In [28]:
```
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.axes as axis
```

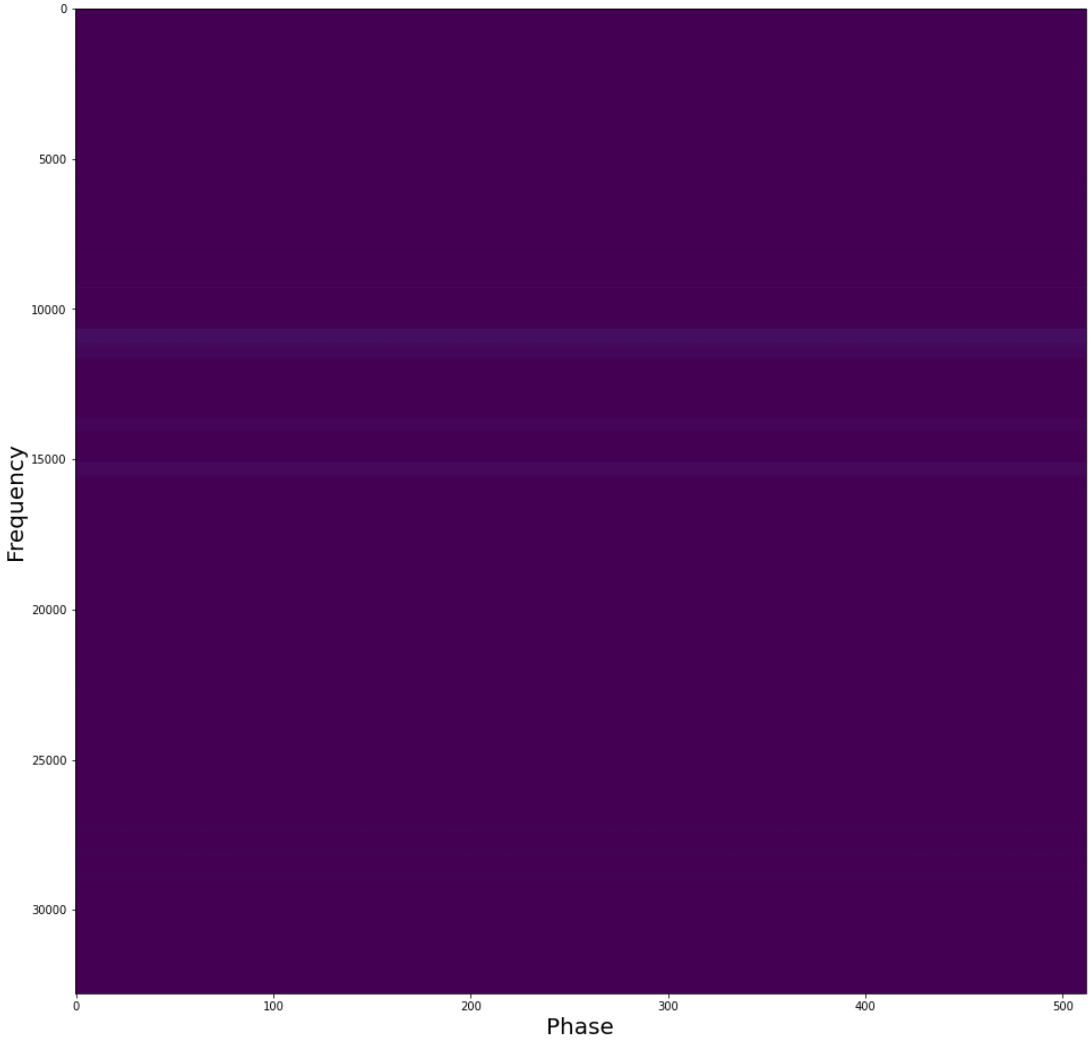
In [2]:
```
def normalize(fold_data,icount_data): #creating function for normalizing fo
lded pulse data

    norm_data = np.zeros_like(fold_data) #initializing array for normalized
data

    for i in range(len(fold_data[:,:,:])): #looping over how ever many itte
rations within the folded data necessary to fill norm_data
        norm_data[:,:,:,i] = fold_data[:,:,:,i]/icount_data[:,:,:] #normali
zing data

    return norm_data
```

