Exploration of the Anonymous videos on Youtube.

How the initial data was generated - Davide's dataset

Here's what Davide wrote in his email:

> The query: "anonymous internet freedom"

Some notes: I have just discovered that some changes have been made to YT api, so the script was not able to retreive keywords associated with videos (that can be interesting for semantic network analysis purposed) and that videos descriptions are truncated. I will try to figure out whether is possible to fix the first shortcome; for the second, luckly the "related videos" descriptions are still complete..and usually there is wide redundancy, so maybe it can be easily overcame (consider that in this sample I restricted related videos to 25 for each); the comments are also in this case limited to 25; videos are ordered by relevance computed by Youtube itself; users are limited to original content submitters; the string I input as keyword was only "anonymous internet freedom".

(1) retrieve up to 1000 videos queried by keywords (including username,date,category,n of favorites, n of views,duration, tags and description) -maybe it is possible to extend the 1000 limit, or by the way it can be somehow done with the following module-

(2) reconstruct the network of "related videos" for each video retrieved (with the metadata above) -the alghoritm for matching related videos is of course not known, but as far as I understood it should be both related to tags and to users' behaviour: the more two videos are viewed in sequence, the more related they'll be... so it can provide interesting insights on network of content-

(3) scrape comments related to the videos/related videos retrieved (including user name, time and parent comments) The geo info are quite tricky cause there is no "nationality" associated to the video, but the language...at least not provided with the api! But maybe something can be inferred with a heuristic approach (nations mentioned in the title/description/comments)

Generating the data from Youtube API -- Adrian's dataset

Davide's dataset is really useful, but I found it hard to replicate the same results. The scripts I wrote to query the YT api bring back different results. This needs to be investigated. Also because Google/Youtube only returns a maximum of 1000 results, we need to think of ways of 'tiling' the queries to get around this limitation. At least, so that we can be reasonably satisfied we are getting all the relevant videos.

In [68]:
```
%load_ext autoreload
%autoreload 2
```

In [69]:
```
import pandas as pd
import YT_api_generate as yt
from pylab import *
from IPython.display import HTML
from IPython.display import YouTubeVideo
pd.set_option("display.max_columns", 6)
pd.set_option("display.max_rows", 15)
pd.set_option("display.notebook_repr_html", True)
from optparse import OptionParser
```

In [7]:
```
an_f = pd.ExcelFile('data/anonymouys internet freedom - VIDEOS.xls')
an_dict = {name:an_f.parse(name) for name in an_f.sheet_names}
video_df = an_dict['VIDEOS']
#clean up data a bit
video_df.VIEWS = video_df.VIEWS.replace('None', 0)
print(video_df.shape)
print(video_df.head(10))
```

...

The dataset has about a 1000 rows, but many seem duplicates. That is, the rows are literally identical in every field So cutting those out: ...

In [17]:
```
video_df = video_df.drop_duplicates()
print(video_df.shape)
print('Users: ', len(video_df.USER.unique()))
print('Titles: ', len(video_df.TITLE.str.encode('utf-8').unique()))
```

```
(289, 11)
('Users: ', 246)
('Titles: ', 283)
```

There are 908 titles in the Beraldo dataset, of which **283** are unique. There are 246 users -- people (robots?) who upload videos.

Comparing Davide's results with the ones I get from the Youtube API:

In [56]:
```
df_am = yt.youtube_search(query='anonymous,internet,freedom', max_results=1000, with_statistics=True)
```

...

In [57]:
```
df_am.columns
df_am.drop_duplicates(inplace=True, cols='videoId')
print(df_am.shape)
print('Titles: ', len(df_am.title))
```

```
(376, 20)
('Titles: ', 376)
```

So not a huge difference in numbers -- 283 vs 366. But are they the same videos more or less?

