AnnualMeeting2025_Aquistore_FirstLook_wplots

April 20, 2025

1 Aquistore Data Processing and Figures for 2025 Annual SEP Meeting: Aquistore DAS First Look Report

Author: Thomas Cullison, Stanford University

1.1 Evironment Setup

1.1.1 Auto Import

[1]:

```
%load_ext autoreload
%autoreload 2
```

1.1.2 Verify Python EXE Path

[2]:

```
from sys import executable as mypyexe
print(mypyexe)
```

```
/home/user/mypy312/bin/python
```

[3]:

```
!which {mypyexe}
```

```
/home/user/mypy312/bin/python
```

1.1.3 Common Imports

[4]:

```
# import pynmea2
import folium
import utm
import re

import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import matplotlib.patches as patches

from copy import deepcopy
from time import time
from functools import partial
from IPython.display import display, HTML

from matplotlib.ticker import FixedLocator
from matplotlib.ticker import FormatStrFormatter

from scipy.ndimage import median_filter
from scipy.signal import detrend as sci_detrend

from pathlib import Path as plib_path
from os.path import join as os_p_join

from copy import deepcopy as dcopy
```

1.1.4 Import SeisBear functions

[5]:
```
from seisbear.core import _write_seisbear, _read_segyio, _read_seisbear,␣
 ↪_merge_file_headers
from seisbear.mapping import get_html_folium_map,  _make_marker_dict,␣
 ↪_make_folium_group_from_dict, _make_folium_base_map, _get_latlot_means,␣
 ↪_add_folium_groups_to_map
```

1.1.5 Import Aquistore functions

[6]:
```
from aquistore.core import _read_sp1, _load_aquistore_das_seisbear
```

1.2 Read SP1 file and save to DataFrame

This file has coordinate locations for all boreholes

[7]:
```
sp1_path = "/shared/data/aquistore/Aquistore_4D_Fibre_Locations_NRCAN_8313.Sp1"
```

[8]:
```
sp1_df = _read_sp1(sp1_path)
display(sp1_df)
```

```
     ID    Lat_DMS     Lon_DMS   Latitude   Longitude   Easting   Northing  \
0    F01  49052134N  103042173W  49.089261 -103.072703  640711.5  5439167.5
1    F02  49052359N  103042136W  49.089886 -103.072600  640717.4  5439237.1
2    F03  49052590N  103042102W  49.090528 -103.072506  640722.4  5439308.5
3    F04  49052822N  103042099W  49.091172 -103.072497  640721.2  5439380.3
4    F05  49053054N  103042101W  49.091817 -103.072503  640718.9  5439451.9
..   …        …          …          …          …           …         …
94   F95  49052709N  103045610W  49.090858 -103.082250  640010.0  5439327.3
95   F96  49052475N  103045497W  49.090208 -103.081936  640034.7  5439255.5
96   F97  49052237N  103045603W  49.089547 -103.082231  640015.3  5439181.7
97   F98  49051998N  103045591W  49.088883 -103.082197  640019.4  5439107.9
98   F99  49051771N  103045653W  49.088253 -103.082369  640008.6  5439037.4
```

```
    Elevation
0       568.3
1       567.5
2       566.7
3       564.7
4       566.6
..        …
94      567.6
95      568.6
96      568.1
97      567.2
98      566.6

[99 rows x 8 columns]
```

1.2.1 Show attrs for SP1 DataFrame

[9]:

```
display(sp1_df.attrs['header'])
display(sp1_df.attrs['easting-northing units'])
display(sp1_df.attrs['zone'])
```

```
['H                   : Terraview Surveys Seismic Survey Data',
 'H Project           : Aquistore 4D 2023',
 'H Client            : NRCAN',
 'H Geophysical. Co.  : Echo Seismic',
 'H Area              : Estevan SK',
 'H Survey Co.        : Terraview Surveys',
 'H Method            : RTK GPS --> Post-Processed -->',
 'H Survey Dates      : Oct 2023',
 'H Datum             : NAD83',
 'H Ellip./SMA/rflat  : GRS80  6378206.400000m  294.97869820',
 'H Shift Parameters  : NTV 2.0 Canada',
 'H Shift to WGS84    :',
 'H Rotation to WGS84 :',
 'H System            : UTM   Units    :  Meters',
 'H Geographics       : DMS',
 'H Comment           : Zone 13',
 'H Comment           :',
 'H Comment           : FIBER SP1',
 'H Comment           :']
```

```
'meters'
```

```
13
```

1.3 Read DAS Data Files

1.3.1 Set Path to Data Files

[10]:

```
data_path = '/shared/data/aquistore'
```

1.3.2 Read DAS Data from seisbear Files

[11]:

```
hset_df, dlst = _load_aquistore_das_seisbear(data_path)
```

1.3.3 Show File and Channel (Trace) Headers

[12]:

```
hset_df
```

[12]:

```
      FileIndex  DataIndex  TRACE_SEQUENCE_LINE  TRACE_SEQUENCE_FILE  \
0             0          0                    0                    1
1             0          1                    1                    2
2             0          2                    2                    3
3             0          3                    3                    4
4             0          4                    4                    5
…         …        …                  …                  …
2127        377       2127                 2127                 2128
2128        377       2128                 2128                 2129
2129        377       2129                 2129                 2130
2130        377       2130                 2130                 2131
2131        377       2131                 2131                 2132

      FieldRecord  TraceNumber  EnergySourcePoint  CDP  CDP_TRACE  \
0              21            0                  0    0          1
1              21            1                  0    0          2
2              21            2                  0    0          3
3              21            3                  0    0          4
4              21            4                  0    0          5
…           …          …                …  …        …
2127          414         2127                  0    0       2128
2128          414         2128                  0    0       2129
2129          414         2129                  0    0       2130
2130          414         2130                  0    0       2131
2131          414         2131                  0    0       2132

      TraceIdentificationCode  …   GroupLat    GroupLon  OffsetX  OffsetY  \
0                           1  …  49.093237 -103.077200  -1227.0   1667.1
1                           1  …  49.093258 -103.077169  -1229.2   1664.7
2                           1  …  49.093279 -103.077137  -1231.5   1662.3
3                           1  …  49.093301 -103.077105  -1233.8   1659.8
4                           1  …  49.093322 -103.077072  -1236.1   1657.4
…                       …  …         …         …      …      …
2127                        1  …  49.093521 -103.076772    215.9    369.5
```

```
2128                            1  …  49.093464 -103.076857    222.0    376.0
2129                            1  …  49.093407 -103.076943    228.1    382.5
2130                            1  …  49.093350 -103.077029    234.2    389.0
2131                            1  …  49.093293 -103.077115    240.3    395.5

      MidpointX  MidpointY    OffsetMag  RecGatherID  SrcGatherID  SrcUniqueID
0       -613.50     833.55  2069.964108         1489            0            0
1       -614.60     832.35  2069.337752         1492            0            0
2       -615.75     831.15  2068.775855         1493            0            0
3       -616.90     829.90  2068.138893         1496            0            0
4       -618.05     828.70  2067.587476         1498            0            0
…           …         …          …           …           …           …
2127     107.95     184.75   427.952170         1521          377          376
2128     111.00     188.00   436.646310         1514          377          376
2129     114.05     191.25   445.349144         1508          377          376
2130     117.10     194.50   454.060172         1502          377          376
2131     120.15     197.75   462.778932         1495          377          376

[805896 rows x 105 columns]
```

[13]:
```
hset_df.attrs['EBCDIC_Headers'][0]
```

[13]:
```
[['C 1 CLIENT  CREW NO
',
  'C 2 WELL  LOCATION
',
  'C 3 DATE 16/11/2023 18:31:59 UTC OBSERVER
',
  'C 4 LOCALTIME 16/11/2023 18:31:59 GMT UTC+0
',
  'C 5 INSTRUMENT ONYX  VERSION 12661 SERIAL NO ONYX 392  FIBER NO 1
',
  'C 6 RECORDING FORMAT 5   2132 TRACES/RECORD   0 AUXILIARY TRACES/RECORD
',
  'C 7 SAMPLE INTERVAL 1000us  60000 SAMPLES/TRACE  ACQUIRED SAMPLE INTERVAL
100us ',
  'C 8 OPTICAL: GAUGE  4.79m  SPATIAL  4.79m  PULSE  3.40m STACKING 4
',
  'C 9 MEASUREMENT: UNITS RADIANS  TYPE STRAIN  POLARITY POSITIVE STANDARD
',
  'C10 HELIX RATIO 0  RI 0
',
  'C11
',
  'C12 SOURCE:
',
  'C13
```

```
',
  'C14
',
  'C15
',
  'C16
',
  'C17 UNITS: m
',
  'C18
',
  'C19
',
  'C20 CABLE ACQUISITION:      0.0 ..  10203.1 m (    0 ..  2131)
',
  'C21 CABLE CALIBRATION:      0.0 ..  10203.1 m OD
',
  'C22
',
  'C23
',
  'C24 NUMBER OF SAMPLES
',
  'C25   CASE   NUM_SAMPLES(N)                FILE HEADER             TRACE
HEADER  ',
  'C26                                  TYPE   OFFSET  VALUE    TYPE   OFFSET
VAL ',
  'C27
------------------------------------------------------------------------- ',
  'C28 1    N <= 32,767                 uint16  3221   N        uint16  115
N  ',
  'C29 1    N <= 32,767                 uint32  3507   N        uint32  225
N  ',
  'C30
------------------------------------------------------------------------- ',
  'C31 2    32,767 < N <= 65,535        uint16  3221   N        uint16  115
N  ',
  'C32 2    N <= 32,767                 uint32  3507   N        uint32  225
N  ',
  'C33
------------------------------------------------------------------------- ',
  'C34
',
  'C35
',
  'C36
',
  'C37
',
  'C38
',
  'C39 SEG-Y REV1.0
',
  'C40 END TEXTUAL HEADER
']]
```

1.3.4 Get All Source Coordinates

[14]:
```
src_latlon_df = hset_df.drop_duplicates(subset=['SourceLat',␣
 ↪'SourceLon'])[['SourceLat', 'SourceLon', 'SourceX', 'SourceY']]
src_latlon_df
```

[14]:
```
    SourceLat   SourceLon   SourceX    SourceY
0   49.108503 -103.093424  639145.0  5441268.1
0   49.103322 -103.099314  638729.6  5440681.3
0   49.108496 -103.091481  639286.8  5441270.8
0   49.108512 -103.089493  639431.8  5441276.3
0   49.108499 -103.087523  639575.6  5441278.5
..        …           …         …          …
0   49.095571 -103.072696  640694.2  5439868.8
0   49.095556 -103.077605  640335.9  5439858.1
0   49.095556 -103.074687  640548.9  5439863.5
0   49.096865 -103.074785  640538.1  5440008.8
0   49.096794 -103.073687  640618.4  5440002.9

[375 rows x 4 columns]
```

1.3.5 Get Channel (receiver) Coordinates

[15]:
```
rec_latlon_df = hset_df.drop_duplicates(subset=['GroupLat',␣
 ↪'GroupLon'])[['GroupLat', 'GroupLon', 'GroupX', 'GroupY']]
rec_latlon_df
```

[15]:
```
       GroupLat    GroupLon    GroupX     GroupY
0     49.093237 -103.077200  640372.0  5439601.0
1     49.093258 -103.077169  640374.2  5439603.4
2     49.093279 -103.077137  640376.5  5439605.8
3     49.093301 -103.077105  640378.8  5439608.3
4     49.093322 -103.077072  640381.1  5439610.7
…           …           …         …          …
2125  49.093635 -103.076600  640414.7  5439646.4
2127  49.093521 -103.076772  640402.5  5439633.4
2128  49.093464 -103.076857  640396.4  5439626.9
2130  49.093350 -103.077029  640384.2  5439613.9
```

```
2131  49.093293 -103.077115  640378.1  5439607.4

[1742 rows x 4 columns]
```

1.3.6 Get Borehole Cordinates Where There is DAS Fiber

```
[16]: idas_overlap_beg = 38 #TAC: starting index where das and boreholes haverlap␣
        ↪coordinates (empirically determined)
      boredas_df = sp1_df[idas_overlap_beg:]
```