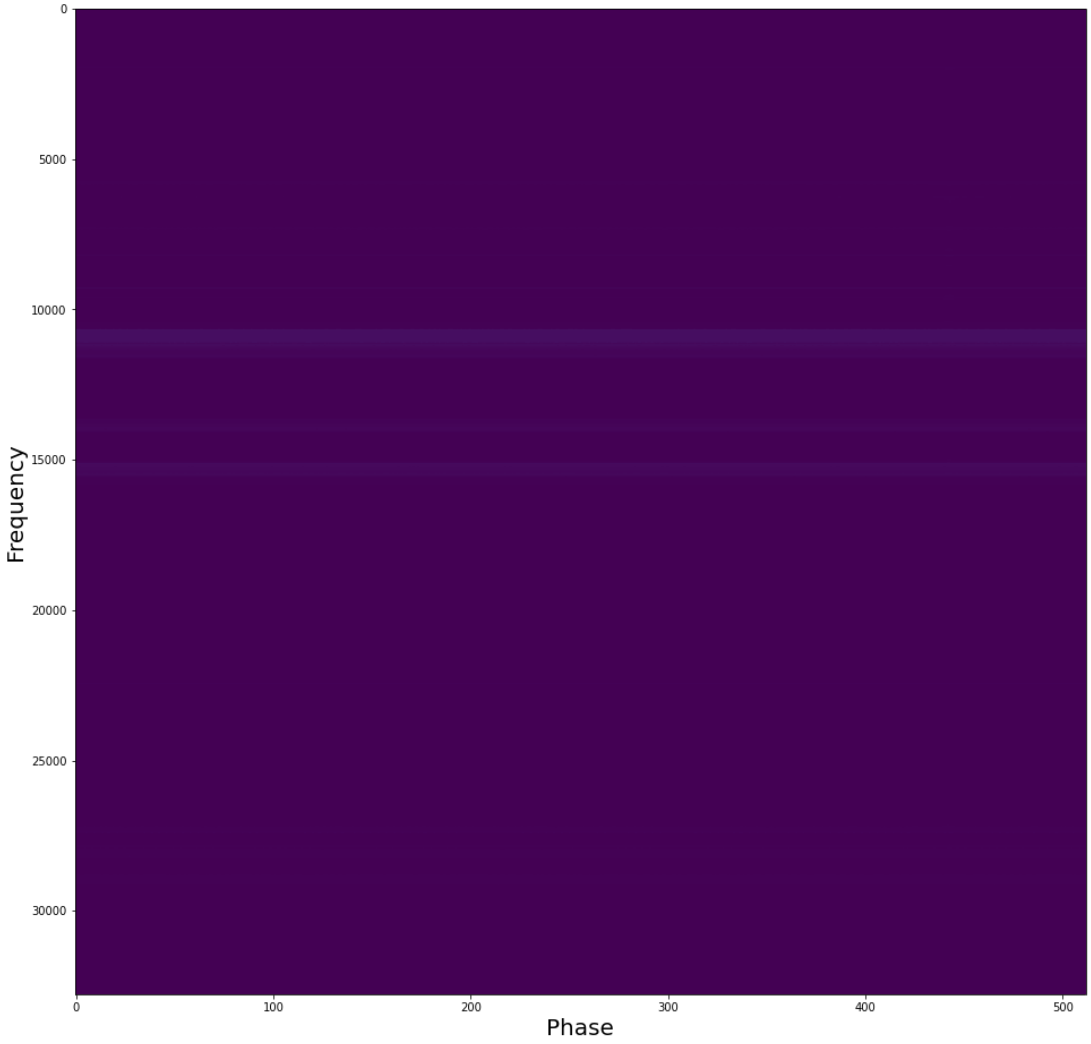
In [3]:
```
def new_norm(fold_data,icount_data): #creating function for normalizing fol
ded pulse data

    norm_data = np.zeros_like(icount_data) #initializing array for normaliz
ed data
    norm_data = fold_data[:,:,:,0]/icount_data[:,:,:] #normalizing data

    return norm_data
```

