Overview and Motivation:

High-throughput screening (HTS) is a method utilizing robotics and biology experiments for fast drug discovery. There are a great number of drugs discovered from HTS. Below is a table of drugs approved by FDA before 2010.

Table 2 | **Examples of recently approved drugs with origins in HTS hits***

Drug (US trade name; company)	Indication	Target class	Year HTS was run	Year of FDA approval	Ref.
Gefitinib (Iressa; AstraZeneca)	Cancer	Tyrosine kinase	c. 1993	2003	57
Erlotinib (Tarceva; Roche)	Cancer	Tyrosine kinase	c. 1993	2004	57
Sorafenib (Nexavar; Bayer/Onyx Pharmaceuticals)	Cancer	Tyrosine kinase	1994	2005	58
Tipranavir (Aptivus; Boehringer Ingelheim)	HIV	Protease	c. 1993	2005	59
Sitagliptin (Januvia; Merck & Co)	Diabetes	Protease	c. 2000	2006	60
Dasatinib (Sprycel; Bristol-Myers Squibb)	Cancer	Tyrosine kinase	1997	2006	61
Maraviroc (Selzentry; Pfizer)	HIV	GPCR	1997	2007	62
Lapatinib (Tykerb; GlaxoSmithKline)	Cancer	Tyrosine kinase	c. 1993	2007	57
Ambrisentan (Letairis; Gilead)	Pulmonary hypertension	GPCR	c. 1995	2007	63
Etravirine (Intelence; Tibotec Pharmaceuticals)	HIV	Reverse transcriptase	c. 1992	2008	64
Tolvaptan (Samsca; Otsuka Pharmaceutical)	Hyponatraemia	GPCR	c. 1990	2009	65
Eltrombopag (Promacta; GlaxoSmithKline)	Thrombocytopaenia	Cytokine receptor	1997	2008	18

FDA, US Food and Drug Administration; GPCR, G protein-coupled receptor; HTS, high-throughput screening. *Based on early work from J. Inglese (REF. 66).

This table is captured from nature reviews drug discovery 10, 188-195 (March 2011)

Recent years, the speed of drug discovery has been enhanced exponentially due to new technologies both in biology and robotic. Aside from a lot more drug candidates had been discovered, the data of HTS has grown exponentially too.

A robotic arm performing HTS.

The problem I would like to address in this project is the identification of hits from oceans of data. Tools had been developed for hit selection: most of them are huge expensive software suits that normal researcher couldn't afford. Another problem of those tools is that most selection is based by setting cutoffs for a dataset without having an idea of how good the data is overall. Because biological experiments tend to have much more errors, there is a balance between sensitivity (true positive rate) and precision (positive predictive value) which is really hard for algorithms to adjust.

Inspired by human-based computation, I took a different approach for hit selection. I propose that human-guided hit selection is better than automated hit selection. Therefore, I want to build a visualization tool that enable the researcher to "see" the data, and through a set of interactions, enable researcher to easily pick hit out of noisy data.

Data wrangling:

The data I used for this project is from a fluorescence polarization based HTS. The screening is carried out in ICCB facility in a 384-well format. The screening is done in duplicates meaning each small molecule is tested twice with the same assay resulting duplicate readings. There are 2 readouts for this screening: one is called fp, stands for fluorescence polarization. Lower fp means better activity. Because the duplication, there are 2 fp readings, so they are called fpA and fpB. The second readout is called fi, unlike fp, fi doesn't represent small molecule's activity. It is a control reading due to the fact that some small molecules can emit fluorescence by themselves. Same as fp, we have fiA and fiB.

Another crucial part of a HTS screening is controls. On each assay plate, certain wells represent what hit will look like, they are called positive controls. Some other wells represent what non-hits look like, they are called negative controls. By comparing a small molecule well to controls, one can tell if this small molecule is active or not.

A 384-well assay plate

Aside from readouts and controls, there are chemical properties to consider when it comes to hit selection: the structure of small molecule; the logP value; the molecular weight.