SEGY Headers for Water-Depth Has "is borehole" Groupings

```
[17]: borehdr_df = hset_df[hset_df['FileIndex']==0]
      borehdr_df = borehdr_df[borehdr_df["GroupWaterDepth"] != 0] # TAC: ignore␣
        ↪channels with WaterDepth = 0
      bhdr_latlon_df = borehdr_df.drop_duplicates(subset=['GroupLat',␣
        ↪'GroupLon'])[['GroupLat', 'GroupLon', 'GroupX', 'GroupY']]
      bhdr_latlon_df['BoreIndex'] = range(len(bhdr_latlon_df))
      bhdr_latlon_df.iloc[:4]
```

```
[17]:      GroupLat    GroupLon    GroupX     GroupY  BoreIndex
      24  49.093749 -103.076428  640426.9  5439659.4          0
      42  49.094424 -103.076437  640424.4  5439734.4          1
      59  49.095071 -103.076435  640422.7  5439806.3          2
      77  49.095719 -103.076433  640421.0  5439878.4          3
```

1.4 Show Overview Field Map

```
[18]: m_width = 1200
      m_height = (1/1.66666)*m_width

      glst = []
      bore_d =␣
        ↪_make_marker_dict(bhdr_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='black',mname='B
      bore_g = _make_folium_group_from_dict(bore_d,gname='DAS Boreholes')
      abore_d =␣
        ↪_make_marker_dict(sp1_df,kw_lat='Latitude',kw_lon='Longitude',color='black',mname='BORE',sy
      abore_g = _make_folium_group_from_dict(abore_d,gname='All Boreholes')
      src_d =␣
        ↪_make_marker_dict(src_latlon_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SR
      src_g = _make_folium_group_from_dict(src_d,gname='All Sources')
      rec_d =␣
        ↪_make_marker_dict(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='REC
      rec_g = _make_folium_group_from_dict(rec_d,gname='All Channels')

      glst.append(src_g)
      glst.append(bore_g)
      glst.append(abore_g)
      glst.append(rec_g)

      clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

      base_map = _make_folium_base_map(clat,clon,zoom_start=14)

      my_map = _add_folium_groups_to_map(glst,base_map)
```

1.4.1 Convert Folium Map to HTML and Display

[19]:

```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

1.4.2 Save Map to HTML File

[20]:

```
 map_fname = data_path + '/field_overview_map.html'
print(map_fname)
```

```
/shared/data/aquistore/field_overview_map.html
```

[21]:

```
# my_map.save(map_fname)
```

1.5 Get Borehole Channel Indices

[22]:

```
file_df = hset_df[hset_df['FileIndex'] == 0] #TAC: We just need one file
```

[23]:

```
file_df[file_df["GroupWaterDepth"] != 0]
```

[23]:

```
      FileIndex  DataIndex  TRACE_SEQUENCE_LINE  TRACE_SEQUENCE_FILE  \
24            0         24                   24                   25
25            0         25                   25                   26
42            0         42                   42                   43
43            0         43                   43                   44
59            0         59                   59                   60
…           …        …                  …                   …
2089          0       2089                 2089                 2090
2106          0       2106                 2106                 2107
2107          0       2107                 2107                 2108
2123          0       2123                 2123                 2124
2124          0       2124                 2124                 2125

      FieldRecord  TraceNumber  EnergySourcePoint  CDP  CDP_TRACE  \
24             21           24                  0    0         25
25             21           25                  0    0         26
42             21           42                  0    0         43
```

```
43             21           43                  0    0         44
59             21           59                  0    0         60
…           …          …                …  …        …
2089           21         2089                  0    0       2090
2106           21         2106                  0    0       2107
2107           21         2107                  0    0       2108
2123           21         2123                  0    0       2124
2124           21         2124                  0    0       2125

      TraceIdentificationCode  …   GroupLat    GroupLon  OffsetX  OffsetY  \
24                          1  …  49.093749 -103.076428  -1281.9   1608.7
25                          1  …  49.093749 -103.076428  -1281.9   1608.7
42                          1  …  49.094424 -103.076437  -1279.4   1533.7
43                          1  …  49.094424 -103.076437  -1279.4   1533.7
59                          1  …  49.095071 -103.076435  -1277.7   1461.8
…                       …  …        …         …      …      …
2089                        1  …  49.095071 -103.076435  -1277.7   1461.8
2106                        1  …  49.094424 -103.076437  -1279.4   1533.7
2107                        1  …  49.094424 -103.076437  -1279.4   1533.7
2123                        1  …  49.093749 -103.076428  -1281.9   1608.7
2124                        1  …  49.093749 -103.076428  -1281.9   1608.7

      MidpointX  MidpointY    OffsetMag  RecGatherID  SrcGatherID  SrcUniqueID
24      -640.95     804.35  2056.984030         1741            0            0
25      -640.95     804.35  2056.984030         1741            0            0
42      -639.70     766.85  1997.273154         1708            0            0
43      -639.70     766.85  1997.273154         1708            0            0
59      -638.85     730.90  1941.488226         1677            0            0
…         …        …          …          …          …          …
2089    -638.85     730.90  1941.488226         1677            0            0
2106    -639.70     766.85  1997.273154         1708            0            0
2107    -639.70     766.85  1997.273154         1708            0            0
2123    -640.95     804.35  2056.984030         1741            0            0
2124    -640.95     804.35  2056.984030         1741            0            0

[224 rows x 105 columns]
```

[24]:

```python
is_bore = file_df["GroupWaterDepth"] != 0
is_bore = is_bore.to_numpy()
isbl = is_bore[:-1]
isbr = is_bore[1:]
is_bore_p1 = is_bore.copy()
is_bore_p1[1:] = is_bore_p1[1:] | isbl
is_bore_p1[:-1] = is_bore_p1[:-1] | isbr
is_surf = ~is_bore
is_surf_p1 = ~is_bore_p1
ibore_mask = file_df.index.to_numpy()[is_bore]
```

```
isurf_mask = file_df.index.to_numpy()[is_surf]
ibore_mask_p1 = file_df.index.to_numpy()[is_bore_p1]
isurf_mask_p1 = file_df.index.to_numpy()[is_surf_p1]
print(len(ibore_mask))
print(len(isurf_mask))
assert len(isurf_mask)+len(ibore_mask) == len(file_df)
```

```
224
1908
```

1.6 Preprocess DAS Data

[25]:

```
from scipy import signal

def butter_bandpass(data,fs=None,b0=None,bN=None,axis=-1,order=5,**kwargs):

    bmode = 'bandpass'
    bands = (b0,bN)

    if b0 is None and bN is None:
        raise Exception('b0 and bN are both None')
    if b0 is None:
        bmode = 'lowpass'
        bands = (bN)
    elif bN is None:
        bmode = 'highpass'
        bands = (b0)

    sos = signal.butter(order, bands, bmode, fs=fs, output='sos')
    return signal.sosfiltfilt(sos,data,axis)
```

[26]:

```
proc_lst = []
for data in dlst:
    dmed_data = sci_detrend(data,type='constant',axis=-1)
    dlin_data = sci_detrend(dmed_data,type='linear',axis=-1)
    butter_data = butter_bandpass(data,fs=1000,b0=0.5,bN=50)
    rms_norm = np.sqrt(np.mean(butter_data**2, axis=1, keepdims=True))
    norm_data = butter_data/rms_norm
    proc_lst.append(norm_data)
```

1.6.1 Get Data for Two Shots

[27]:

```
fid = 0
file_proc = proc_lst[fid]
print(file_proc.shape)

fid2 = 201
file_proc2 = proc_lst[fid2]
```

```
print(file_proc2.shape)
```

```
(2132, 4000)
(2132, 4000)
```

1.6.2 Get Source Coordinates for Each File (shot)

[28]:
```
src_df = hset_df[hset_df['FileIndex']==fid]
src_df = src_df[['SourceLat','SourceLon']].drop_duplicates()
src_df
```

[28]:
```
   SourceLat   SourceLon
0  49.108503 -103.093424
```

[29]:
```
src2_df = hset_df[hset_df['FileIndex']==fid2]
src2_df = src2_df[['SourceLat','SourceLon']].drop_duplicates()
src2_df
```

[29]:
```
   SourceLat   SourceLon
0  49.099163 -103.066758
```

1.6.3 Get Metadata for Data Plotting

[30]:
```
nt = hset_df.attrs['SampleCount'][fid]
dt = hset_df.attrs['TimeDelta'][fid]
times = dt*np.arange(nt)
```

1.7 Plot Shot for First File

1.7.1 Show Field Map

[31]:
```
m_width = 1200
m_height = (1/1.66666)*m_width

glst = []
bore_d =␣
 ↪_make_marker_dict(bhdr_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='black',mname='B
bore_g = _make_folium_group_from_dict(bore_d,gname='DAS Boreholes')
src_d =␣
 ↪_make_marker_dict(src_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SRC',symb
src_g = _make_folium_group_from_dict(src_d,gname='Source')
rec_d =␣
 ↪_make_marker_dict(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='REC
rec_g = _make_folium_group_from_dict(rec_d,gname='All Channels')

glst.append(src_g)
glst.append(bore_g)
glst.append(rec_g)
```

```
clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

base_map = _make_folium_base_map(clat,clon,zoom_start=14)

my_map = _add_folium_groups_to_map(glst,base_map)
```

[32]:

```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

[33]:

```
src_map_fname = data_path + '/src_map.html'
print(src_map_fname)
```

```
/shared/data/aquistore/src_map.html
```

[34]:

```
# my_map.save(src_map_fname)
```

1.7.2 Plot All Channels

[35]:

```
#TAC: uncomment after all initial plots (Cloud Workstation Cluster Problem␣
  ↪Sometimes)
%matplotlib inline
```

1.7.3 Zoom Box

[36]:

```
z_anch = (540, 2.3)
x_zlim = (540,740)
y_zlim = (2.3,3.3)
zoom_rect = patches.Rectangle(
    z_anch,                         # anchor
    x_zlim[1] - x_zlim[0],          # channels
    y_zlim[1] - y_zlim[0],          # time range
    linewidth=4,
    edgecolor='red',
    facecolor='none'
)
```

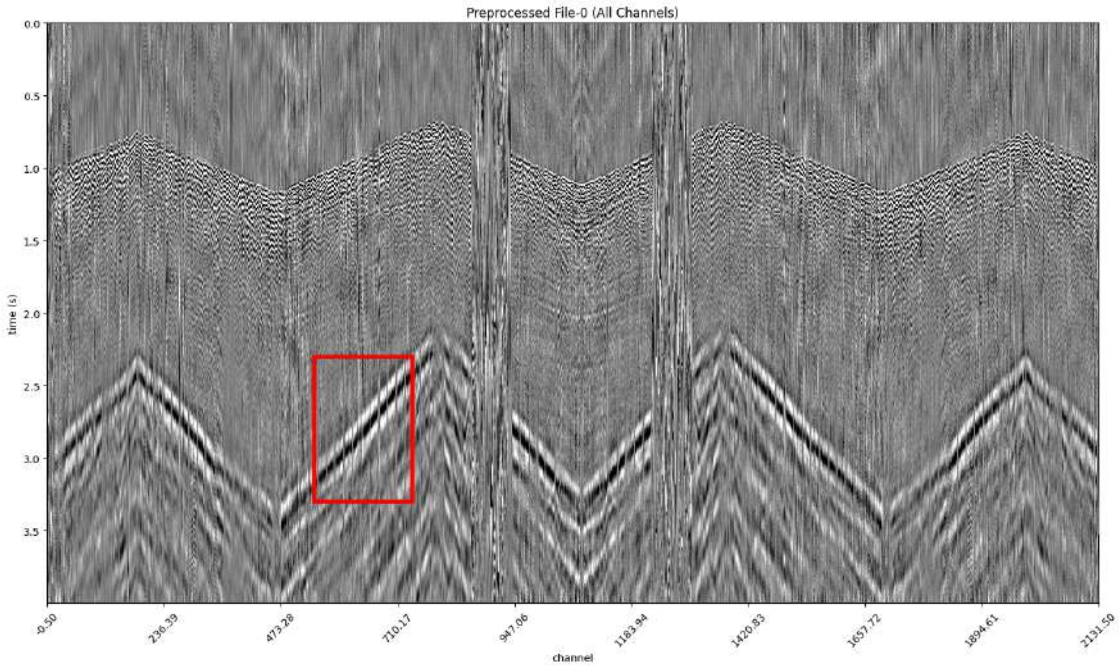
[37]:

```
AC, TA = np.meshgrid(np.arange(file_proc.shape[0]), times)
```