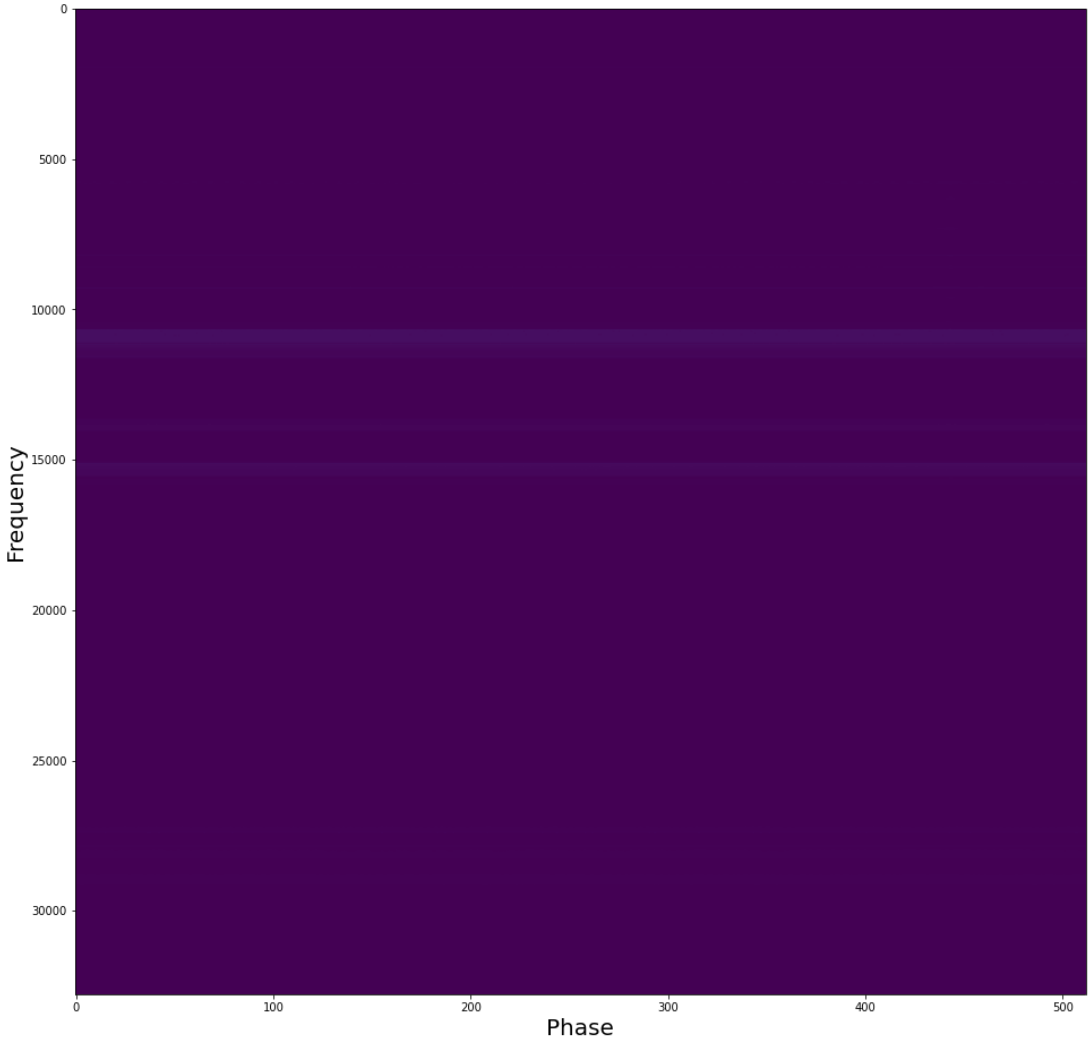
In [4]:
```
start = "arochime-invpfbB0329+54_32768chan3ntbin"
fold = "foldspec_2018-08-16T10:"
icount = "icount_2018-08-16T10:"
end = ".000+30.000000000000004sec"

data1 = np.load(start+fold+str(38)+":"+str(30)+end+".npy")
data2 = np.load(start+icount+str(38)+":"+str(30)+end+".npy")
```

In [56]:
```
normalized_new_data = new_norm(data1,data2)

for i in range(len(data1)):
    plt.figure(figsize=(16,16))
    plt.imshow(normalized_new_data[i,:,:],cmap='viridis',aspect='auto')
    plt.xlabel('Phase', size='20')
    plt.ylabel('Frequency', size='20')
    plt.savefig('figure1.'+str(i)+'.png')
%time
```

```
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 8.58 µs
```