Data used in this project is acquired from MightyScreen website (http://mightyscreen.sbgrid.org). Each row of data includes information like plate, well, platewell, readouts. There is a chemical database stored in mightyscreen that contain chemical's properties, logp, molecular weight, structure and so on. The id I used to connect screening data to chemical database is the platewell because it is a unique id. All the scripts I wrote for data wrangling are in get_data.ipynb.

Data is saved to a csv file, the header of this csv file looks like this:

hit,library,plate,platewell,score,well,welltype,fp2,fp3,fp4,logp,molecular_weight,inchikey,svg,formula,sub_library_name,chemical_name,fpA,fpB,fiA,fiB

It is important to point out that this project is not limited to this screening method. The design is to let it be suitable for all HTS datasets. This data is just an example.

Design Evolution

My original plan involves the following core funtions:

a) Assay plates selector. Draw rectangles that represent each assay plates in the dataset. Within each plate, also present the distribution of positive controls and negative controls (boxplot). Allow users to select and unselect plates by clicking.

b) Distribution of one readout. I will use D3 area to present the distribution of readouts. Brushing feature will also be included for user to selection a readout domain. And this selection will be reflected in Scatter plot and Chemical Cloud.

c) Scatter plot of assay readouts. The scatter plot will include 2 drop down boxes allow users to selection which 2 readouts to use. Each chemicals will be color/sized coded for their chemical properties like molecular weight or logp.

d) Chemical Cloud. It is a svg that displays all selected chemical structures. The size of a chemical encodes its activity.