[38]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc.T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(AC, TA, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)
```

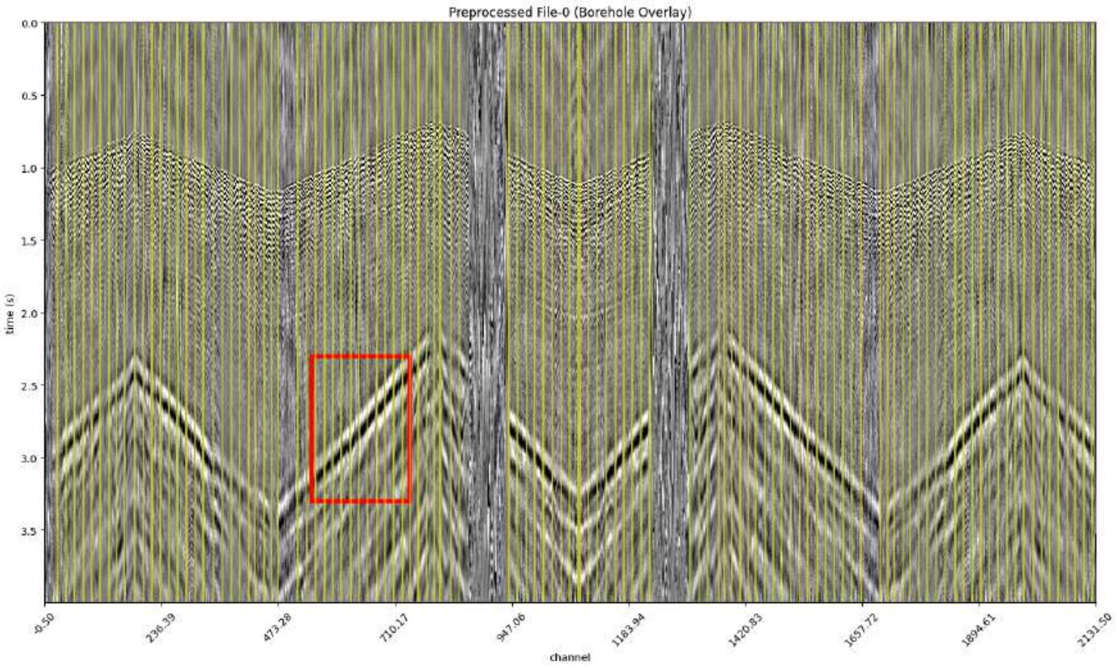
```
ax.add_patch(dcopy(zoom_rect))

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (All Channels)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.7.4 Plot All Channels with Borehole Channel Markers Overlain (indicated in channel headers)

[39]:
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc.T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(AC, TA, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)
```

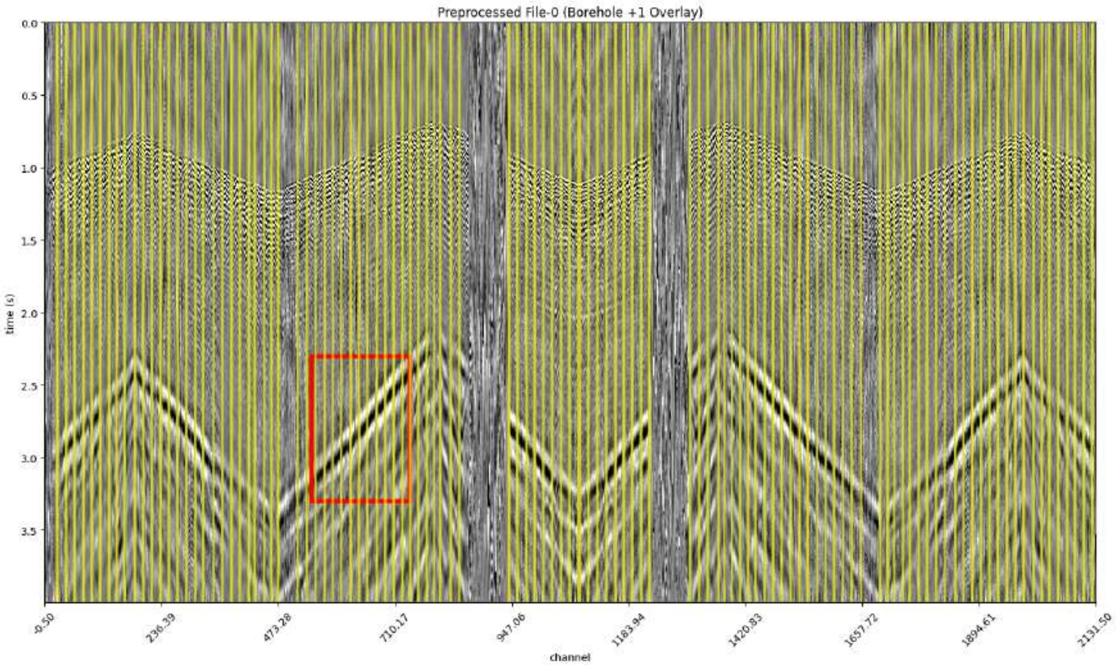
```
ax.add_patch(dcopy(zoom_rect))

for ic in ibore_mask:
    ax.axvspan(ic-0.5, ic+0.5, color='yellow', alpha=0.3)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Borehole Overlay)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.7.5 Plot All Channels with Borehole Channel Markers Overlain (header locations plus nearest channel neighbors)

[40]:

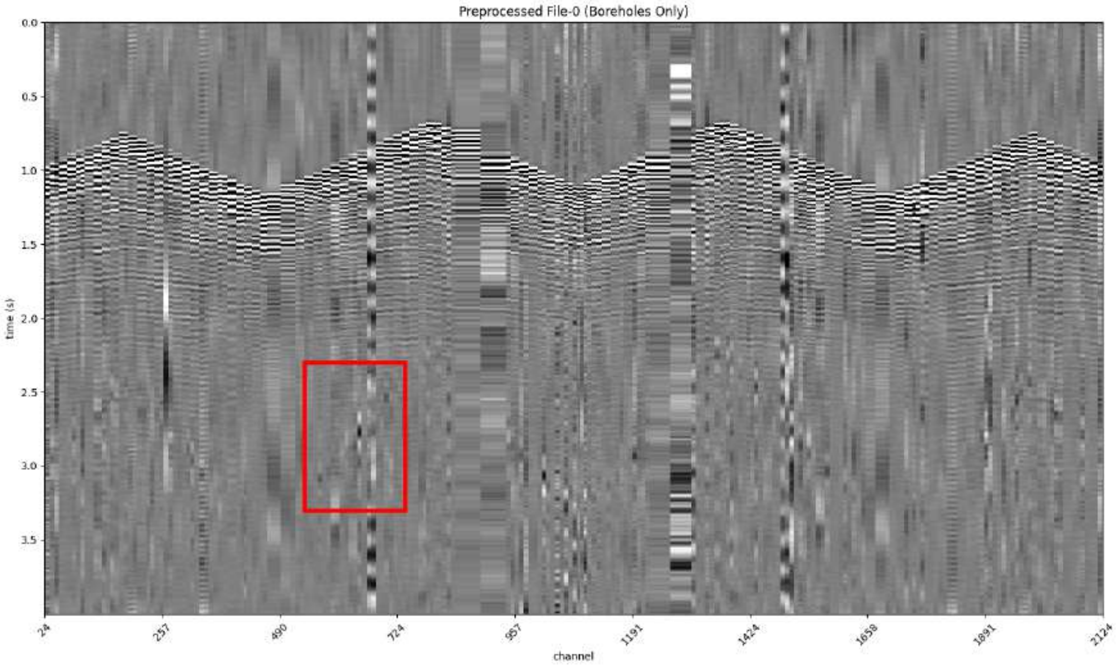
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc.T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(AC, TA, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

ax.add_patch(dcopy(zoom_rect))

for ic in ibore_mask_p1:
    ax.axvspan(ic-0.5, ic+0.5, color='yellow', alpha=0.3)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Borehole +1 Overlay)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.7.6 Plot Borehole Channels Only (as indecated by channel headers)

[41]:
```
BC1, TB1 = np.meshgrid(ibore_mask, times)
```

[42]:
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc[ibore_mask,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(BC1, TB1, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

ax.add_patch(dcopy(zoom_rect))

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Boreholes Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



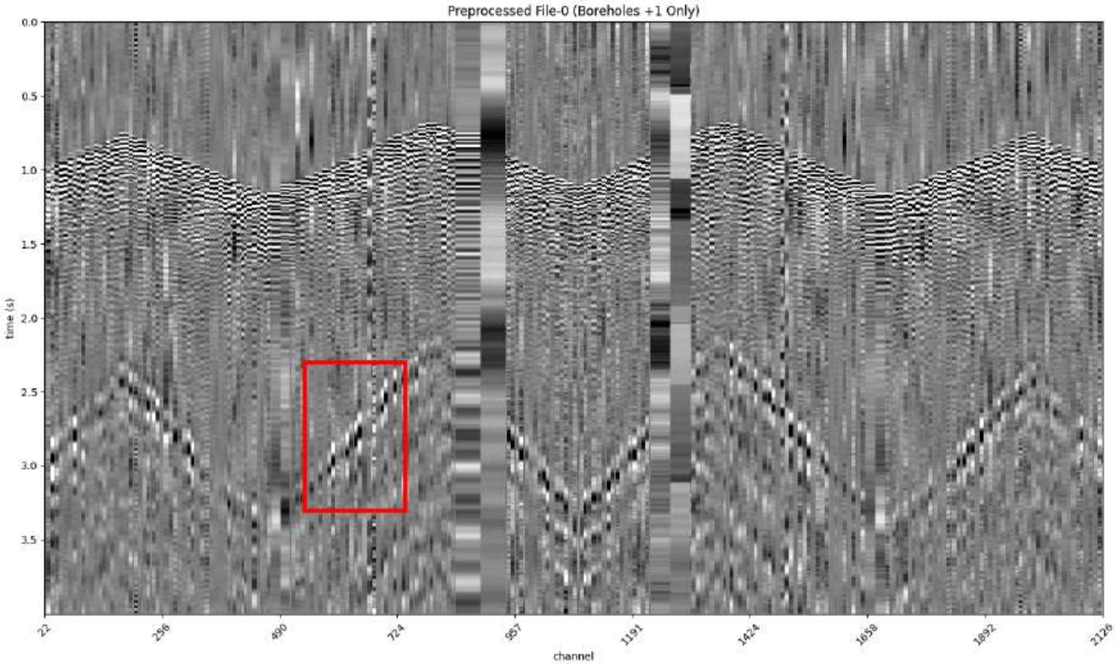
1.7.7 Plot Borehole Channels Plus Nearest Neighbor Channels

[43]:

```
BC2, TB2 = np.meshgrid(ibore_mask_p1, times)
```

[44]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc[ibore_mask_p1,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(BC2, TB2, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

ax.add_patch(dcopy(zoom_rect))

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Boreholes +1 Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.7.8 Zoom Plot All vs Borehole (vertical-loop) Channels