In [58]:
```
davide_set = set(video_df.ID.tolist())
adrian_set = set(df_am.videoId.tolist())
comb_id = davide_set.union(adrian_set)
intersecting_id = adrian_set.intersection(davide_set)
adrian_unique = adrian_set.difference(davide_set)
davide_unique = davide_set.difference(adrian_set)
print('Set of combined video ids: ' + str(len(comb_id)))
print('They have ' + str(len(intersecting_id)) + ' in common')
print('Davide datatset has ' + str(len(davide_unique)) + ' that Adrian doesnt')
print('Adrian has ' + str(len(adrian_unique)) + ' that Davide doesnt')
```

```
Set of combined video ids: 534
They have 131 in common
Davide datatset has 158 that Adrian doesnt
Adrian has 245 that Davide doesnt
```

So it looks like there is in **poor overlap** between them. Altogether there are 525 unique video ids, but 25% are common to both. The rest are different. How to explain this? If Davide used the Youtube API like I did, does this mean that Google is giving different results to different people? Or that the results depend on when/where you run the query? The question of getting the same data is maybe only an interesting methodological wrinkle, or it might be something worth investigating in its own right.

I'll leave this aside for the moment, and just focus on Davide's dataset.

The uniqueness of the videos

But once all the dupicates are removed, it seems we are left with a unique list of 283 videos, and they are not mirrored at all. Here are the mirrored videos:

In [60]:
```
vc=video_df.TITLE.value_counts()
vc[vc>1];
```

That's it in Davide's dataset: 3 copies of 'Anonymous Declaration of Freedom,' and 2 of the rest. **5** mirrored videos, and not highly mirrored.

To check the same in Adrian's dataset -- are they the duplicated ones there too?

In [61]:
```
vc2 = df_am.title.value_counts()
vc2[vc2>1]
```

Out[61]:
```
Anonymous - Message on PRISM & Edward Snowden                  6
Anonymous Declaration of Freedom                               3
Anonymous - A Message to All The Freedom Fighters of The World   2
Anonymous promises to shake up Internet in 2013                2
'ACTA not over, people await Internet revolution'              2
Anonymous vs NorthKorea                                        2
Anonymous Philippines message to Cybercrime Law supporters     2
Anonymous   Message on PRISM & Edward Snowden                  2
dtype: int64
```

Here there is a bit more duplication, and some overlap. There are also some time-specific results -- PRISM and Snowden probably arrives after Davide has generated his dataset. But still only 8 mirrored videos.

I guess we should discuss this Adam. Should we look for more hidden duplicates? Is it something to do with the youtube search? Are there other ways of identifying videos associated with Anonymous that need to look into.

The length of the videos

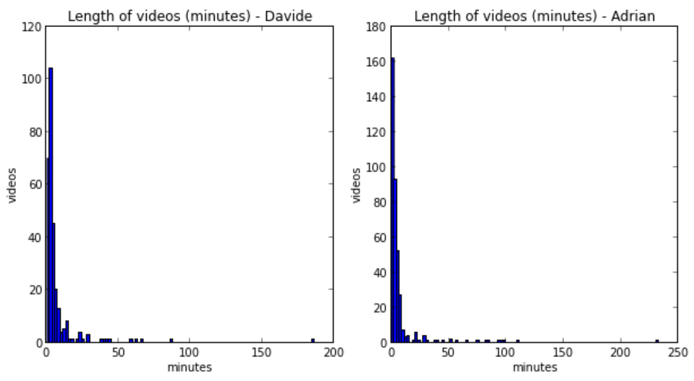
Not sure what the length of videos tells us. Something about the kind of object we are dealing with. Also, I guess length affects mirroring. Would many people upload a 60 minute documentary?

In [104]:
```
#davide's videos
duration = video_df.DURATION.order().tolist()
duration = [d/60 for d in duration]
f = pylab.figure(figsize=(10,5))
sp=f.add_subplot(1,2,1)

h=sp.hist(duration, bins=100)
print('Less than 10 minutes: '+ str(float(sum(video_df.DURATION/60 < 10))/video_df.DURATION.shape[0]*100)+ '%')
sp.set_title('Length of videos (minutes) - Davide')
sp.set_xlabel('minutes')
sp.set_ylabel('videos')
#adrian's videos
df_am = yt.format_durations(df_am)
df_am.columns
#df_am['duration_second'].hist(bins=100)
sp2=f.add_subplot(1,2,2)
sp2.set_title('Length of videos (minutes) - Adrian')
sp2.set_xlabel('minutes')
sp2.set_ylabel('videos')
dur_min = df_am.duration_second/60
h2 = sp2.hist(dur_min, bins=100)
YouTubeVideo(video_df.ID.iloc[1])
```

```
Less than 10 minutes: 87.5432525952%
```

Out[104]:



Again, the differences between the datasets. But can't see any pattern here.