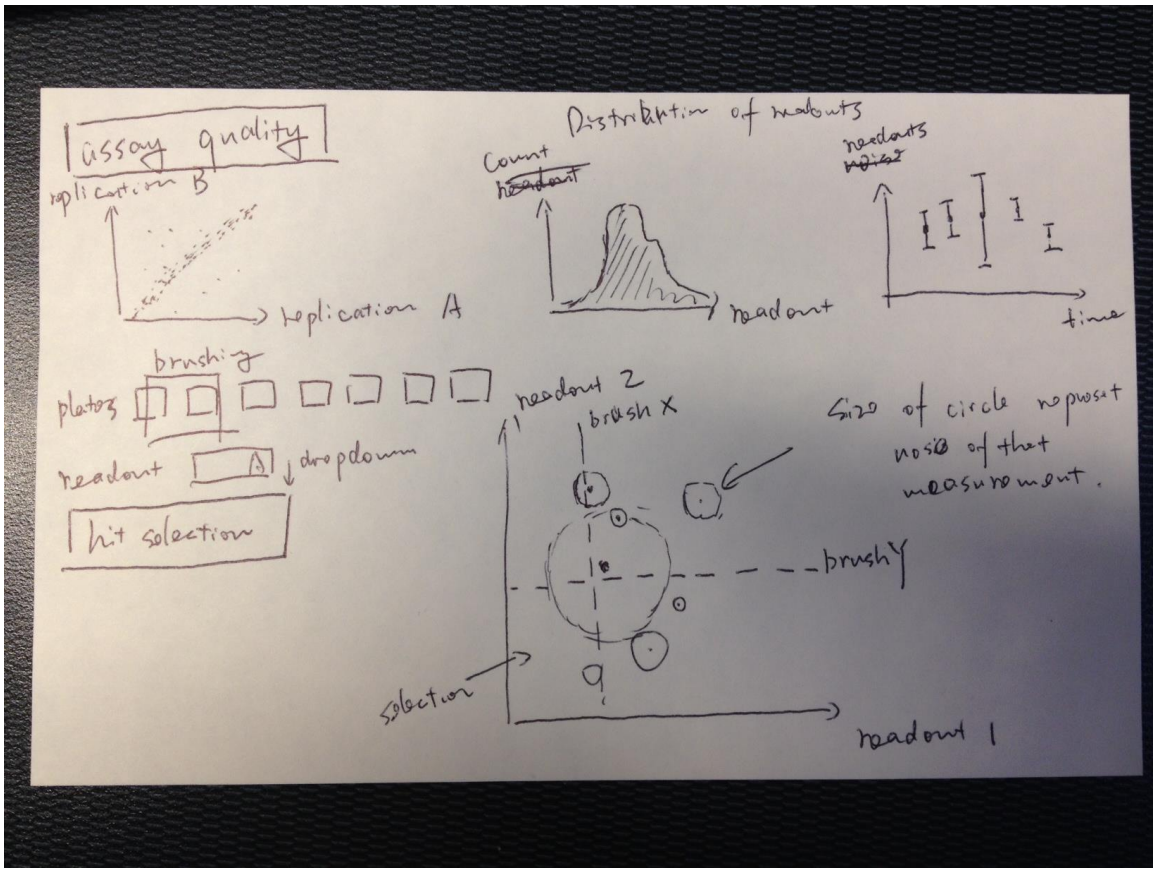


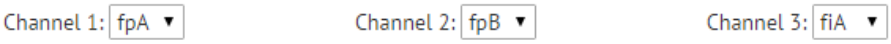
Plate Selector:

The current form of plate selector only has the plate number on it. And its interactive function hasn't been finished yet. The first improvement I did is to add a "plate change" event so that user can select which plate he wants to see. The selected plate is colored in red.

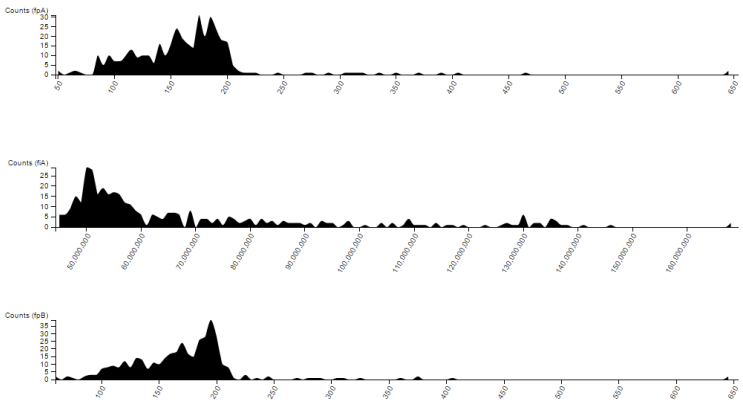
In the future, I want to include more information in this part, like the deviation of controls.

Channels (Distribution of readouts):

A channel is a type of readout that users choose to evaluate a chemical. In this project, there are three channels that user can use. And there is a channel selector for user to select which readout he wants to see in which channel.

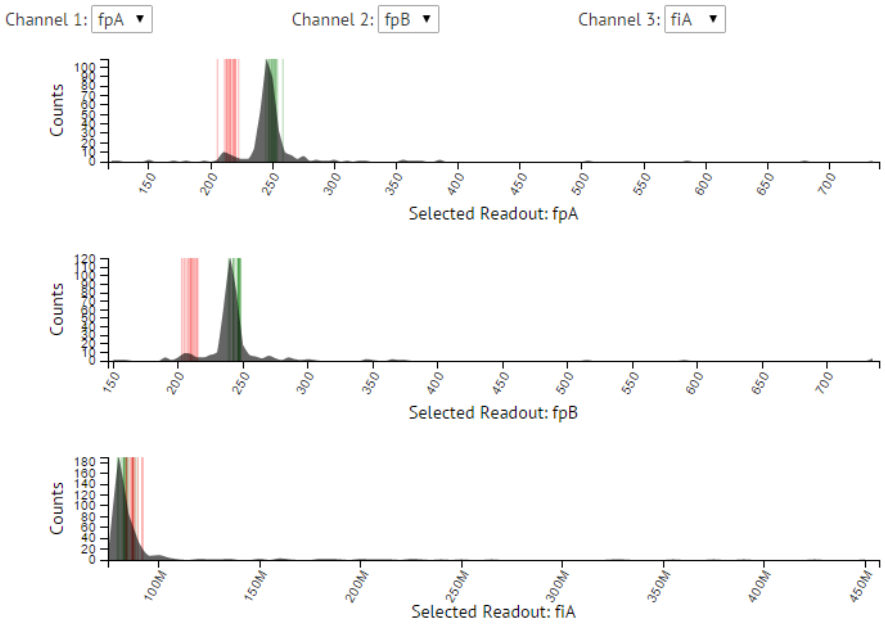


After the channel is selected, the distribution plot for each channel is displayed.