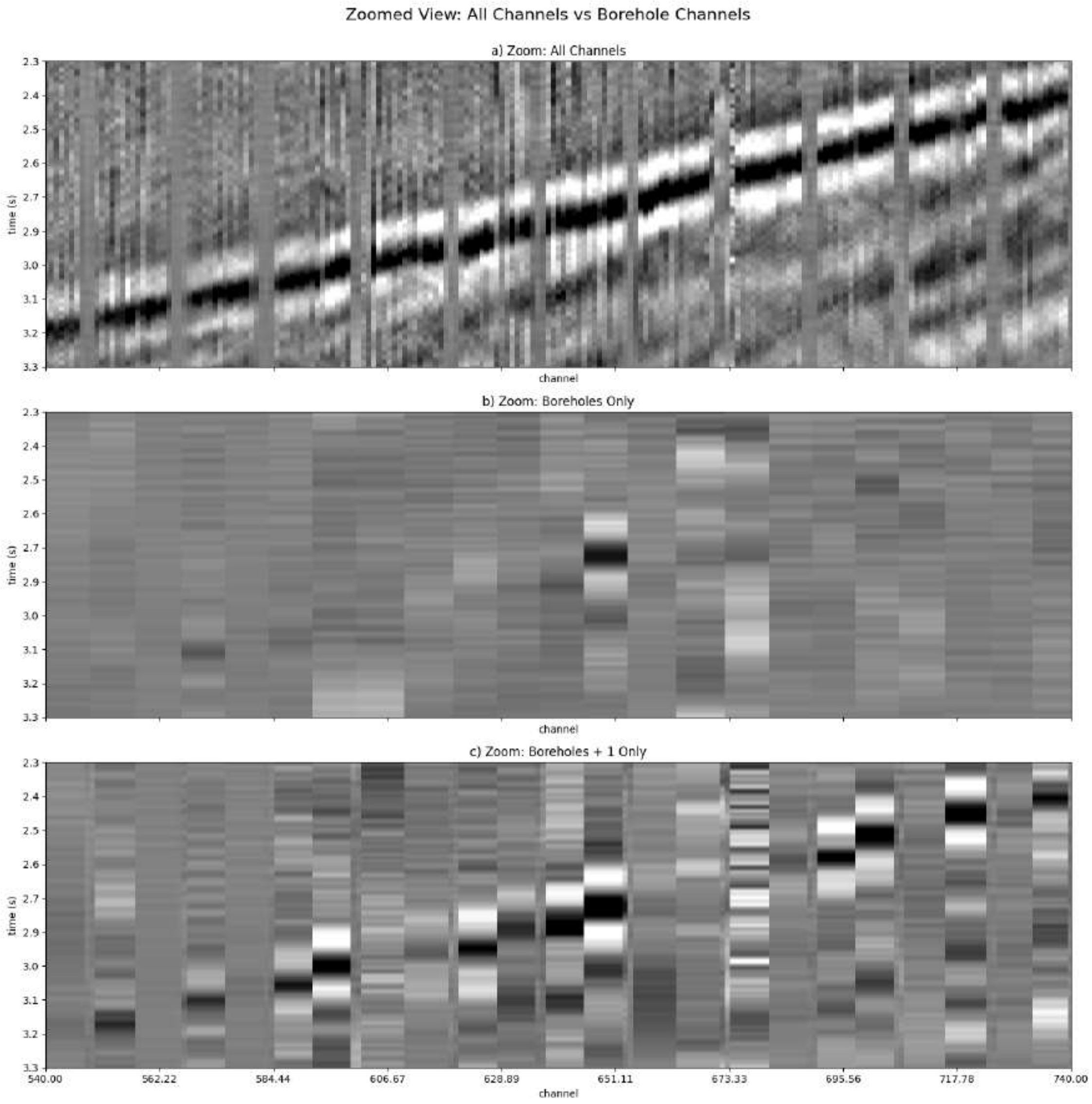
[45]:

```
plt_data1 = file_proc.T
plt_data2 = file_proc[ibore_mask,:].T
plt_data3 = file_proc[ibore_mask_p1,:].T
```

[46]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
xlim_zoom = x_zlim
ylim_zoom = y_zlim
tick_positions_x = np.linspace(*xlim_zoom, 10)
tick_positions_y = np.linspace(*ylim_zoom, 10)[::-1]  # Reversed for inverted␣
 ↪axis



fig, axes = plt.subplots(nrows=3, figsize=(15, 15), sharex=True, sharey=True)

axes[0].pcolormesh(AC, TA, plt_data1, cmap='gray',vmin=-vmax,vmax=vmax)
axes[0].set_xlim(*x_zlim)
axes[0].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[0].set_xticks(tick_positions_x)
axes[0].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
axes[0].set_yticks(tick_positions_y)
axes[0].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[0].set_title(f'a) Zoom: All Channels')
axes[0].set_ylabel('time (s)')
axes[0].set_xlabel('channel')

axes[1].pcolormesh(BC1, TB1, plt_data2, cmap='gray',vmin=-vmax,vmax=vmax)
axes[1].set_xlim(*x_zlim)
axes[1].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[1].set_xticks(tick_positions_x)
axes[1].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
axes[1].set_yticks(tick_positions_y)
axes[1].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[1].set_title(f'b) Zoom: Boreholes Only')
axes[1].set_ylabel('time (s)')
axes[1].set_xlabel('channel')

axes[2].pcolormesh(BC2, TB2, plt_data3, cmap='gray',vmin=-vmax,vmax=vmax)
axes[2].set_xlim(*x_zlim)
axes[2].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[2].set_xticks(tick_positions_x)
axes[2].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
```

```
axes[2].set_yticks(tick_positions_y)
axes[2].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[2].set_title(f'c) Zoom: Boreholes + 1 Only')
axes[2].set_ylabel('time (s)')
axes[2].set_xlabel('channel')

fig.suptitle("Zoomed View: All Channels vs Borehole Channels", fontsize=16, y=1.0)

plt.tight_layout()
plt.show()
```



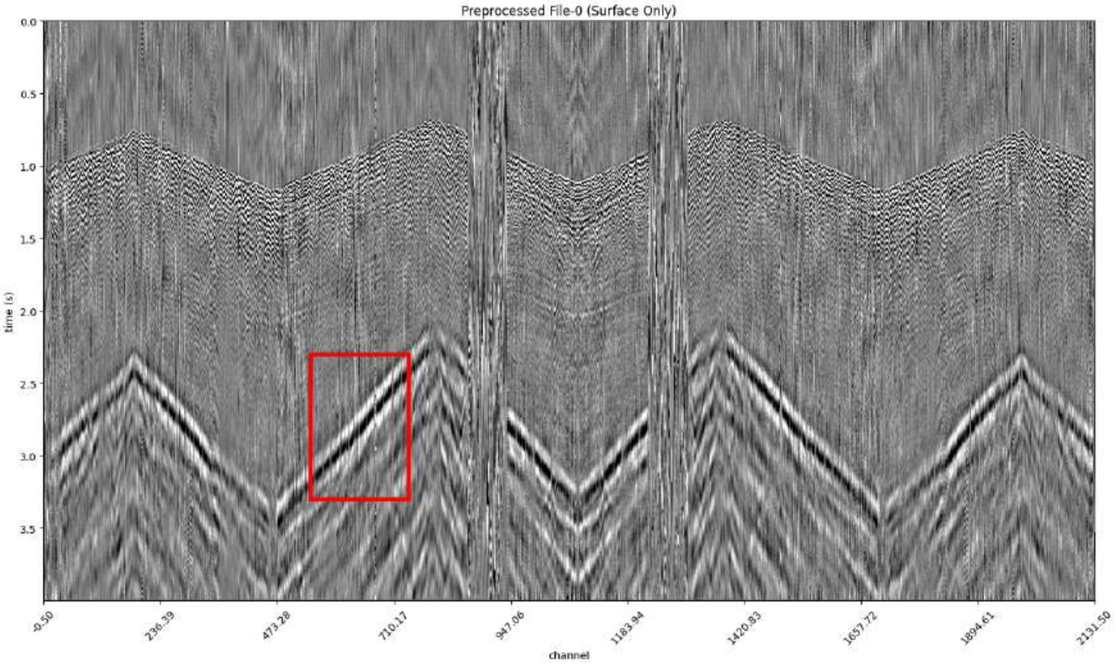
1.7.9 Plot Surface Channels Only (as indecated by channel headers)

[47]:

```
SC1, TS1 = np.meshgrid(isurf_mask, times)
```

[48]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc[isurf_mask,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC1, TS1, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

ax.add_patch(dcopy(zoom_rect))

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Surface Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



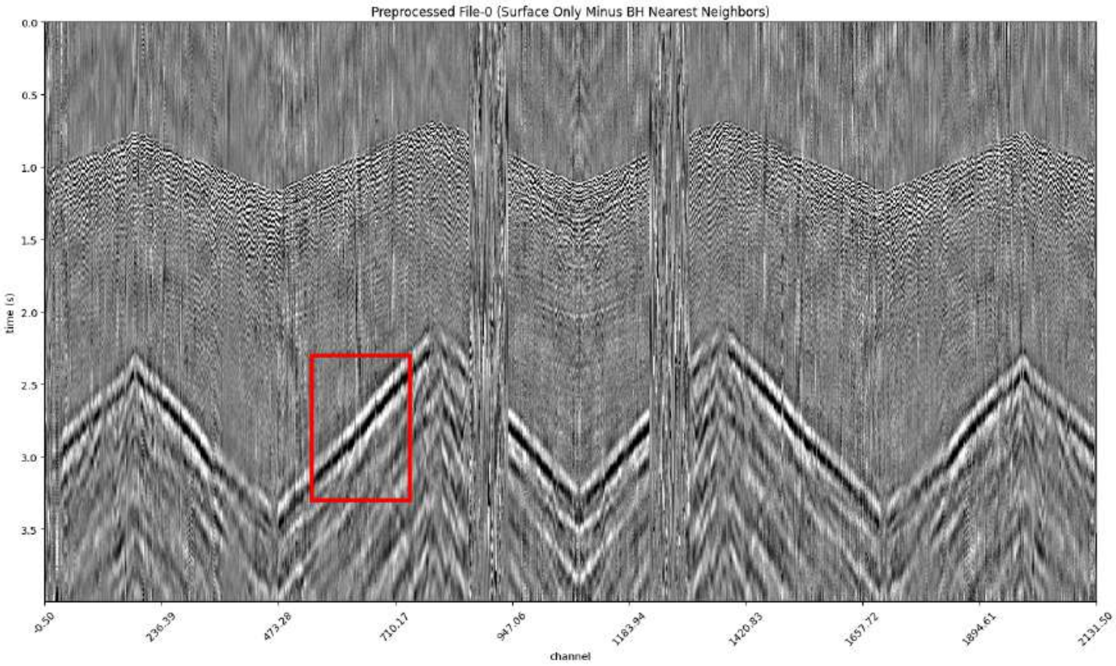
1.7.10 Plot Surface Channels without Borehole Channels and without Borehole Nearest Neighbors

[49]:

```
SC2, TS2 = np.meshgrid(isurf_mask_p1, times)
```

[50]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc[isurf_mask_p1,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC2, TS2, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

ax.add_patch(dcopy(zoom_rect))

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Surface Only Minus BH Nearest␣
 ↪Neighbors)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.7.11 Zoom Plot All vs Surface Channels

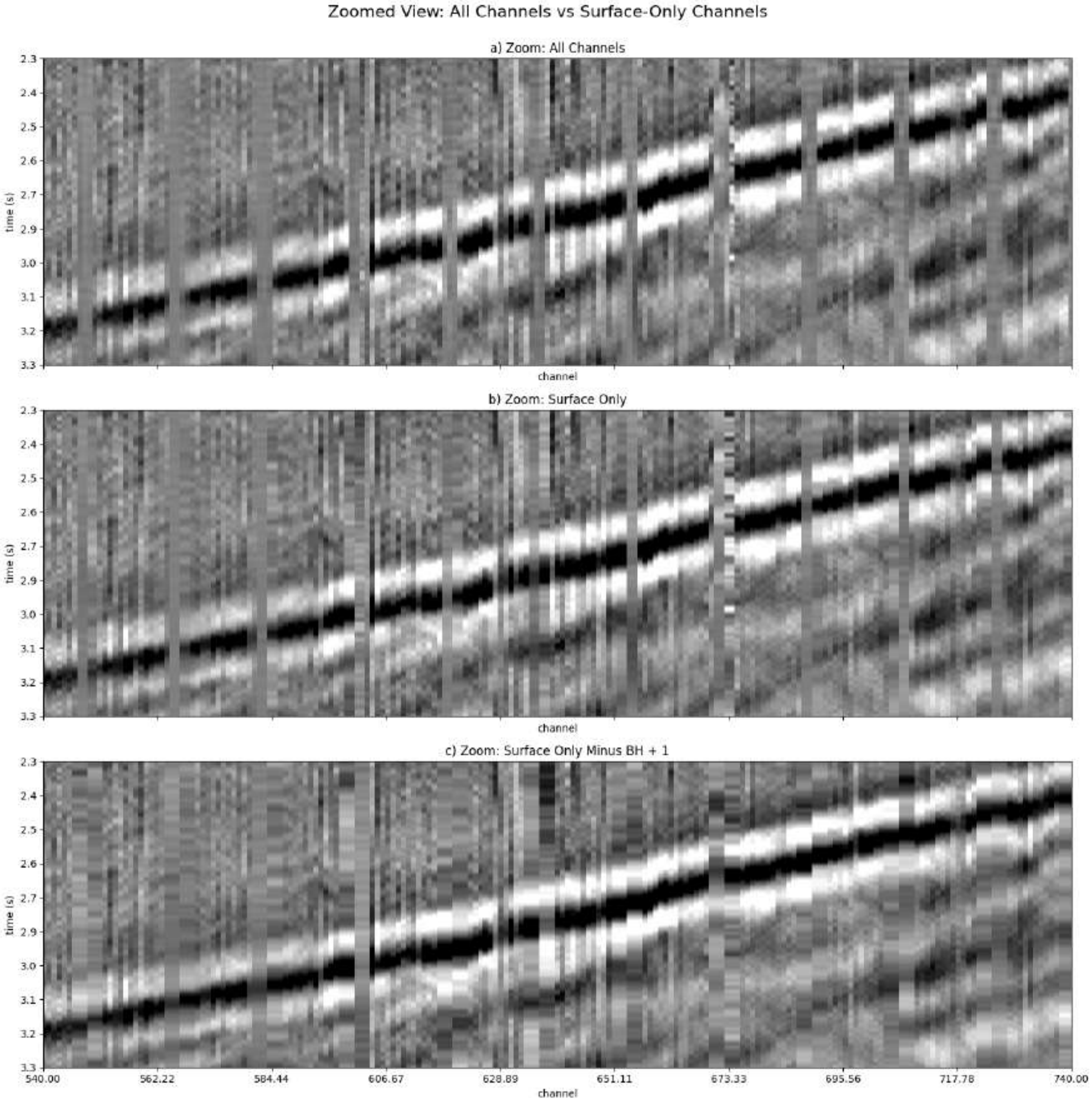
[51]:

```python
plt_data1 = file_proc.T
plt_data2 = file_proc[isurf_mask,:].T
plt_data3 = file_proc[isurf_mask_p1,:].T
```

[52]:

```python
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
xlim_zoom = x_zlim
ylim_zoom = y_zlim
tick_positions_x = np.linspace(*xlim_zoom, 10)
tick_positions_y = np.linspace(*ylim_zoom, 10)[::-1]  # Reversed for inverted axis



fig, axes = plt.subplots(nrows=3, figsize=(15, 15), sharex=True, sharey=True)

axes[0].pcolormesh(AC, TA, plt_data1, cmap='gray',vmin=-vmax,vmax=vmax)
axes[0].set_xlim(*x_zlim)
axes[0].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[0].set_xticks(tick_positions_x)
axes[0].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
axes[0].set_yticks(tick_positions_y)
axes[0].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[0].set_title(f'a) Zoom: All Channels')
axes[0].set_ylabel('time (s)')
axes[0].set_xlabel('channel')

axes[1].pcolormesh(SC1, TS1, plt_data2, cmap='gray',vmin=-vmax,vmax=vmax)
axes[1].set_xlim(*x_zlim)
axes[1].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[1].set_xticks(tick_positions_x)
axes[1].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
axes[1].set_yticks(tick_positions_y)
axes[1].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[1].set_title(f'b) Zoom: Surface Only')
axes[1].set_ylabel('time (s)')
axes[1].set_xlabel('channel')

axes[2].pcolormesh(SC2, TS2, plt_data3, cmap='gray',vmin=-vmax,vmax=vmax)
axes[2].set_xlim(*x_zlim)
axes[2].set_ylim(*y_zlim[::-1])  # y-axis inverted
axes[2].set_xticks(tick_positions_x)
axes[2].xaxis.set_major_formatter(FormatStrFormatter('%.2f'))
```

```
axes[2].set_yticks(tick_positions_y)
axes[2].yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
axes[2].set_title(f'c) Zoom: Surface Only Minus BH + 1')
axes[2].set_ylabel('time (s)')
axes[2].set_xlabel('channel')

fig.suptitle("Zoomed View: All Channels vs Surface-Only Channels", fontsize=16,␣
 ↪y=1.0)

plt.tight_layout()
plt.show()
```



1.8 Plot Shot for Second File

1.8.1 Show Field Map

[53]:
```
m_width = 1200
m_height = (1/1.66666)*m_width

glst = []
bore_d =␣
 ↪_make_marker_dict(bhdr_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='black',mname='B
bore_g = _make_folium_group_from_dict(bore_d,gname='DAS Boreholes')
src_d =␣
 ↪_make_marker_dict(src2_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SRC',sym
src_g = _make_folium_group_from_dict(src_d,gname='Source')
rec_d =␣
 ↪_make_marker_dict(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='REC
rec_g = _make_folium_group_from_dict(rec_d,gname='All Channels')

glst.append(src_g)
glst.append(bore_g)
glst.append(rec_g)

clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

base_map = _make_folium_base_map(clat,clon,zoom_start=14)

my_map = _add_folium_groups_to_map(glst,base_map)
```

[54]:
```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

[55]:
```
src_map_fname = data_path + '/src2_map.html'
print(src_map_fname)
```

```
/shared/data/aquistore/src2_map.html
```

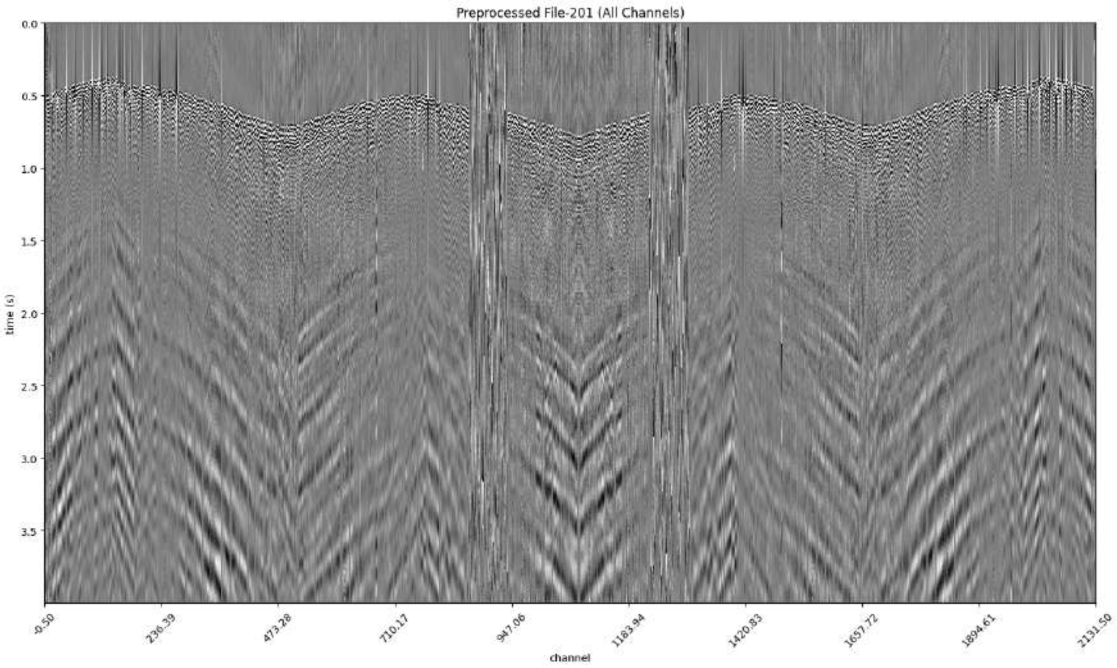
[56]:
```
# my_map.save(src_map_fname)
```

1.8.2 Plot All Channels

[57]:
```
AC, TT = np.meshgrid(np.arange(file_proc2.shape[0]), times)
```

[58]:
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2.T

fig, ax = plt.subplots(figsize=(15,9))
```

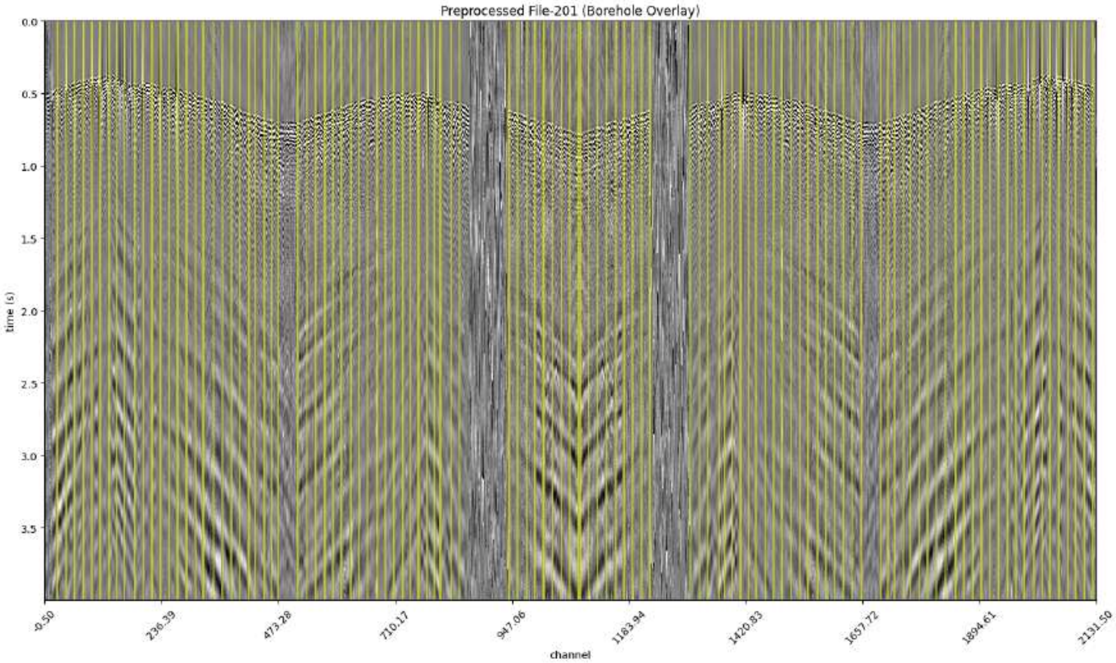
```
im = ax.pcolormesh(AC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (All Channels)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.8.3 Plot All Channels with Borehole Channel Markers Overlain (indicated in channel headers)

[59]:
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2.T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(AC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)
```

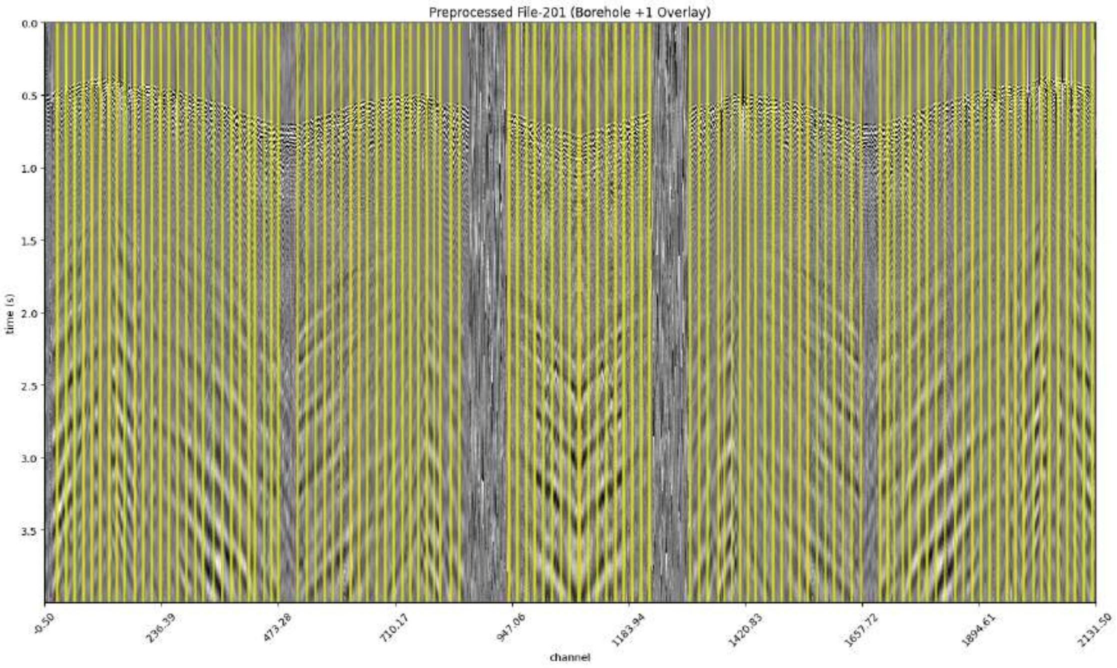
```
for ic in ibore_mask:
    ax.axvspan(ic-0.5, ic+0.5, color='yellow', alpha=0.3)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Borehole Overlay)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.8.4 Plot All Channels with Borehole Channel Markers Overlain (header locations plus nearest channel neighbors)