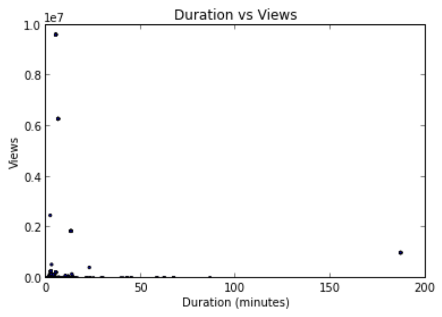
In general, it looks like most of the videos are relatively short (less than 10 minutes). Does duration correlate with either viewing or duplication?

In [8]:
```
fig=figure(dpi=200)
sp = fig.add_subplot(1,2,1)
sp.scatter(video_df.DURATION/60, video_df.VIEWS, s=5,marker='o')
sp.set_title('Duration vs Views')
sp.set_xlabel('Duration (minutes)')
sp.set_ylabel('Views')
sp.set_xlim(0, 200)
sp.set_ylim(0, 1e7)

hms =[re.sub('H|M', ':', s) for s in [re.sub('PT|S', '', t)   for t in df_am.duration]]

sp = fig.add_subplot(1,2,2)
sp.scatter(video_df.DURATION/60, video_df.VIEWS, s=5,marker='o')
sp.set_title('Duration vs Views')
sp.set_xlabel('Duration (minutes)')
sp.set_ylabel('Views')
sp.set_xlim(0, 200)
sp.set_ylim(0, 1e7)
```

Out[8]:
```
(0, 10000000.0)
```



At a glance, it looks like the longer videos are viewed much less. Very short videos are not viewed much either. But this doesn't take into account the fact that there are many more short videos.

Viewing figures

Another way to look at videos -- more in terms of reception. Does anything stand out from the viewing figures?

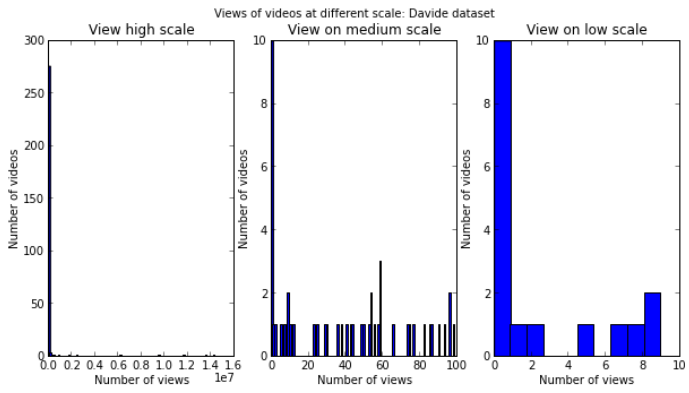
As usually, totally different results from the Adrian dataset -- maximum view is 66 million, not 14 million

