# AIS Gaps

This notebook uses Datashader to visualize segments in AIS voyages where there are unusually long delays between subsequent pings (ping deltas). Such gaps are common in the dataset and can occur for many reasons.

In [1]:
```
import warnings
import pandas as pd
import numpy as np
import panel as pn
import datetime as dt
import holoviews as hv
import colorcet as cc
from holoviews.util.transform import lon_lat_to_easting_northing as ll2en
from holoviews.operation.datashader import rasterize, dynspread
hv.extension('bokeh')
```

In [2]:
```
warnings.filterwarnings('ignore', category=RuntimeWarning)
```

In [3]:
```
%%time
zone_num=10
basedir = './data/vessel data/Cleaned AIS/Zone10_2014_01/'
broadcast = pd.read_csv(basedir+'Broadcast.csv', parse_dates=[1])
broadcast.head()
```

```
CPU times: user 14.8 s, sys: 1.91 s, total: 16.7 s
Wall time: 16.7 s
```

Out[3]:

| | mmsi_id | date_time | lat | lon | speed_over_ground | course_over_ground | voyage_id | heading | status |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 366025993 | 2013-12-31 23:57:44 | 47.581332 | -122.361145 | 0.0 | 39.599998 | 1 | 511 | 0 |
| **1** | 367160890 | 2013-12-31 23:57:44 | 45.835737 | -123.990592 | 6.7 | 355.399990 | 2 | 359 | 15 |
| **2** | 366490600 | 2013-12-31 23:57:44 | 47.631067 | -122.382117 | 0.0 | 192.100010 | 3 | 180 | 7 |
| **3** | 338000406 | 2013-12-31 23:57:44 | 48.123443 | -123.444115 | 0.0 | 14.200000 | 4 | 511 | 0 |
| **4** | 367840001 | 2013-12-31 23:57:44 | 48.121267 | -122.726412 | 11.4 | 55.400002 | 5 | 57 | 0 |

### Helper functions to generate segments between unusually delayed subsequent AIS pings

In [4]:
```
def close_to_border(slon, elon, zone, border):
    l,r = (-180+6*(zone-1),-180+6*zone)
    lborder, rborder = l+border, r-border
    if (slon < lborder) or (elon < lborder):
        return True
    if (slon > rborder) or (elon > rborder):
        return True
    return False

def voyage_outlier_coords(voyage_dataframe, deviations, zone, border):
    """
    Output eastings and northing 4-tuples [e1, n1, e2, n2] corresponding to
    pings 1 and 2 with a temporal delta greater than the requested
    standard deviations than the mean delta.
    """
    coords = []
    voyage_df = voyage_dataframe.sort_index()
    seconds = np.diff(voyage_df.index).astype(int) / (1.0e9)
    delta = (seconds - seconds.mean()) # Deviation from the mean
    outliers = np.abs(delta) > (seconds.std() * deviations)
    outlier_diff_inds = np.nonzero(outliers)[0]
    if len(outlier_diff_inds) == 0:
        return []
    for diff_ind in outlier_diff_inds:
        start = voyage_df.iloc[diff_ind]
        end = voyage_df.iloc[diff_ind+1]
        slat, slon = start['lat'], start['lon']
        elat, elon = end['lat'], end['lon']
        if close_to_border(slon, elon, zone, border):
            continue
        seasting, snorthing = ll2en(slon, slat)
        eeasting, enorthing = ll2en(elon,elat)
        coords.append((seasting, snorthing, eeasting, enorthing))
    return coords

def voyage_dfs(vessel_df):
    "Given a vessel DataFrame with a single mssid, split by voyage ids"
    voyages = list(vessel_df['voyage_id'].unique())
    return {vid: vessel_df[vessel_df['voyage_id']==vid] for vid in voyages}

def outlier_segments(vessels, zone, deviations=5, border=1, max=100):
    "For all vessel voyages compute segments for ping pairs exceeding deviation"
    coords = []
    for vessel, vessel_df in vessels.items():
        voyages = voyage_dfs(vessel_df)
        for voyage, voyage_df in voyages.items():
            coords += voyage_outlier_coords(voyage_df, deviations=deviations, zone=zone, border=border)
    return hv.Segments(np.array(coords[:max+1]))
```

### Split AIS data by mmsi_id into DataFrames with sorted time index

In [5]:
```
vessels = {name:df.drop_duplicates().sort_values(by='date_time').set_index('date_time')
           for name,df in broadcast.groupby('mmsi_id')}
```

### Projected AIS into WebMercator

In [6]:
```
%%time
broadcast.loc[:,'x'], broadcast.loc[:,'y'] = ll2en(broadcast.lon,broadcast.lat)
```

```
CPU times: user 2.03 s, sys: 1.09 s, total: 3.12 s
Wall time: 3.12 s
```

### Datashade all AIS data as points and unusually delayed pings as segments

Note that when zooming in around Portland, you can see segments crossing land. These mark vessels that generated an AIS ping while out to sea and did not generate another AIS ping until inland.