I set three channels: first one is fluorescence polarization A (fpA); second one is fluorescence polarization B (fpB); third one is total fluorescence A (fiA).

The next improvement of channel distribution plot is the distribution of controls. As is shown below, each red line represents a positive controls and each green line represent a negative control. Through this design, user can immediately see the consistency and deviation between controls.



Another feature added to the distribution plot is the brushing. By brushing the distribution plot, user can specify a subset of data to show in other plots like heatmap, or scatterplot.

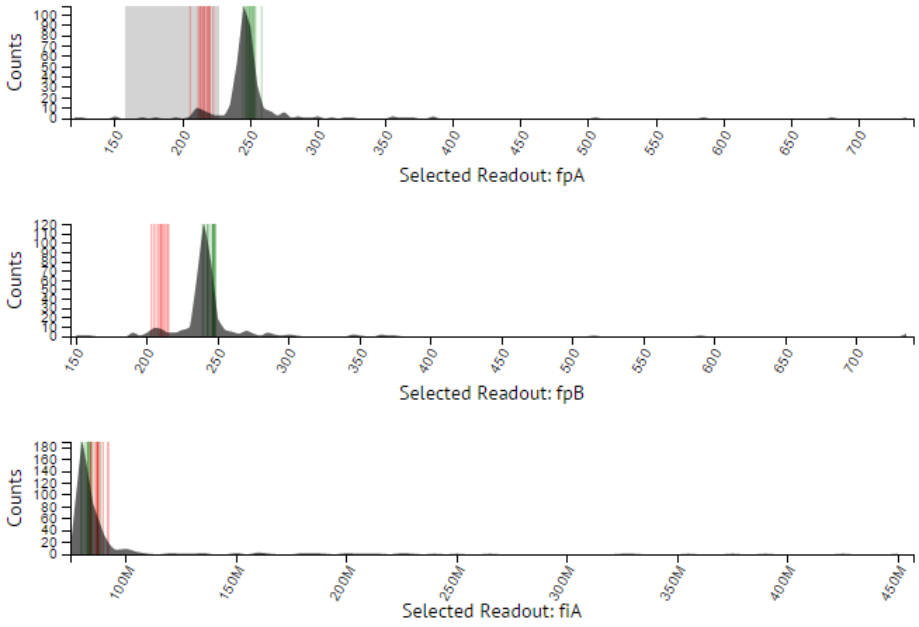
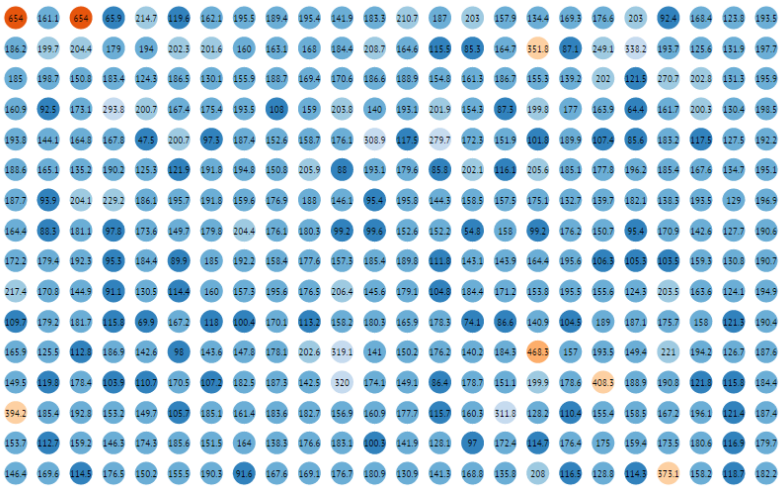
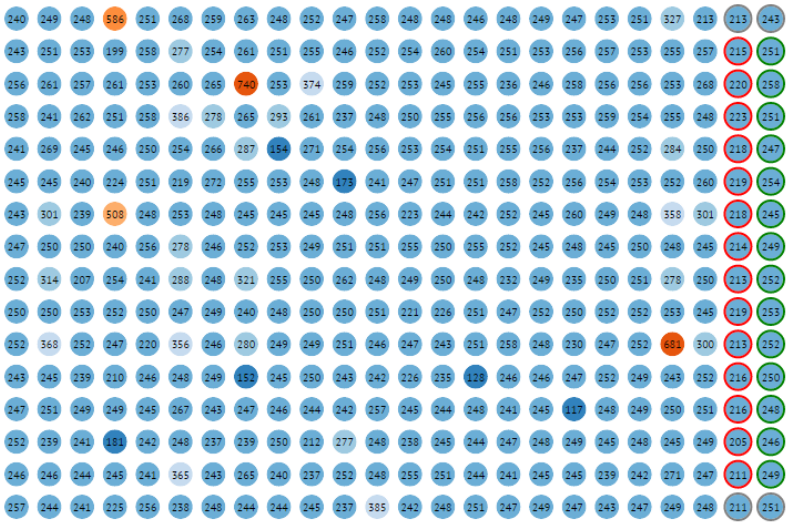