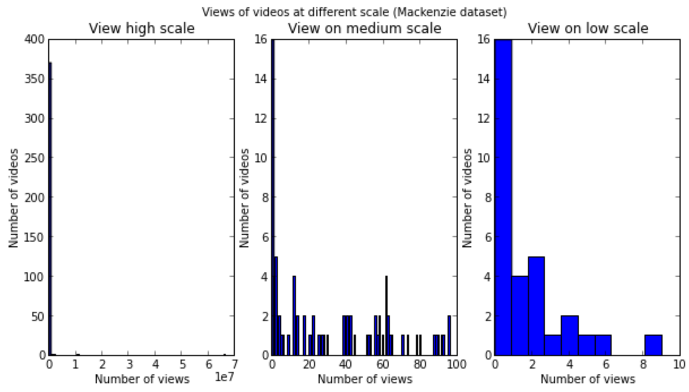
In [101]:
```
video_df.VIEWS = video_df.VIEWS.replace('None', 0)
views = video_df.VIEWS.order()
print('Maximum views: ' + str(views.max()))
f=figure(figsize(10,5))
suptitle('Views of videos at different scale: Davide dataset')
sp1 = f.add_subplot(1,3,1)
h1=sp1.hist(views, bins=100)
sp1.set_ylabel('Number of videos')
sp1.set_xlabel('Number of views')
sp1.set_title('View high scale')
sp2 = f.add_subplot(1,3,2)
h2 = sp2.hist(views[views<100], bins=100)
sp2.set_title('View on medium scale')
sp2.set_ylabel('Number of videos')
sp2.set_xlabel('Number of views')
sp3 = f.add_subplot(1,3,3)
h3 = sp3.hist(views[views<10], bins = 10)
sp3.set_title('View on low scale')
sp3.set_ylabel('Number of videos')
sp3.set_xlabel('Number of views')

#Adrian dataset
views = df_am.viewCount.astype('int').order()
print('Max views in adrian dataset: ' + str(views.max()))
f=figure(figsize=(10,5))
suptitle('Views of videos at different scale (Mackenzie dataset)')
sp1 = f.add_subplot(1,3,1)
h1=sp1.hist(views, bins=100)
sp1.set_ylabel('Number of videos')
sp1.set_xlabel('Number of views')
sp1.set_title('View high scale')
sp2 = f.add_subplot(1,3,2)
h2 = sp2.hist(views[views<100], bins=100)
sp2.set_title('View on medium scale')
sp2.set_ylabel('Number of videos')
sp2.set_xlabel('Number of views')
sp3 = f.add_subplot(1,3,3)
h3 = sp3.hist(views[views<10])
sp3.set_title('View on low scale')
sp3.set_ylabel('Number of videos')
sp3.set_xlabel('Number of views')
```

```
Maximum views: 14381492.0
Max views in adrian dataset: 66601564
```

Out[101]:
```
<matplotlib.text.Text at 0x34d5b110>
```





As usual, most videos are only viewed by a small number of people, and a couple are viewed by 100,000s up to 14 million. Actually, viewing figures are quite widely distributed even for the less-viewed videos. It might be interesting to look at the videos that are viewed 8, 30 or 80 times as well as the ones viewed 14 millon times. This analysis *does not* take into account view figures for copies of videos.

The most viewed videos

If we aggregate views for videos with the same title, things start to look different.

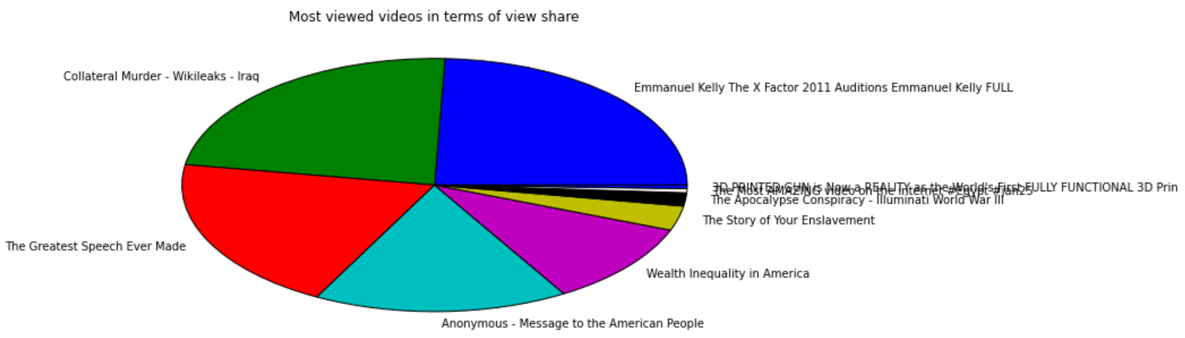
In [11]:
```
top_views = video_df.loc[video_df.VIEWS>100000, ['TITLE', 'VIEWS']]
gby=top_views.groupby(by='TITLE')['VIEWS']
print('total viewings:'+str(video_df['VIEWS'].sum()))
print('total views of top videos:', sum(gby.sum()))
print('total views of top 5 videos:', sum(gby.sum()[:4]))

print(gby.ngroups)
gbyo=gby.sum().order(ascending=False)
pie(gbyo[0:9], labels=gbyo.keys()[0:9])
pylab.title('Most viewed videos in terms of view share')
pd.DataFrame(gbyo).head(n=5)
```

```
total viewings:480078782.0
('total views of top videos:', 477150437.0)
('total views of top 5 videos:', 4606329.0)
18
```