[60]:
```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2.T
```

```
fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(AC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

for ic in ibore_mask_p1:
    ax.axvspan(ic-0.5, ic+0.5, color='yellow', alpha=0.3)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Borehole +1 Overlay)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```

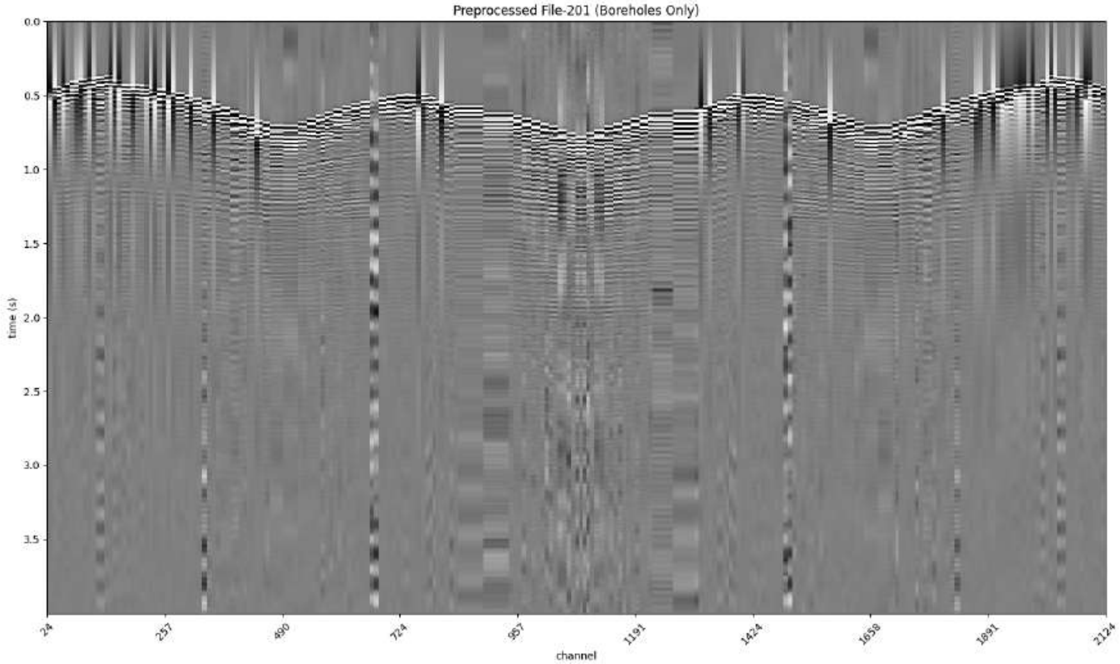


1.8.5 Plot Borehole Channels Only (as indecated by channel headers)

```
[61]: BC, TT = np.meshgrid(ibore_mask, times)
```

[62]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2[ibore_mask,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(BC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Boreholes Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



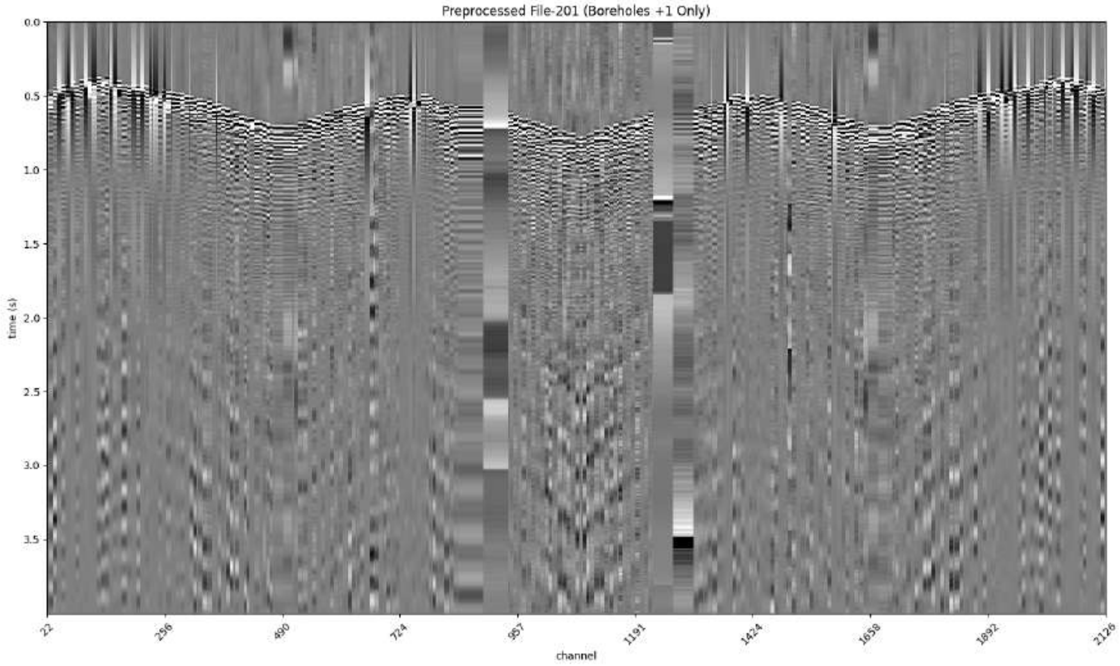
1.8.6 Plot Borehole Channels Plus Nearest Neighbor Channels

[63]:

```
BC, TT = np.meshgrid(ibore_mask_p1, times)
```

[64]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2[ibore_mask_p1,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(BC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Boreholes +1 Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



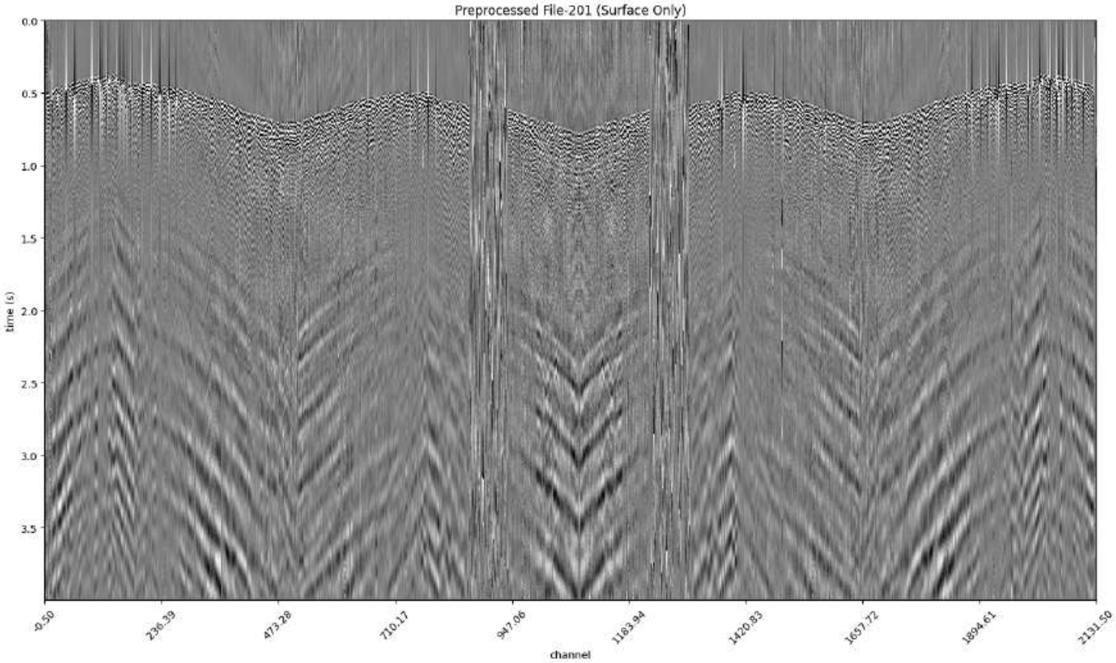
1.8.7 Plot Surface Channels Only (as indecated by channel headers)

[65]:

```
SC, TT = np.meshgrid(isurf_mask, times)
```

[66]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2[isurf_mask,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Surface Only)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



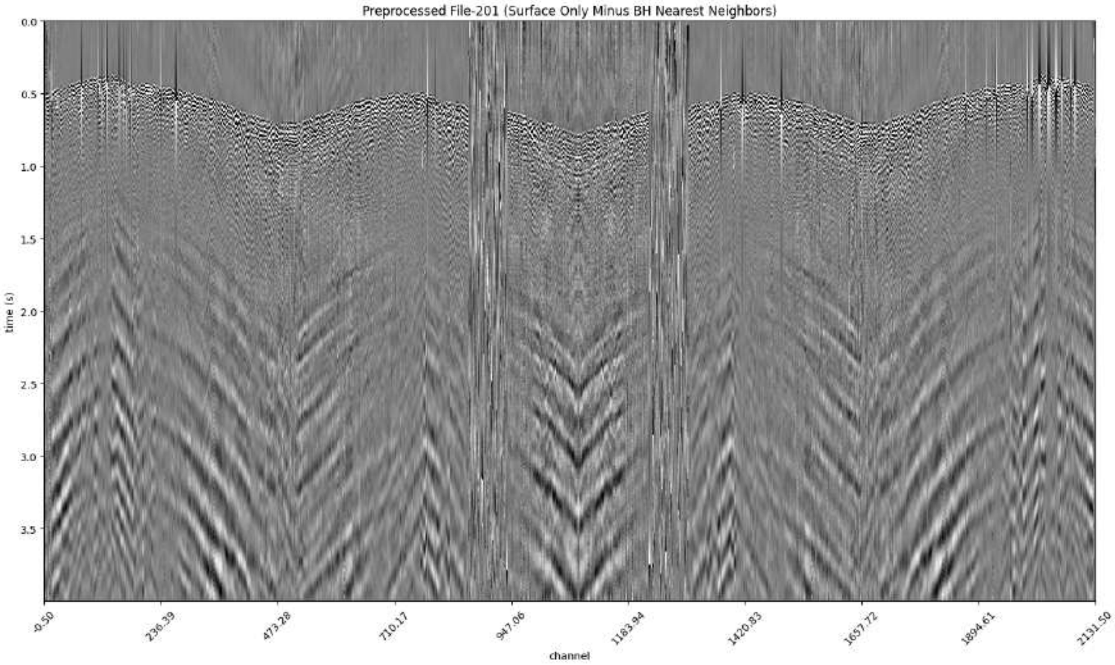
1.8.8 Plot Surface Channels without Borehole Channels and without Borehole Nearest Neighbors

[67]:

```
SC, TT = np.meshgrid(isurf_mask_p1, times)
```

[68]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc2).max()
plt_data = file_proc2[isurf_mask_p1,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid2} (Surface Only Minus BH Nearest␣
  ↪Neighbors)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.9 Remove Surface Slack Channels

Emerically Searched for Channels

[69]:
```
#TAC: emerically found
ifrom = 46
ito = 47

slack_df = bhdr_latlon_df[(ifrom <= bhdr_latlon_df['BoreIndex']) & ␣
 ↪(bhdr_latlon_df['BoreIndex'] <= ito)]
display(slack_df.iloc[:4])
```

```
      GroupLat    GroupLon    GroupX     GroupY  BoreIndex
839  49.097319 -103.082228  639993.5  5440045.5         46
940  49.093428 -103.082242  640003.4  5439612.9         47
```

[70]:
```
sortx = np.sort(slack_df['GroupX'].to_numpy())
sorty = np.sort(slack_df['GroupY'].to_numpy())
gdx = 0.5*abs(sortx[1] - sortx[0])
gdy = 1
min_gx = sortx[0] - gdx
max_gx = sortx[1] + gdx
min_gy = sorty[0] + gdy
max_gy = sorty[1] - gdy
```

Add Column to Channel (receiver) DataFrame for "Is_Slack" Mask

[71]:
```
slack_mask = (
    (hset_df['GroupX'] >= min_gx) & (hset_df['GroupX'] <= max_gx) &
    (hset_df['GroupY'] >= min_gy) & (hset_df['GroupY'] <= max_gy)
)
tagslack_df = hset_df.copy()
tagslack_df.loc[:,'Is_Slack'] = slack_mask
tagslack_df
```

[71]:
```
      FileIndex  DataIndex  TRACE_SEQUENCE_LINE  TRACE_SEQUENCE_FILE  \
0             0          0                    0                    1
1             0          1                    1                    2
2             0          2                    2                    3
3             0          3                    3                    4
4             0          4                    4                    5
…           …          …                  …                  …
2127        377       2127                 2127                 2128
2128        377       2128                 2128                 2129
2129        377       2129                 2129                 2130
2130        377       2130                 2130                 2131
2131        377       2131                 2131                 2132