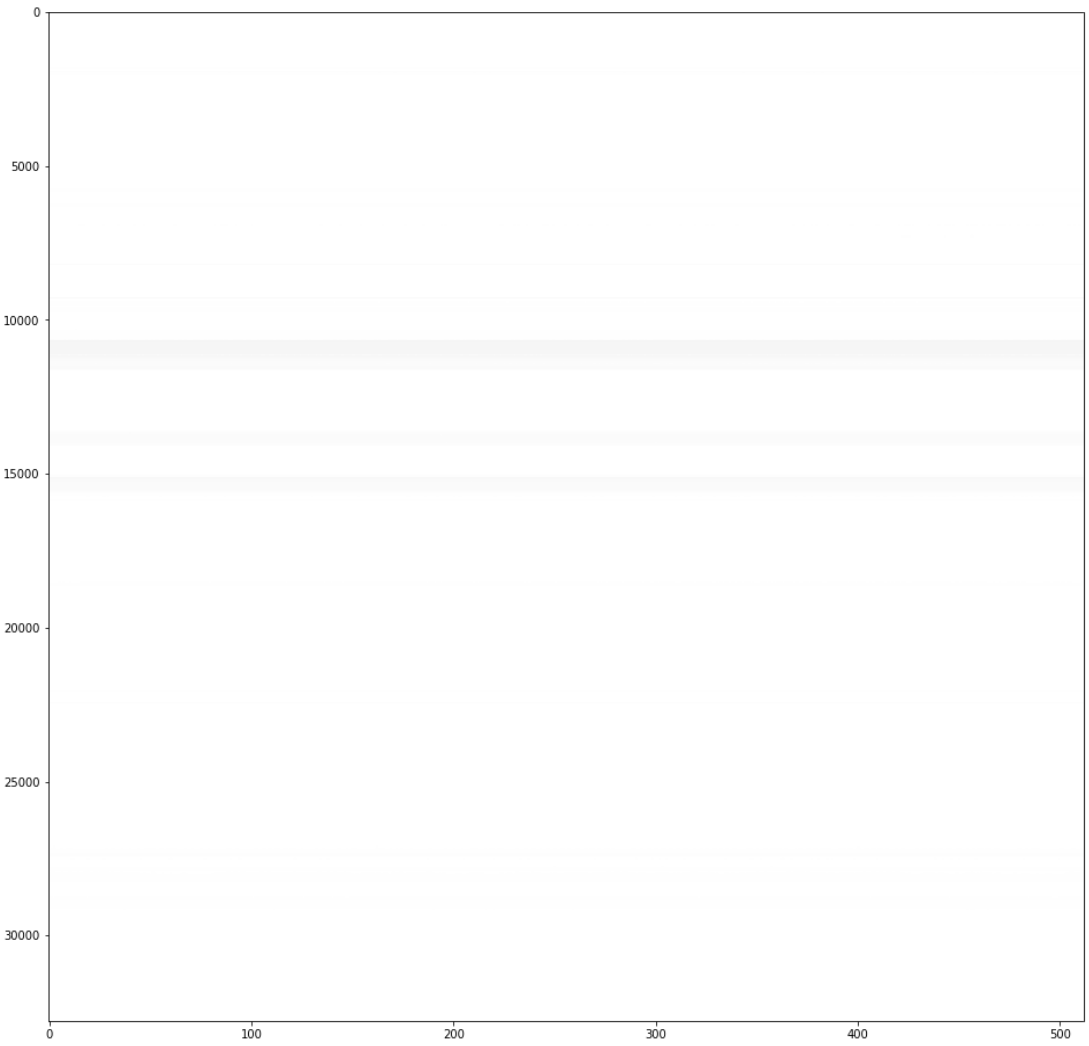



Frequency
0
5000
10000
15000
20000
25000
30000
Phase
0
100
200
300
400
500


Frequency
0
5000
10000
15000
20000
25000
30000
Phase
0
100
200
300
400
500

In [46]:
```
plt.figure(figsize=(16,16))
plt.imshow(normalized_new_data[0,:,:],cmap='binary',aspect='auto')
#help(np.)
#plt.tick_params(axis='x', which='major', labelsize=100)
#plt.xlim(0,32768)
```

Out[46]: <matplotlib.image.AxesImage at 0x7fd5483d6880>



In [3]:
```
start = "arochime-invpfbB0329+54_32768chan3ntbin"
fold = "foldspec_2018-08-16T10:"
icount = "icount_2018-08-16T10:"
end = ".000+30.000000000000004sec"

#final code will look something like:
#need to add plotting line, need to add second for loop for strings with :0
0 instead of :30
# i = 0
# for filename in filenames:
#     fold = np.load(start+fold+str(i+38)+":"+str(30)+end+".npy")
#     count = np.load(start+icount+str(i+38)+":"+str(30)+end+".npy")
#     norm = normalize(fold,count)
#     #plotting line
#     plt.savefig(start+fold+str(i+38)+":"+str(30)+end+".png")
#     i = i+1

test = np.load(start+fold+str(38)+":"+str(30)+end+".npy")
```

In [4]:
```
#what metadata reads:
    #arochime - data from arochime
    #invpfb - something specific to arochime???????
    #B0329+54 - pulsar name
    #32768 - number of entries in the frequency axis
    #chan3t - ??????????
    #foldspec/icount - folded pulse signals or icount data
    #_2018_08-16 - date at which data was taken
    #T - time
    #10:38:30.00 - 10 O'clock and 38 minutes and 30 seconds
    #30.000000000000004sec - data taken over 30 second interval?????????
    #.npy - filetype
data1 = np.load("arochime-invpfbB0329+54_32768chan3ntbinfoldspec_2018-08-16
T10:38:30.000+30.000000000000004sec.npy")
data2 = np.load("arochime-invpfbB0329+54_32768chan3ntbinicount_2018-08-16T1
0:38:30.000+30.000000000000004sec.npy")
data3 = np.load("arochime-invpfbB0329+54_32768chan3ntbinfoldspec_2018-08-16
T10:39:30.000+30.000000000000004sec.npy")
#data4 = np.load("arochime-invpfbB0329+54_32768chan3ntbinicount_2018-08-16T
10:39:00.000+30.000000000000004sec.npy")
```