Out[11]:

	VIEWS
TITLE	
Emmanuel Kelly The X Factor 2011 Auditions Emmanuel Kelly FULL	115051936
Collateral Murder - Wikileaks - Iraq	109121536
The Greatest Speech Ever Made	93953440
Anonymous - Message to the American People	76903592
Wealth Inequality in America	50298512



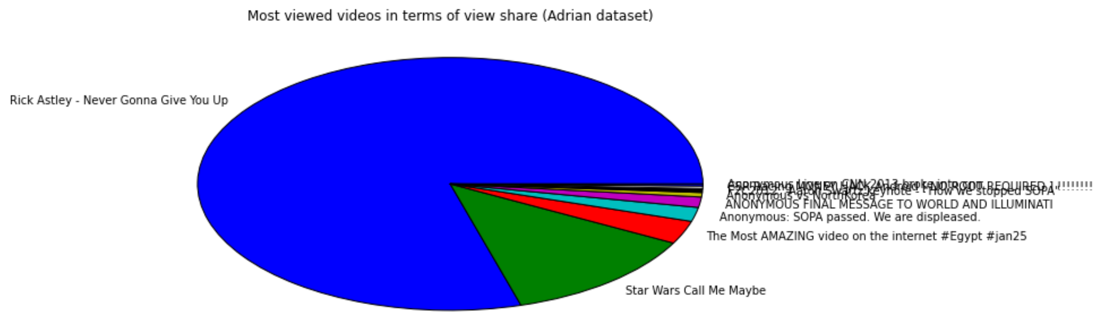
In [103]:
```
# for the Adrian dataset
df_am.viewCount = df_am.viewCount.astype('int32')
top_views_am = df_am.loc[df_am.viewCount>100000, ['title', 'viewCount']]
gby_am=top_views_am.groupby(by='title')['viewCount']
print('total viewings:'+str(df_am['viewCount'].sum()))
print('total views of top videos:', sum(gby_am.sum()))
print('total views of top 5 videos:', sum(gby_am.sum()[:4]))

print(gby_am.ngroups)
gbyo_am=gby_am.sum().order(ascending=False)
pie(gbyo_am[0:9], labels=gbyo_am.keys()[0:9])
pylab.title('Most viewed videos in terms of view share (Adrian dataset)')
pd.DataFrame(gbyo_am).head(n=5)
```

```
total viewings:87020971
('total views of top videos:', 84911040)
('total views of top 5 videos:', 1458506)
18
```

Out[103]:

	viewCount
title	
Rick Astley - Never Gonna Give You Up	66601564
Star Wars Call Me Maybe	10658788
The Most AMAZING video on the internet #Egypt #jan25	2480617
Anonymous: SOPA passed. We are displeased.	1515428
ANONYMOUS FINAL MESSAGE TO WORLD AND ILLUMINATI	1114730



There are 18 videos that each have more 100,00 views. There have been around 480 million views in total of Anonymous-related videos. The top 18 videos account for 477 million of the views. The top 5 videos account for 460 million of the 480 million. Looking at the titles, is the 'Emmanuel Kelly X Factor 2011 Auditions' video 'noise'? Maybe not -- he's an Iraqi orphan living in Australia, and 'Collateral Murder - Wikileaks - Iraq' (#2) doesn't look like that. The top four videos after Kelly's all have more than 50 million viewings. So they dominate the total views of Anonymous videos.

In [12]:
```
#video_df.TITLE.
emmanuel_kelly = video_df.ID[video_df.TITLE == gbyo.index[0]]
YouTubeVideo(emmanuel_kelly.iloc[0])
```

Out[12]:

Are video duplicates viewed?

