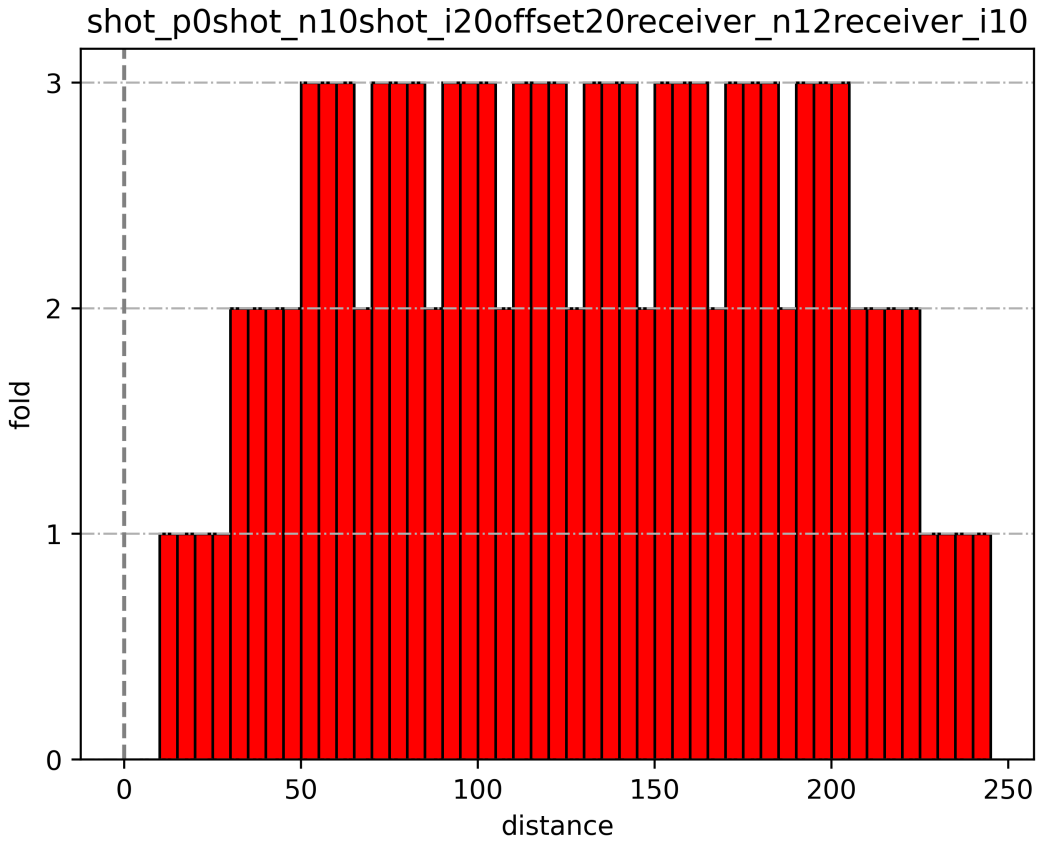

Figure 1: shot position: 0, shot number: 10, shot interval: 20, offset: 20, receiver number: 20, receiver interval: 10

Figure 2: except for receiver interval equals to 1, the others are the same with Figure 1