In [5]:
```
new_data = normalize(data1,data2)
#print(new_data[0,0,:,0]) #phase x
#print(new_data[0,:,0,0]) #freuecy y
#plt.plot(new_data[0,0,:,0],new_data[0,:,0,0])
```

In [6]:
```
ndata = np.zeros_like(data2)
ndata2 = np.zeros_like(data1)

for i in range(len(data1[:,:,:])):
    ndata2[:,:,:,i] = data1[:,:,:,i]/data2[:,:,:]

#print(ndata2)
```

In [ ]:
```
#################### EVERYTHING BELOW IS SCRATCH WORK
############################
```

In [30]:
```
print(Array.shape)
```

```
(2, 32768)
```

In [7]:
```
# Array = np.vstack((data2[0,:,0],data1[0,:,0,0]))
# print(Array.shape)
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
        for k in range(len(data1[0,0,:,0])):
            Array = np.vstack((data2[i,:,j],data1[i,:,k,0]))
            plt.imshow(Array,cmap='viridis')
    #plt.xlim(237,240)
plt.savefig('fig1.1.png')
%time
```

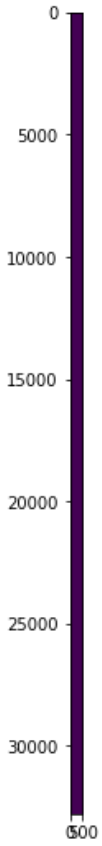
```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-7-d856dba7fcc6> in <module>
      6         for k in range(len(data1[0,0,:,0])):
      7             Array = np.vstack((data2[i,:,j],data1[i,:,k,0]))
----> 8             plt.imshow(Array,cmap='viridis')
      9     #plt.xlim(237,240)
     10 plt.savefig('fig1.1.png')

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/pyplot.py in imsho
w(X, cmap, norm, aspect, interpolation, alpha, vmin, vmax, origin, extent,
shape, filternorm, filterrad, imlim, resample, url, data, **kwargs)
   2650         resample=resample, url=url, **({"data": data} if data is no
t
   2651         None else {}), **kwargs)
-> 2652     sci(__ret)
   2653     return __ret
   2654

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/pyplot.py in sci(i
m)
   3023 @docstring.copy(Axes._sci)
   3024 def sci(im):
-> 3025     return gca()._sci(im)
   3026
   3027

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/axes/_base.py in _
sci(self, im)
   1736             if im.collections[0] not in self.collections:
   1737                 raise ValueError("ContourSet must be in current Axe
s")
-> 1738         elif im not in self.images and im not in self.collections:
   1739             raise ValueError("Argument must be an image, collectio
n, or "
   1740                              "ContourSet in this Axes")

KeyboardInterrupt:

Error in callback <function flush_figures at 0x7f3d2a0f2af0> (for post_exec
ute):
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
/opt/python/3.8.2/lib/python3.8/site-packages/ipykernel/pylab/backend_inlin
e.py in flush_figures()
    119         # ignore the tracking, just draw and close all figures
    120         try:
--> 121             return show(True)
    122         except Exception as e:
    123             # safely show traceback if in IPython, else raise

/opt/python/3.8.2/lib/python3.8/site-packages/ipykernel/pylab/backend_inlin
e.py in show(close, block)
     39     try:
     40         for figure_manager in Gcf.get_all_fig_managers():
---> 41             display(
     42                 figure_manager.canvas.figure,
     43                 metadata=_fetch_figure_metadata(figure_manager.canv
as.figure)

/opt/python/3.8.2/lib/python3.8/site-packages/IPython/core/display.py in di
splay(include, exclude, metadata, transient, display_id, *objs, **kwargs)
    311             publish_display_data(data=obj, metadata=metadata, **kwa
rgs)
    312         else:
--> 313             format_dict, md_dict = format(obj, include=include, exc
lude=exclude)
    314             if not format_dict:
    315                 # nothing to display (e.g. _ipython_display_ took o
ver)

/opt/python/3.8.2/lib/python3.8/site-packages/IPython/core/formatters.py in
format(self, obj, include, exclude)
    178             md = None
    179             try:
--> 180                 data = formatter(obj)
    181             except:
    182                 # FIXME: log the exception

<decorator-gen-9> in __call__(self, obj)

/opt/python/3.8.2/lib/python3.8/site-packages/IPython/core/formatters.py in
catch_format_error(method, self, *args, **kwargs)
    222     """show traceback on failed format call"""
    223     try:
--> 224         r = method(self, *args, **kwargs)
    225     except NotImplementedError:
    226         # don't warn on NotImplementedErrors

/opt/python/3.8.2/lib/python3.8/site-packages/IPython/core/formatters.py in
__call__(self, obj)
    339                 pass
    340             else:
--> 341                 return printer(obj)
    342             # Finally look for special method names
    343             method = get_real_method(obj, self.print_method)

/opt/python/3.8.2/lib/python3.8/site-packages/IPython/core/pylabtools.py in
<lambda>(fig)
    246
    247     if 'png' in formats:
--> 248         png_formatter.for_type(Figure, lambda fig: print_figure(fi
g, 'png', **kwargs))
    249     if 'retina' in formats or 'png2x' in formats:
```