Are views related to duplication/mirroring practices? First, we need to find the duplicates. Title is an ok starting point. Presumably, videos with the same title are likely to be duplicates.

In [13]:
```
title = video_df.TITLE.tolist()
video_df.TITLE = [t.encode('utf8', errors='ignore') for t in title]
print('distinct titles:', len(video_df.TITLE.unique()))
print('distinct ids', len(video_df.ID.unique()))
print('distinct durations:', len(video_df.DURATION.unique()))
```

```
('distinct titles:', 283)
('distinct ids', 289)
('distinct durations:', 221)
```

While there are 908 videos listed, there are only 283 distinctly titled videos and 289 distinct video IDS. That means that potentially ~620 are copies. But if copies have the same ID, does that mean that they have been uploaded? Or are they simply different views on the same video? This is a crucial question - are we dealing here with duplicates or just references to one instance?
Things are a bit more complicated in the duration data. There are only 220 distinct durations. But it could be that different videos happen to have the same duration. What are the most commonly duplicated videos?

In [14]:
```
dup_video_counts =video_df.TITLE.value_counts()
dup_df = pd.DataFrame(dup_video_counts.head())
dup_df
```

Out[14]:

	0
Christopher Hitchens Wishful Thinking vs Evidence 2008)	16
Anonymous is Freedom	15
Anonymous Declaration of Freedom	10
Anonymous - Truth is a Virus	10
Anonymous Exposed for Crimes Against Humanity	9

In [15]:
```
top_dup_video_counts=dup_video_counts[dup_video_counts>5]
top_dup_video_counts = top_dup_video_counts.order(ascending=False)
top_dup_videos = dup_video_counts.index
top_dup_video_counts
```

...

Does duplication correlate with viewing figures? Do these duplicates get the same number of views? Or are some versions much more widely viewed than others?

In [16]:
```
dup_views=video_df.loc[video_df.TITLE.isin(top_dup_videos), ['TITLE', 'VIEWS']]
dup_views['copies'] = top_dup_video_counts
dv =pd.DataFrame(dup_views.groupby(['TITLE'])['VIEWS'].sum().order(ascending=False))
dv.head()
```

Out[16]:

	VIEWS
TITLE	
Emmanuel Kelly The X Factor 2011 Auditions Emmanuel Kelly FULL	115051936
Collateral Murder - Wikileaks - Iraq	109121536
The Greatest Speech Ever Made	93953440
Anonymous - Message to the American People	76903592
Wealth Inequality in America	50298512

Some interesting points here. High rates of duplication does not equate with high view counts. What is the Christopher Hitchens piece? It is the 4th most duplicated video, but shows 0 views - is that possible? More importantly, the most duplicated video 'Anonymous - Truth is a Virus' only has 1380 views.

Some stuff on users

In [17]:
```
video_df.columns
video_df['USER'].unique().shape
```

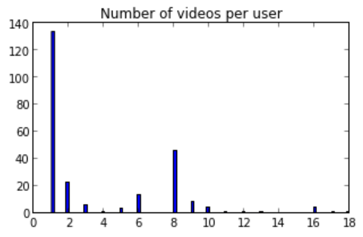
Out[17]:
```
(246,)
```

So there are 246 users (people?) who upload videos. How much do they upload?

In [18]:
```
user_counts = video_df['USER'].value_counts()
f3 = figure(figsize=(5,3))
sp=f3.add_subplot(111)
h4 = sp.hist(video_df['USER'].value_counts(), bins=100)
sp.set_title('Number of videos per user')
top_users = user_counts[user_counts>5].index
top_users
print(user_counts.describe())
user_counts.quantile(0.6)
```

```
count    246.000000
mean       3.691057
std        3.804680
min        1.000000
25%        1.000000
50%        1.000000
75%        8.000000
max       18.000000
dtype: float64
```

Out[18]:
```
2
```



So, more than 60% of people upload less than 2 videos.

How top contributing users upload top-viewed videos -- that is, do they duplicate them?

I'm calling 'top users' anyone who uploads more than 5 videos, and 'top videos' any video that is duplicated more than 5 times.