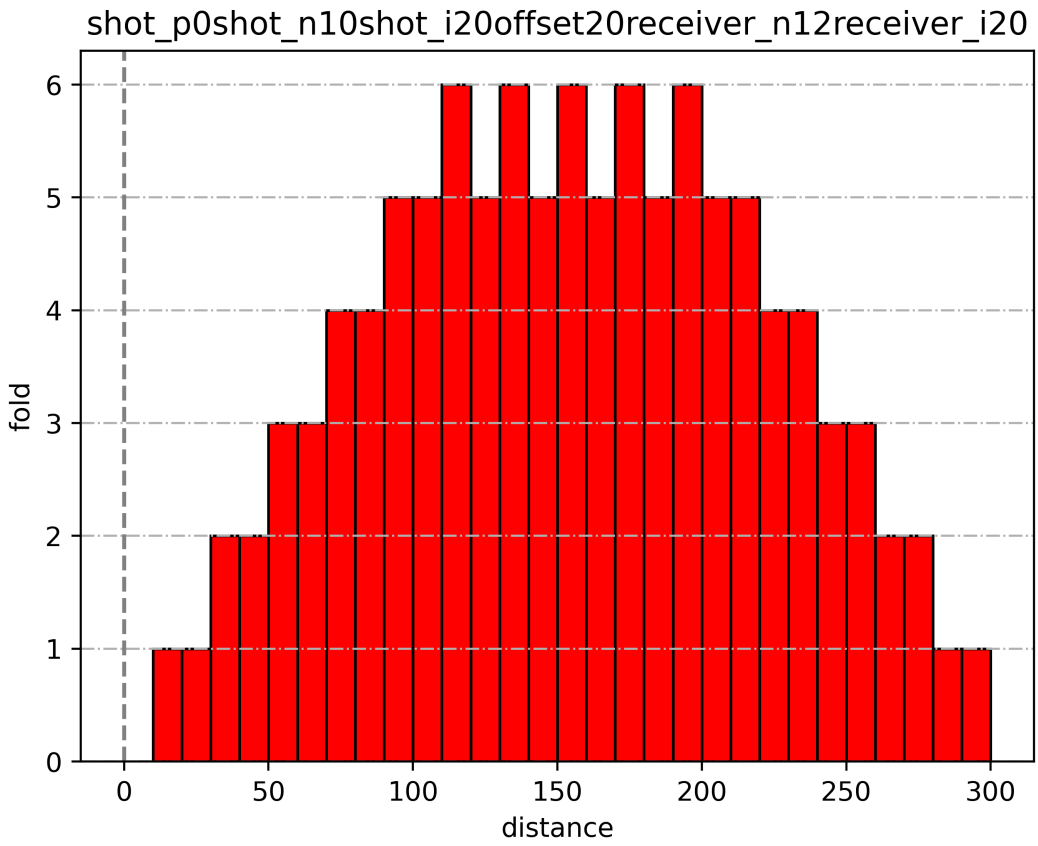

Figure 3: except for receiver interval equals to 20, the others are the same with Figure 1

2.2 Python code (Numpy and Matplotlib are required)

```
# Author: Jun Zhu
# Date:  MAR, 28 2021
# Email: Jun__Zhu@outlook.com

import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import MaxNLocator

def inwindow(x=[4,4.1],y=[2,4]):
   """check if the box is in the window
      notes:
```

```
        if True, return 1
        if False, return 0
    """
    if (x[1]>y[0]) and (x[0]<y[1]):
        tick = 1
    else:
        tick =0
    return tick

def fold(
        shot_p=0,
        shot_n=2,
        shot_i=1,
        offset=1,
        receiver_n=2,
        receiver_i=1):
    """calculate multiplicity
    """
    first_CMPs = []
    CMP_lists = []
    CMPs = []
    shot_pn = shot_p
    for n in range(shot_n):
        first_CMP = shot_pn + 1/2*offset
        first_CMPs.append(first_CMP)
        CMP_n = first_CMP + np.arange(receiver_n)*receiver_i/2
        CMP_lists.append(CMP_n)
        CMPs += list(CMP_n)
        shot_pn += shot_i
    CMPs = list(set(CMPs))
    CMPs.sort()
    boxes = [[x, CMPs[i+1]] for i,x in enumerate(CMPs[:-1])]
    boxes = [[shot_p, first_CMPs[0]]] + boxes
    windows = [[x[0],x[-1]] for x in CMP_lists]
    multiplicity = [sum([inwindow(box,window) for window in windows]) for
        box in boxes]
    n = multiplicity
    lastbox_width = boxes[-1][-1]-boxes[-1][0]
    bins = [x[0] for x in boxes]
    width = np.append(np.diff(bins), lastbox_width)
    ax = plt.figure().gca()
    ax.bar(bins, n, width=width, align='edge', edgecolor='k', color='r')
    ax.axvline(x=shot_p, ls='--', c='gray')
    ax.set_xlabel('distance')
    ax.set_ylabel('fold')
    parameters =
        "shot_p"+str(shot_p)+"shot_n"+str(shot_n)+"shot_i"+str(shot_i)+"offset"+str(offset)+"receiver_n
    ax.set_title(parameters)
    ax.yaxis.set_major_locator(MaxNLocator(integer=True))
    ax.grid(linestyle='-.', axis='y')
```

```
    fname = "../image/"+parameters+".png"
    plt.savefig(fname, dpi=500)
    #plt.show()
    pass

if __name__ == "__main__":
    candidate_receiver_i = np.arange(20) + 1
    for receiver_i in candidate_receiver_i:
        fold(
            shot_p=0,
            shot_n=10,
            shot_i=20,
            offset=20,
            receiver_n=12,
            receiver_i=receiver_i)
    pass
```