In [ ]:
```
Array = np.vstack((data2[0,:,0],data1[0,:,0,0]))
print(Array.shape)
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
        for k in range(len(data1[0,0,:,0])):
            Array = np.vstack((data2[i,:,j],data1[i,:,k,0]))
            plt.imshow(Array,cmap='viridis')
    #plt.xlim(237,240)
plt.savefig('fig1.1.png')
%time
```

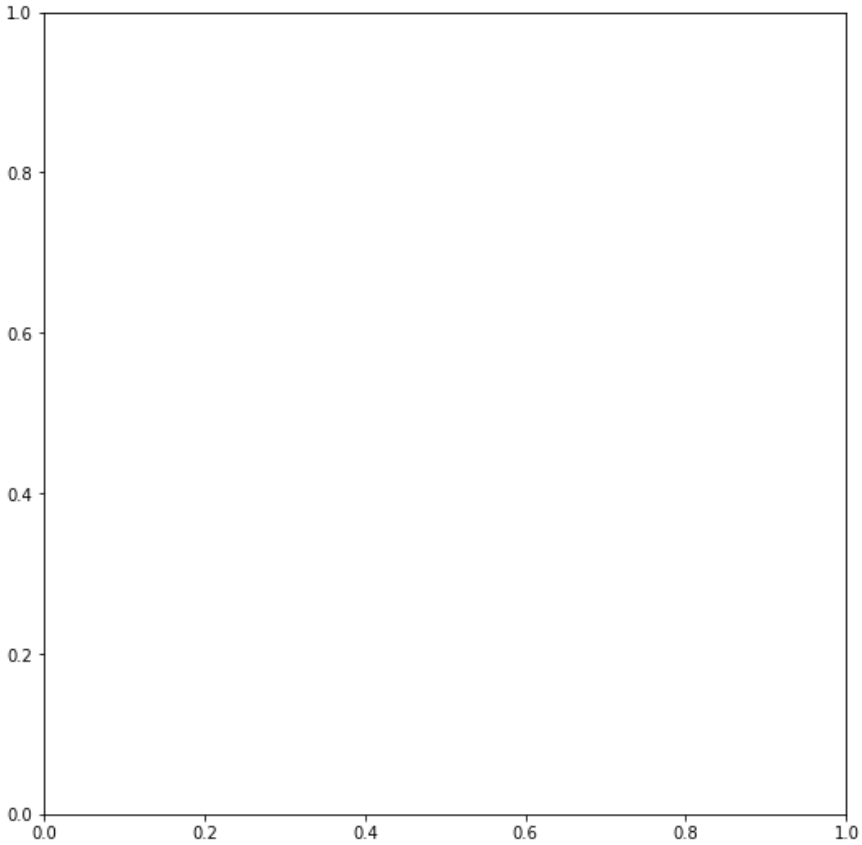
```
(2, 32768)

---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-31-dffa007d7f4a> in <module>
      5     for j in range(len(data1[0,0,:,0])):
      6         for k in range(len(data1[0,0,:,0])):
----> 7             Array = np.vstack((data2[i,:,j],data1[i,:,k,0]))
      8             plt.imshow(Array,cmap='viridis')
      9     #plt.xlim(237,240)

<__array_function__ internals> in vstack(*args, **kwargs)

/opt/python/3.8.2/lib/python3.8/site-packages/numpy/core/shape_base.py in v
stack(tup)
    281     if not isinstance(arrs, list):
    282         arrs = [arrs]
--> 283     return _nx.concatenate(arrs, 0)
    284
    285

<__array_function__ internals> in concatenate(*args, **kwargs)

KeyboardInterrupt:
```

In [27]:
```
new_data = normalize(data1,data2)
X,Y = new_data[0,0,:,0],new_data[0,:,0,0]
plt.figure(figsize=(16,9))
#plt.imshow(new_data[0,1,:,0])
i = 0
j = 0
k = 0
for i in range(len(data1[:,0,0,0])):
    for j in range(len(data1[0,0,0,:])):
        plt.imshow(new_data[i,:,:,j])

# plt.imshow()
```