Plate Heatmap:

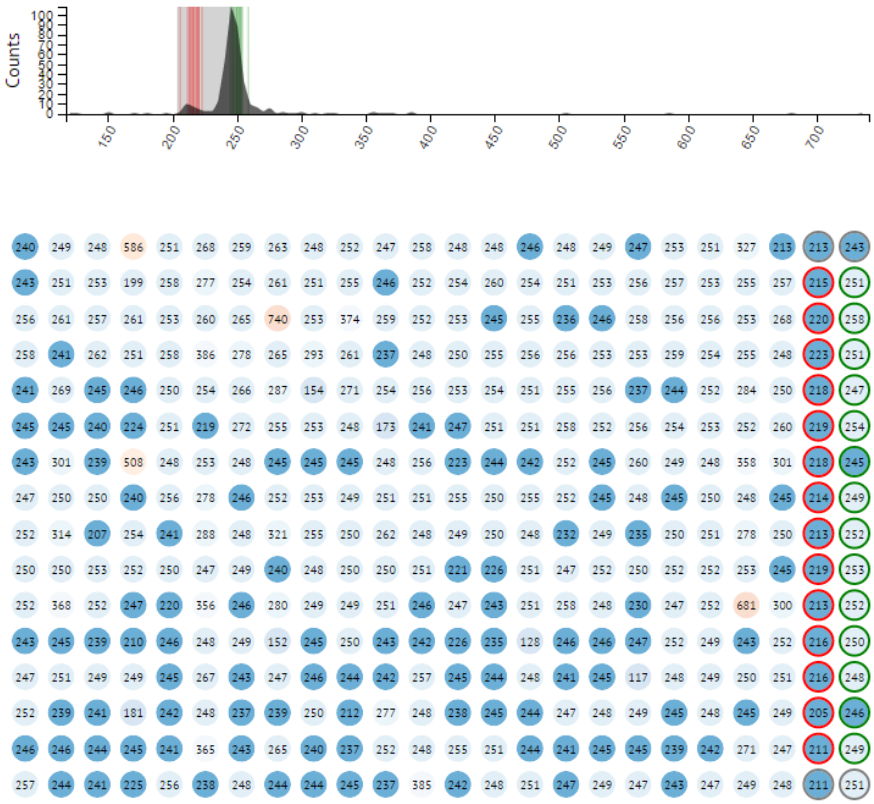
The assay plate is drawn on screen with each well colored according to its readout (channel 1). It is missing labels.