      FieldRecord  TraceNumber  EnergySourcePoint  CDP  CDP_TRACE  \
0              21            0                  0    0          1
```

```
1              21            1                  0    0          2
2              21            2                  0    0          3
3              21            3                  0    0          4
4              21            4                  0    0          5
…           …          …                …  …        …
2127          414         2127                  0    0       2128
2128          414         2128                  0    0       2129
2129          414         2129                  0    0       2130
2130          414         2130                  0    0       2131
2131          414         2131                  0    0       2132

      TraceIdentificationCode  …    GroupLon  OffsetX  OffsetY  MidpointX  \
0                           1  … -103.077200  -1227.0   1667.1    -613.50
1                           1  … -103.077169  -1229.2   1664.7    -614.60
2                           1  … -103.077137  -1231.5   1662.3    -615.75
3                           1  … -103.077105  -1233.8   1659.8    -616.90
4                           1  … -103.077072  -1236.1   1657.4    -618.05
…                       …  …         …      …      …        …
2127                        1  … -103.076772    215.9     369.5     107.95
2128                        1  … -103.076857    222.0     376.0     111.00
2129                        1  … -103.076943    228.1     382.5     114.05
2130                        1  … -103.077029    234.2     389.0     117.10
2131                        1  … -103.077115    240.3     395.5     120.15

      MidpointY    OffsetMag  RecGatherID  SrcGatherID  SrcUniqueID  Is_Slack
0        833.55  2069.964108         1489            0            0     False
1        832.35  2069.337752         1492            0            0     False
2        831.15  2068.775855         1493            0            0     False
3        829.90  2068.138893         1496            0            0     False
4        828.70  2067.587476         1498            0            0     False
…         …          …          …          …          …       …
2127     184.75   427.952170         1521          377          376     False
2128     188.00   436.646310         1514          377          376     False
2129     191.25   445.349144         1508          377          376     False
2130     194.50   454.060172         1502          377          376     False
2131     197.75   462.778932         1495          377          376     False

[805896 rows x 106 columns]
```

[72]:

```
noslack_df = tagslack_df[tagslack_df['FileIndex'] == 0] #TAC: We just need one file
```

Slice/Remove Channels Part of the Surface Slack (non-entrenched) Cable Portion

[73]:

```
rec_noslack_df = noslack_df.drop_duplicates(subset=['GroupLat', 'GroupLon'])[['SourceX','SourceY','OffsetX','OffsetY','GroupLat', 'GroupLon', 'GroupX', 'GroupY','Is_Slack']]
rec_noslack_df = rec_noslack_df[~rec_noslack_df['Is_Slack']]
```

```
rec_noslack_df
```

[73]:

```
        SourceX    SourceY  OffsetX  OffsetY   GroupLat    GroupLon    GroupX  \
0     639145.0  5441268.1  -1227.0   1667.1  49.093237 -103.077200  640372.0
1     639145.0  5441268.1  -1229.2   1664.7  49.093258 -103.077169  640374.2
2     639145.0  5441268.1  -1231.5   1662.3  49.093279 -103.077137  640376.5
3     639145.0  5441268.1  -1233.8   1659.8  49.093301 -103.077105  640378.8
4     639145.0  5441268.1  -1236.1   1657.4  49.093322 -103.077072  640381.1
…          …          …        …        …          …           …         …
2125  639145.0  5441268.1  -1269.7   1621.7  49.093635 -103.076600  640414.7
2127  639145.0  5441268.1  -1257.5   1634.7  49.093521 -103.076772  640402.5
2128  639145.0  5441268.1  -1251.4   1641.2  49.093464 -103.076857  640396.4
2130  639145.0  5441268.1  -1239.2   1654.2  49.093350 -103.077029  640384.2
2131  639145.0  5441268.1  -1233.1   1660.7  49.093293 -103.077115  640378.1

         GroupY  Is_Slack
0     5439601.0     False
1     5439603.4     False
2     5439605.8     False
3     5439608.3     False
4     5439610.7     False
…           …       …
2125  5439646.4     False
2127  5439633.4     False
2128  5439626.9     False
2130  5439613.9     False
2131  5439607.4     False

[1565 rows x 9 columns]
```

1.9.1 Show Field Overview Map Without the Slack Channels

[74]:

```
m_width = 1200
m_height = (1/1.66666)*m_width

glst = []
bore_d =␣
 ↪_make_marker_dict(bhdr_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='black',mname='B
bore_g = _make_folium_group_from_dict(bore_d,gname='DAS Boreholes')
abore_d =␣
 ↪_make_marker_dict(sp1_df,kw_lat='Latitude',kw_lon='Longitude',color='black',mname='BORE',sy
abore_g = _make_folium_group_from_dict(abore_d,gname='All Boreholes')
src_d =␣
 ↪_make_marker_dict(src_latlon_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SR
src_g = _make_folium_group_from_dict(src_d,gname='All Sources')
rec_d =␣
 ↪_make_marker_dict(rec_noslack_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='RE
rec_g = _make_folium_group_from_dict(rec_d,gname='Channels')

glst.append(src_g)
glst.append(bore_g)
glst.append(abore_g)
glst.append(rec_g)

clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

base_map = _make_folium_base_map(clat,clon,zoom_start=14)

my_map = _add_folium_groups_to_map(glst,base_map)
```

[75]:

```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

[76]:

```
noslack_map_fname = data_path + '/noslack_overview_field_map.html'
print(noslack_map_fname)
```

```
/shared/data/aquistore/noslack_overview_field_map.html
```

[77]:

```
# my_map.save(noslack_map_fname)
```

1.10 Plot Surface Channels With Slack Channels Set to Zero and Dropped

[78]:

```
is_slack = noslack_df['Is_Slack'].to_numpy()
is_not_slack = ~is_slack
is_not_slack
```

[78]:

```
array([ True,  True,  True, …,  True,  True,  True], shape=(2132,))
```

Zero-out Slack Channels

[79]:

```
nos_file_proc = file_proc.copy()
nos_file_proc[is_slack,:] = 0.
```

To “Drop” Slack Channels

[80]:

```
isurf_nos_mask = file_df.index.to_numpy()[is_surf & is_not_slack]
isurf_nos_mask
```

[80]:

```
array([   0,    1,    2, …, 2129, 2130, 2131], shape=(1729,))
```

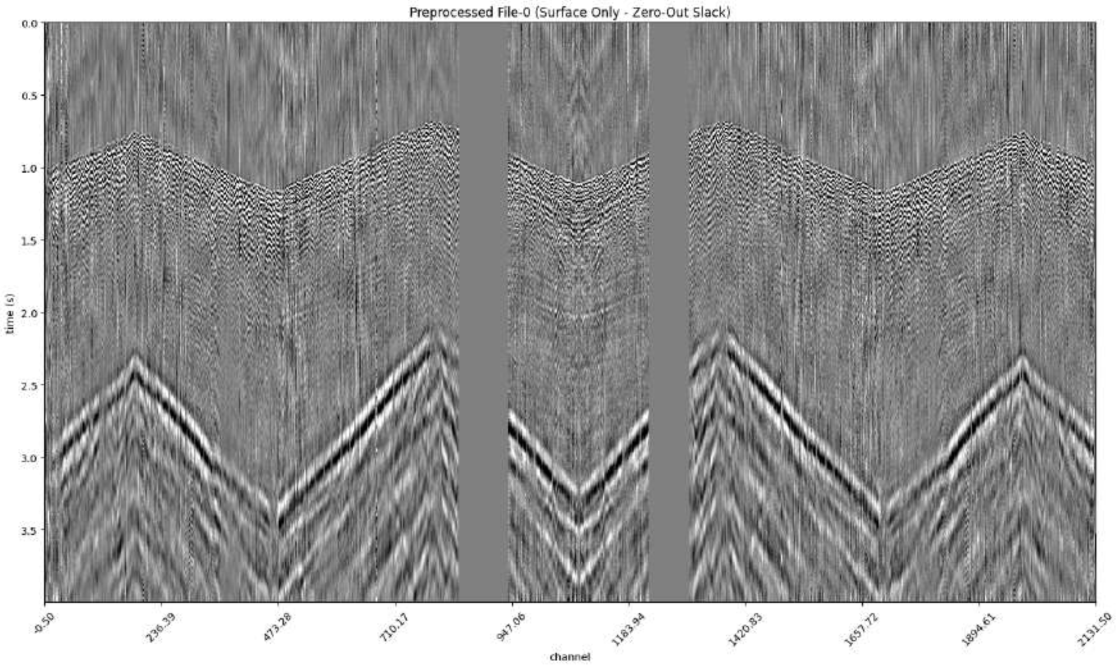
1.10.1 Plot Zero-out Slack Surface Channels

[81]:

```
SC, TT = np.meshgrid(isurf_mask, times)
```

[82]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = nos_file_proc[isurf_mask,:].T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Surface Only - Zero-Out Slack)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



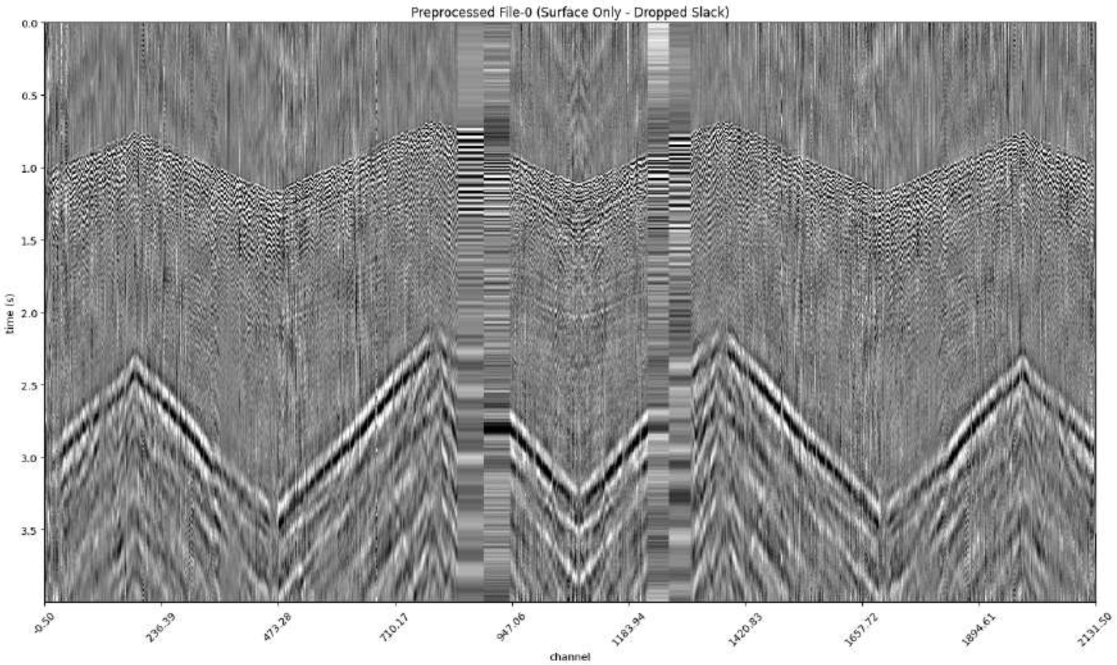
1.10.2 Plot Dropped Slack Surface Channels

[83]:

```
SC, TT = np.meshgrid(isurf_nos_mask, times) #TAC: Drop slack instead
```

[84]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(file_proc).max()
plt_data = file_proc[isurf_nos_mask,:].T #TAC: Drop slack instead

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(SC, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.xaxis.set_major_formatter(FormatStrFormatter('%.2f'))

ax.invert_yaxis()

ax.set_title(f'Preprocessed File-{fid} (Surface Only - Dropped Slack)')
ax.set_ylabel('time (s)')
ax.set_xlabel('channel')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.11 Plot Channel (Receiver) Gathers: Surface Only (no slack), and Boreholes Only