In [12]:
```
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
        plt.imshow(data2[i,:,j],data1[i,:,j,0])
    #plt.xlim(237,240)
plt.savefig('fig1.1.png')
%time
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-12-cd29c38c8697> in <module>
      2 for i in range(len(data1)):
      3     for j in range(len(data1[0,0,:,0])):
----> 4         plt.imshow(data2[i,:,j],data1[i,:,j,0])
      5     #plt.xlim(237,240)
      6 plt.savefig('fig1.1.png')

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/pyplot.py in imsho
w(X, cmap, norm, aspect, interpolation, alpha, vmin, vmax, origin, extent,
shape, filternorm, filterrad, imlim, resample, url, data, **kwargs)
   2643         filterrad=4.0, imlim=cbook.deprecation._deprecated_paramete
r,
   2644         resample=None, url=None, *, data=None, **kwargs):
-> 2645     __ret = gca().imshow(
   2646         X, cmap=cmap, norm=norm, aspect=aspect,
   2647         interpolation=interpolation, alpha=alpha, vmin=vmin,

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/__init__.py in inn
er(ax, data, *args, **kwargs)
   1563     def inner(ax, *args, data=None, **kwargs):
   1564         if data is None:
-> 1565             return func(ax, *map(sanitize_sequence, args), **kwarg
s)
   1566
   1567         bound = new_sig.bind(ax, *args, **kwargs)

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/cbook/deprecation.
py in wrapper(*args, **kwargs)
    356                 f"%(removal)s.  If any parameter follows {name!r},
they "
    357                 f"should be pass as keyword, not positionally.")
--> 358         return func(*args, **kwargs)
    359
    360     return wrapper

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/cbook/deprecation.
py in wrapper(*args, **kwargs)
    356                 f"%(removal)s.  If any parameter follows {name!r},
they "
    357                 f"should be pass as keyword, not positionally.")
--> 358         return func(*args, **kwargs)
    359
    360     return wrapper

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/axes/_axes.py in i
mshow(self, X, cmap, norm, aspect, interpolation, alpha, vmin, vmax, origi
n, extent, shape, filternorm, filterrad, imlim, resample, url, **kwargs)
   5609             aspect = rcParams['image.aspect']
   5610         self.set_aspect(aspect)
-> 5611         im = mimage.AxesImage(self, cmap, norm, interpolation, orig
in, extent,
   5612                               filternorm=filternorm, filterrad=filt
errad,
   5613                               resample=resample, **kwargs)

/opt/python/3.8.2/lib/python3.8/site-packages/matplotlib/image.py in __init
__(self, ax, cmap, norm, interpolation, origin, extent, filternorm, filterr
ad, resample, **kwargs)
    888         self._extent = extent
    889
--> 890         super().__init__(
```



In [15]:
```
i = 0
for j in range(38,59):
    fold = np.load("arochime-invpfbB0329+54_32768chan3ntbinfoldspec_2018-08
-16T10:"+str(39+i)+":00.000+30.000000000000004sec.npy")
    print(fold.shape)
    i = i+1
```

```
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
```

In [13]:
```
i = 0
for j in range(38,59):
    fold = np.load("arochime-invpfbB0329+54_32768chan3ntbinfoldspec_2018-08
-16T10:"+str(38+i)+":30.000+30.000000000000004sec.npy")
    print(fold.shape)
    i = i+1
```

```
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
(3, 32768, 512, 4)
```