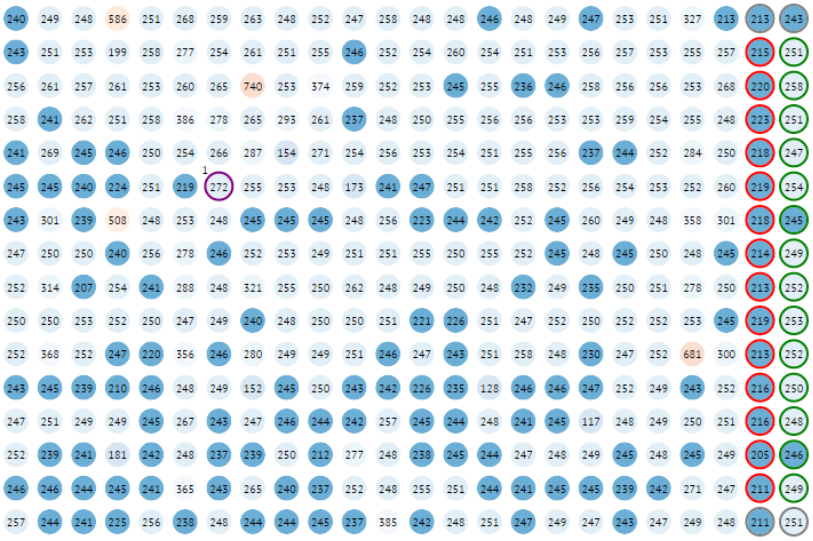
The controls is then label by using red and green color circle outside wells.

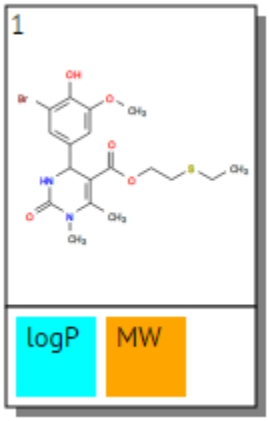


The next implemented feature is when user brushes on the distribution plot, it will highlight only the wells within brush extent.



The final function added is when clicking any of the well. The molecule card will be displated. The selected well is labeled with purple circle and a number. The structure and chemical property of this selected small molecule is displayed.

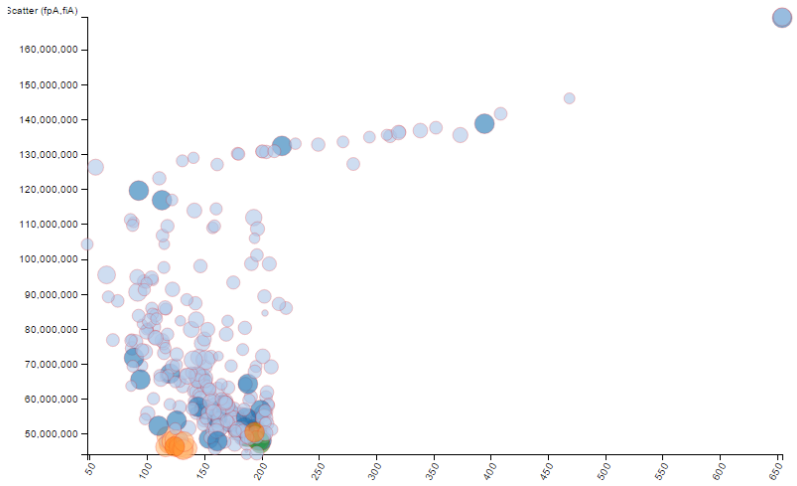




In the chemical card, the top number is its index of all selected chemicals. The chemical structure is shown and on the bottom, there are 2 colored squared, logP and MW, the presence of logP square indicates the molecule's logP value is less than 5, the presence of MW square means the molecular weight is less than 500.

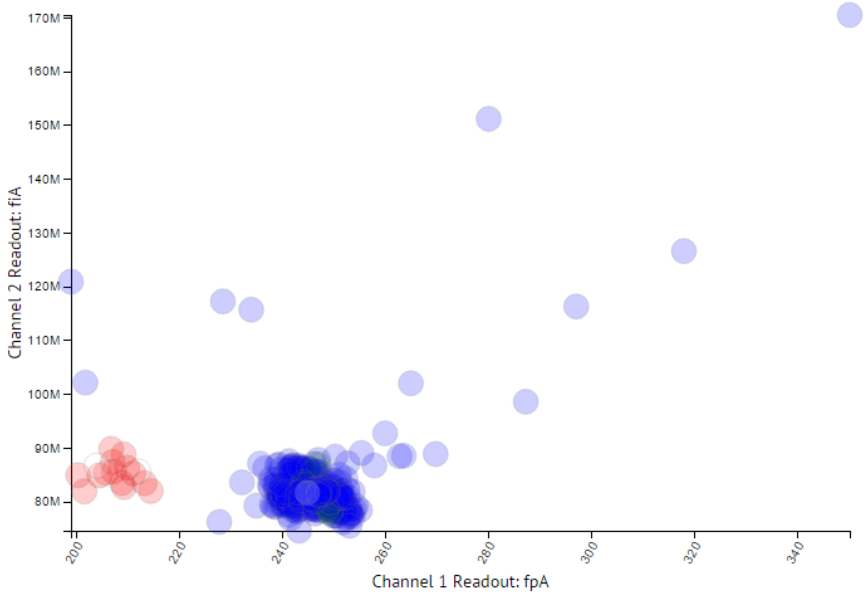
By re-clicking the well or click the chemical card, the chemical will unselected.