In [7]:
```
deviations=20 # Segments show ping deltas more than 20 standard deviations from the mean
border=0.5 # Border from UTM boundary in degrees of longitude to filter by
```

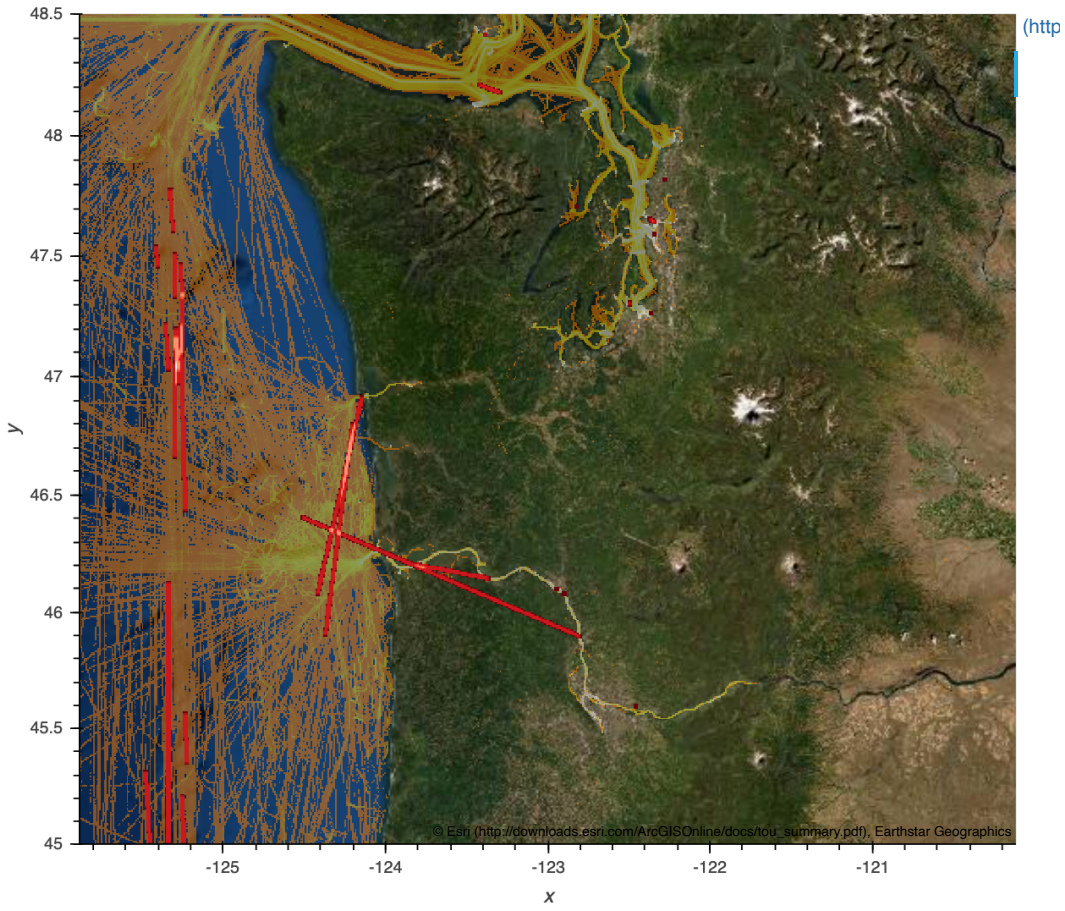
In [8]:
```
def zone(i):
    """
    Return plottable bounds object for a given UTM zone
    (see https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system#UTM_zone)
    """
    lrbt = ((-180+6*(i-1),-180+6*i),(-80,84))
    m    = hv.util.transform.lon_lat_to_easting_northing(*lrbt)
    bnds = hv.Bounds((m[0][0],m[1][0],m[0][1],m[1][1])).opts(color="white")
    text = hv.Text(m[0][0]+(m[0][1]-m[0][0])/2, 0, f"{i}").opts(color="white", text_font_size="5pt")
    return bnds * text
```

In [9]:
```
x_range, y_range = ll2en([-125,-121], [45.0,48.5])

bounds = dict(x=tuple(x_range), y=tuple(y_range))
points = rasterize(hv.Points(broadcast, ['x', 'y'])).redim.range(**bounds)
points = points.opts(cmap=cc.fire[170:], width=700, height=600, cnorm='eq_hist', alpha=0.5)
tiles  = hv.element.tiles.ESRI()
outliers = outlier_segments(vessels, zone_num, deviations=deviations, border=border)

(tiles * points * zone(zone_num) * dynspread(rasterize(outliers)).opts(cmap='Reds_r'))
```

Out[9]:




Having highlighted unusual gaps in this way, code from other notebooks could be added to overlay a selectable ship marker that can provide more detail about that journey and how it compares to others.