In [8]:
```
data3.shape
```

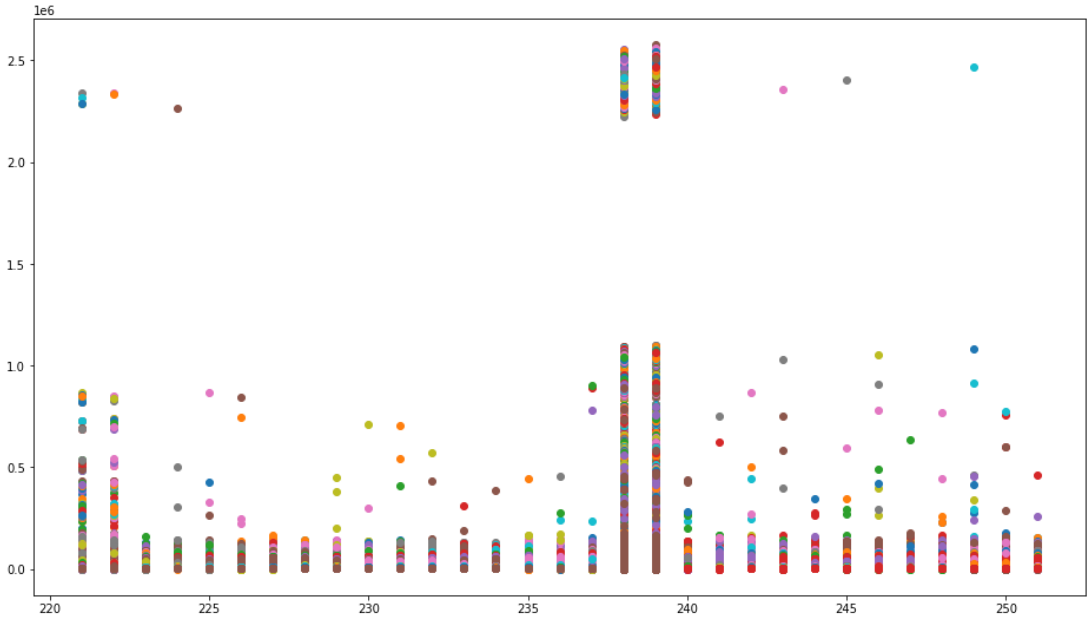
Out[8]:
```
(3, 32768, 512, 4)
```

In [45]:
```
len(data1)
print(data1[0,:,0,0])
#print(data1[0,0,:,0])
print(data1[:,0,1,0])
```

```
[20.894543 22.04368  23.128796 ... 35.283733 35.07841  34.861423]
[19.661644 20.953207 16.106157]
```

In [70]:
```
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
            plt.plot(data2[i,:,j],data1[i,:,j,0], 'o')
    #plt.xlim(237,240)
plt.savefig('fig1.png')
%time
```

```
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 10.3 µs
```



In [83]:
```
%time
#test1 = np.zeros_like(data1)
interm = np.zeros_like(data1)
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
        test1[i,:,j,0] = test1[i,:,j,0]+data1[i,:,j,0]
    #plt.xlim(237,240)
```

```
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 19.1 µs

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-83-d599308a1be0> in <module>
      4 for i in range(len(data1)):
      5     for j in range(len(data1[0,0,:,0])):
----> 6         test1e[i,:,j,0] = test1e[i,:,j,0]+data1[i,:,j,0]
      7     #plt.xlim(237,240)

NameError: name 'test1e' is not defined
```

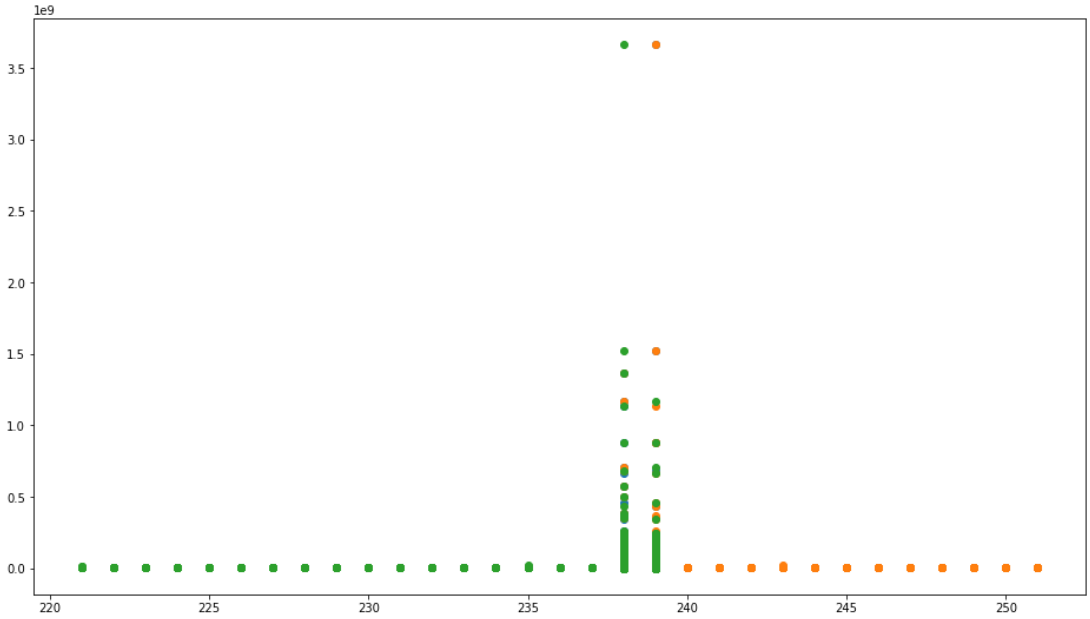
In [86]:
```
%time
#test1 = np.zeros_like(data1)
test_01 = np.zeros(len(data1[0,:,0,0]))
interm = np.zeros_like(data1)
for i in range(len(data1)):
    for j in range(len(data1[0,0,:,0])):
        test_01 = test_01+data1[i,:,j,0]
    #plt.xlim(237,240)
```

```
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 11.9 µs
```

Out[86]:
```
32768
```

In [89]:
```
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    plt.plot(data2[i,:,0],test_01, 'o')
plt.savefig('fig2.png')
%time
```

```
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 9.78 µs
```



In [82]:
```
print(len(test1[0,0,:,0]))
```

```
512
```

In [ ]:
```
plt.figure(figsize=(16,9))
for i in range(len(data1)):
    plt.plot(data2[i,:,j],test1[i,:,j,0], 'o')
plt.savefig('fig2.png')
%time
```