Sort By Offset

```
[85]: hset_df['SignSortX'] = np.sign(hset_df['OffsetX']) * hset_df['OffsetMag']
      hset_df['SignSortY'] = np.sign(hset_df['OffsetY']) * hset_df['OffsetMag']
      hset_df['SignSort'] = np.sign(hset_df['OffsetX']) * np.sign(hset_df['OffsetY'])␣
       ↪* hset_df['OffsetMag']
      sorted_df = hset_df.sort_values(by=['RecGatherID', 'SignSort'],␣
       ↪ascending=[True, False])

      sorted_df[['FileIndex','DataIndex','GroupX','GroupY','OffsetX','OffsetY','OffsetMag','RecGathe
```

```
[85]:       FileIndex  DataIndex    GroupX     GroupY  OffsetX  OffsetY  \
      839          94        839  639993.5  5440045.5   -244.2  -1931.2
      840          94        840  639993.5  5440045.5   -244.2  -1931.2
      1307         94       1307  639993.5  5440045.5   -244.2  -1931.2
      1308         94       1308  639993.5  5440045.5   -244.2  -1931.2
      839          96        839  639993.5  5440045.5   -118.1  -1927.8
      …           …         …        …          …        …        …
      2124          0       2124  640426.9  5439659.4  -1281.9   1608.7
      24          115         24  640426.9  5439659.4   1467.3  -1620.6
      25          115         25  640426.9  5439659.4   1467.3  -1620.6
      2123        115       2123  640426.9  5439659.4   1467.3  -1620.6
      2124        115       2124  640426.9  5439659.4   1467.3  -1620.6

              OffsetMag  RecGatherID     SignSort
      839   1946.578301            0  1946.578301
      840   1946.578301            0  1946.578301
      1307  1946.578301            0  1946.578301
      1308  1946.578301            0  1946.578301
      839   1931.414106            0  1931.414106
      …             …          …           …
      2124  2056.984030         1741 -2056.984030
      24    2186.164141         1741 -2186.164141
      25    2186.164141         1741 -2186.164141
      2123  2186.164141         1741 -2186.164141
      2124  2186.164141         1741 -2186.164141

      [805896 rows x 9 columns]
```

Construct Rec Gather Header

```
[86]: rec_id = 0
      rec_df = sorted_df[sorted_df['RecGatherID'] == rec_id]
      print(len(rec_df))
      rec_df[['FileIndex','DataIndex','GroupX','GroupY','OffsetX','OffsetY','OffsetMag','RecGatherII
```

```
     1512
```

```
[86]:       FileIndex  DataIndex    GroupX     GroupY  OffsetX  OffsetY  \
      839          94        839  639993.5  5440045.5   -244.2  -1931.2
```

```
840           94         840  639993.5  5440045.5   -244.2  -1931.2
1307          94        1307  639993.5  5440045.5   -244.2  -1931.2
1308          94        1308  639993.5  5440045.5   -244.2  -1931.2
839           96         839  639993.5  5440045.5   -118.1  -1927.8
…          …        …       …        …        …       …
1308         121        1308  639993.5  5440045.5   1744.2  -1920.0
839          115         839  639993.5  5440045.5   1900.7  -2006.7
840          115         840  639993.5  5440045.5   1900.7  -2006.7
1307         115        1307  639993.5  5440045.5   1900.7  -2006.7
1308         115        1308  639993.5  5440045.5   1900.7  -2006.7

        OffsetMag  RecGatherID  SrcGatherID
839   1946.578301            0           94
840   1946.578301            0           94
1307  1946.578301            0           94
1308  1946.578301            0           94
839   1931.414106            0           96
…            …          …          …
1308  2593.960994            0          121
839   2763.965517            0          115
840   2763.965517            0          115
1307  2763.965517            0          115
1308  2763.965517            0          115

[1512 rows x 9 columns]
```

[87]:
```
# rec_df[['GroupX','GroupY']].drop_duplicates()
rec_df.drop_duplicates(subset=['GroupLat', 'GroupLon'])[['GroupLat',␣
 ↪'GroupLon', 'GroupX', 'GroupY']]
```

[87]:
```
       GroupLat    GroupLon    GroupX     GroupY
839  49.097319 -103.082228  639993.5  5440045.5
```

1.11.1 Show Field Map For Receiver (channel)

[88]:
```
m_width = 1200
m_height = (1/1.66666)*m_width

glst = []
bore_d =␣
 ↪_make_marker_dict(bhdr_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon',color='black',mname='B
bore_g = _make_folium_group_from_dict(bore_d,gname='DAS Boreholes')
src_d =␣
 ↪_make_marker_dict(src_latlon_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SR
src_g = _make_folium_group_from_dict(src_d,gname='All Sources')
rec_d =␣
 ↪_make_marker_dict(rec_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='REC',symbo
rec_g = _make_folium_group_from_dict(rec_d,gname='Channel')

glst.append(rec_g)
glst.append(src_g)
glst.append(bore_g)

clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

base_map = _make_folium_base_map(clat,clon,zoom_start=14)

my_map = _add_folium_groups_to_map(glst,base_map)
```

[89]:

```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

[90]:

```
rgath_map_fname = data_path + '/rec-gather_field_map.html'
print(rgath_map_fname)
```

```
/shared/data/aquistore/rec-gather_field_map.html
```

[91]:

```
# my_map.save(rgath_map_fname)
```

Get Offsets for Plotting

[92]:

```
index_pairs = list(zip(rec_df['FileIndex'], rec_df['DataIndex']))
rec_data = np.array([proc_lst[fid][tid] for fid, tid in index_pairs])
offsets = rec_df['SignSort'].to_numpy()
offsets
```

[92]:

```
array([ 1946.57830051,  1946.57830051,  1946.57830051, …,
       -2763.96551715, -2763.96551715, -2763.96551715], shape=(1512,))
```

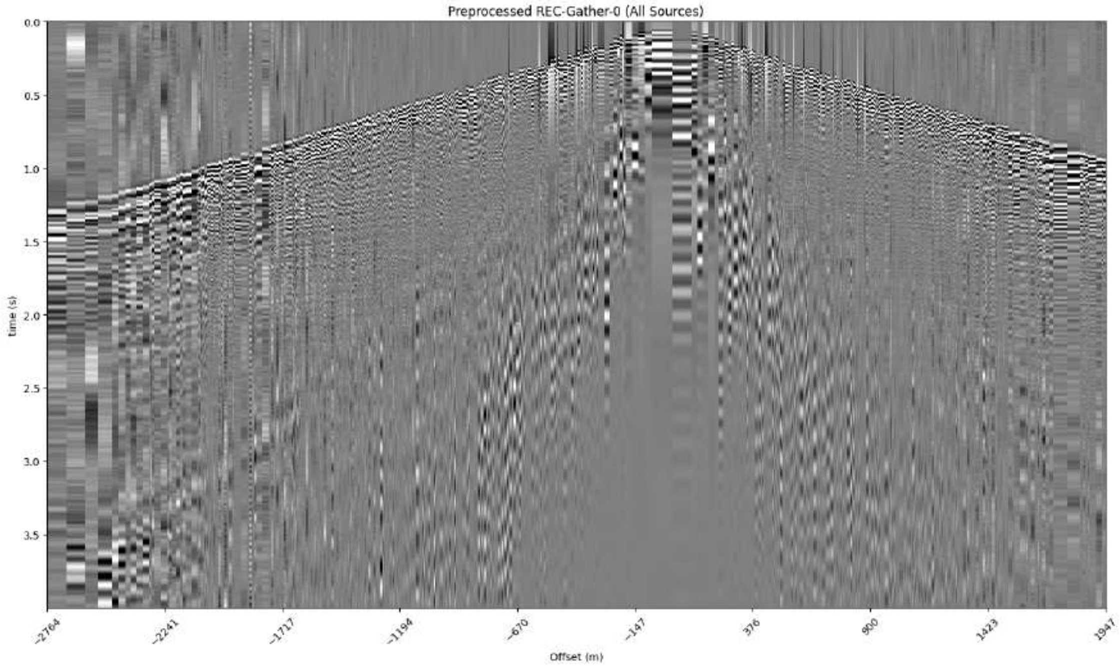
1.11.2 Plot Receiver (channel) Gather

[93]:

```
HH, TT = np.meshgrid(offsets, times)
```

[94]:

```
pclip = 0.8
vmax = (1.-pclip)*np.abs(rec_data).max()
plt_data = rec_data.T

fig, ax = plt.subplots(figsize=(15,9))
im = ax.pcolormesh(HH, TT, plt_data, cmap='gray',vmin=-vmax,vmax=vmax)

xlim = ax.get_xlim()
tick_positions = np.linspace(xlim[0], xlim[1], 10)
ax.set_xticks(tick_positions)
ax.invert_yaxis()

ax.set_title(f'Preprocessed REC-Gather-{rec_id} (All Sources)')
ax.set_ylabel('time (s)')
ax.set_xlabel('Offset (m)')
plt.setp(ax.get_xticklabels(), rotation=45)
plt.tight_layout()
plt.show()
```



1.12 Create Midpoint-Bin Map

1.12.1 Create the Midpoint-Bin Headers

[95]:
```
mid_noslack_df = tagslack_df[~tagslack_df['Is_Slack']].copy()
```

[96]:
```
mbin_size = 50

mid_noslack_df['MidpointBinX'] = mbin_size*np.round((mid_noslack_df['SourceX']␣
  ↪- 0.5*mid_noslack_df['OffsetX'])/mbin_size)
mid_noslack_df['MidpointBinY'] = mbin_size*np.round((mid_noslack_df['SourceY']␣
  ↪- 0.5*mid_noslack_df['OffsetY'])/mbin_size)
```

1.12.2 Calculate Lat-Lon Bins from Easting-Northing Bins

[97]:

```
mid_eastnorth = list(mid_noslack_df[['MidpointBinX', 'MidpointBinY']].
  ↪itertuples(index=False, name=None))

zone_number = 13  # looked up
zone_letter = 'N'

mid_lats,mid_lons = zip(*[utm.to_latlon(e, n, zone_number, zone_letter) for e,␣
  ↪n in mid_eastnorth])
```

1.12.3 Add Midpoint Lat-Lon Bins to Header

[98]:

```
mid_noslack_df['MidpointBinLat'] = mid_lats
mid_noslack_df['MidpointBinLon'] = mid_lons
```

1.12.4 Slice/Filter for Lat-Lon Bin Coordinate Headers

[99]:

```
mid_latlon_df = mid_noslack_df[['MidpointBinLat', 'MidpointBinLon']].
  ↪drop_duplicates()
display(mid_latlon_df)
```

```
      MidpointBinLat  MidpointBinLon
0          49.101011     -103.085422
15         49.101000     -103.084737
30         49.101449     -103.084720
54         49.101899     -103.084703
79         49.102348     -103.084685
…                …              …
777        49.099873     -103.071080
1054       49.094120     -103.076783
1068       49.094132     -103.077467
1078       49.093682     -103.077484
467        49.093671     -103.076800

[1387 rows x 2 columns]
```

1.12.5 Show Field Map with Midpoint Lat-Lon Bins

[100]:

```
m_width = 1200
m_height = (1/1.66666)*m_width

glst = []
src_d =␣
  ↪_make_marker_dict(src_latlon_df,kw_lat='SourceLat',kw_lon='SourceLon',color='red',mname='SR
src_g = _make_folium_group_from_dict(src_d,gname='All Sources')
rec_d =␣
  ↪_make_marker_dict(rec_noslack_df,kw_lat='GroupLat',kw_lon='GroupLon',color='blue',mname='RE
rec_g = _make_folium_group_from_dict(rec_d,gname='Channels')
mid_d =␣
 ↪_make_marker_dict(mid_latlon_df,kw_lat='MidpointBinLat',kw_lon='MidpointBinLon',color='yell
mid_g = _make_folium_group_from_dict(mid_d,gname='Midpoint-Bins')

glst.append(rec_g)
glst.append(src_g)
glst.append(mid_g)

clat,clon = _get_latlot_means(rec_latlon_df,kw_lat='GroupLat',kw_lon='GroupLon')

base_map = _make_folium_base_map(clat,clon,zoom_start=14)

my_map = _add_folium_groups_to_map(glst,base_map)
```

[101]:

```
m_html = get_html_folium_map(my_map,width=m_width,height=m_height)
display(m_html)
```

```
<IPython.core.display.HTML object>
```

[102]:

```
mid_map_fname = data_path + '/midpt-bin_field_map.html'
print(mid_map_fname)
```

```
/shared/data/aquistore/midpt-bin_field_map.html
```

[103]:

```
# my_map.save(mid_map_fname)
```