In [32]:
```
topuser_toptitle=pd.crosstab(video_df[video_df.USER.isin(top_users)]['USER'], video_df[video_df.TITLE.isin(top_dup_videos)
top_user_title_df = pd.DataFrame(topuser_toptitle.sum(axis=1).order(ascending=False))
top_user_title_df.head()
```

Out[32]:

	0
USER	
YourRightsVigilante	18
Truthloader	17
wikispeak10	16
Tom Craig	16
MatriXCr3w	16

This suggests that there is some link between between uploading a lot and duplicating a lot. I'm not sure about this actually -- needs further thought. There is still the question of whether the duplicates are viewed a lot. And do these high duplicating users duplicate the same videos?

What is viewed

Many questions could be asked here. For highly duplicated videos, are all copies viewed at similar levels or not?

In [39]:
```
date = pd.to_datetime(video_df.DATA)
video_df.DATA = date
video_df.groupby(by=['TITLE'])['VIEWS'].sum().order()
```

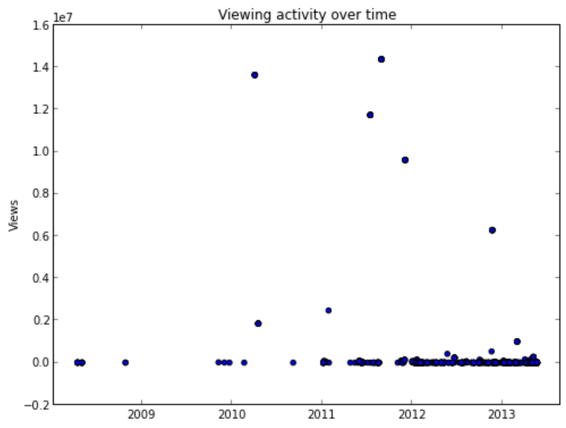
...

In [28]:
```
f = figure(figsize=(8,6), dpi=400)
s = f.add_subplot(111)
s.scatter(date, video_df.VIEWS)
s.set_title('Viewing activity over time')
s.set_ylabel('Views')

top_views.head()
```

Out[28]:

	TITLE	VIEWS
21	Anonymous Philippines message to Cybercrime La...	143349
45	Table Talk: Cons, Internet Freedom, & Spoiler ...	155988
58	Anonymous Calls for CISPA Internet Blackout Day	138356
60	F2C2012: Aaron Swartz keynote - "How we stopp...	415163
76	Anonymous - A Message to Congress on SOPA	101092

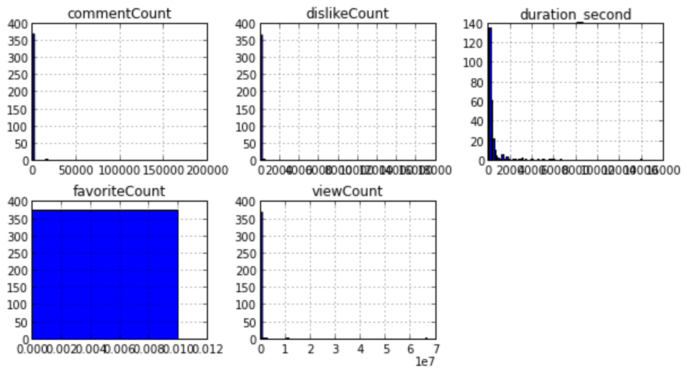


In [ ]:
```
## Overviews of the data
```

In [98]:
```
df_am.columns
# df_am.commentCount.astype('int64').hist()

df_am[['commentCount', 'dislikeCount', 'favoriteCount', 'viewCount', 'duration_second']].astype('int64').hist(bins=100)
```

Out[98]:
```
array([[Axes(0.125,0.563043;0.215278x0.336957),
        Axes(0.404861,0.563043;0.215278x0.336957),
        Axes(0.684722,0.563043;0.215278x0.336957)],
       [Axes(0.125,0.125;0.215278x0.336957),
        Axes(0.404861,0.125;0.215278x0.336957),
        Axes(0.684722,0.125;0.215278x0.336957)]], dtype=object)
```



Hard to see what is happening in the viewing of these videos.