Scatter plot.



It is a scatterplot of channel 1 VS channel 2. Each dot is colored by its welltype. The size of a dot encodes its logp. The interactive feature is not implemented yet. But the idea is the same as plate heatmap. I realized that too many encodings will make the scatterplot too complicated to read. So I simplified the encoding, the size of each dot

is the same and red color represents positive controls, green color represents negative controls, blue means a test small molecule.



I included 2 scatterplot in the final view, scatterplot 1 is channel 1 vs channel 2; Scatterplot 2 is channel 1 vs channel 3.

Cloud Display:

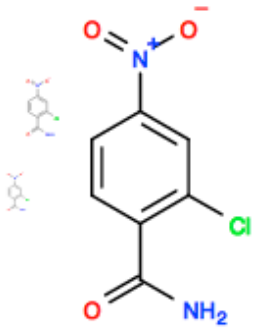
This first step is to display scalable chemical structures. The svg code generated by openbabel is something looks like this:

```
","<?xml version='1.0'?>
<svg version='1.1' id='topsvg'
xmlns='http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink'
xmlns:cml='http://www.xml-cml.org/schema' x='0' y='0' width='200px' height='200px' viewBox='0 0 100 100'>
<title>OBDepict</title>
<rect x='0' y='0' width='100' height='100' fill='white'/>
<text text-anchor='middle' font-size='6' fill ='black' font-family='sans-serif'
x='50' y='98' >Structure641</text>
<g transform='translate(0,0)'>
<svg width='100' height='100' x='0' y='0' viewBox='0 0 184.286 280.083'
font-family='sans-serif' stroke='rgb(0,0,0)' stroke-width='2'  stroke-linecap='round'>
<line x1='74.6' y1='73.0' x2='74.6' y2='100.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='85.8' y1='53.5' x2='97.9' y2='46.5' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='61.8' y1='56.1' x2='49.7' y2='49.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='64.8' y1='50.9' x2='52.8' y2='43.9' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='180.1' x2='74.6' y2='220.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='160.0' x2='133.0' y2='173.6' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='76.1' y1='222.7' x2='52.8' y2='236.2' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='73.1' y1='217.5' x2='49.7' y2='231.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='220.1' x2='97.9' y2='233.6' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='160.0' x2='74.6' y2='180.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='100.3' y1='156.8' x2='76.1' y2='170.8' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='180.1' x2='40.0' y2='160.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='40.0' y1='160.0' x2='40.0' y2='120.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='47.2' y1='154.0' x2='47.2' y2='126.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='40.0' y1='120.0' x2='74.6' y2='100.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='100.0' x2='109.1' y2='120.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='76.1' y1='109.3' x2='100.3' y2='123.3' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='120.0' x2='109.1' y2='160.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<text x='79.360492' y='55.975205' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >+</text>
<text x='68.560492' y='68.007205' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >N</text>
<text x='113.920984' y='25.888000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >_</text>
<text x='103.120984' y='48.000000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='34.000000' y='48.000000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='34.000000' y='248.083457' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='105.120984' y='248.083457' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >NH</text>
<text x='129.120984' y='251.763457' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='13' >2</text>
<text x='138.286200' y='188.058990' fill='rgb(30,239,30)'  stroke='rgb(30,239,30)' stroke-width='1' font-
size='16' >Cl</text>
</svg>
</g>
</svg>
```

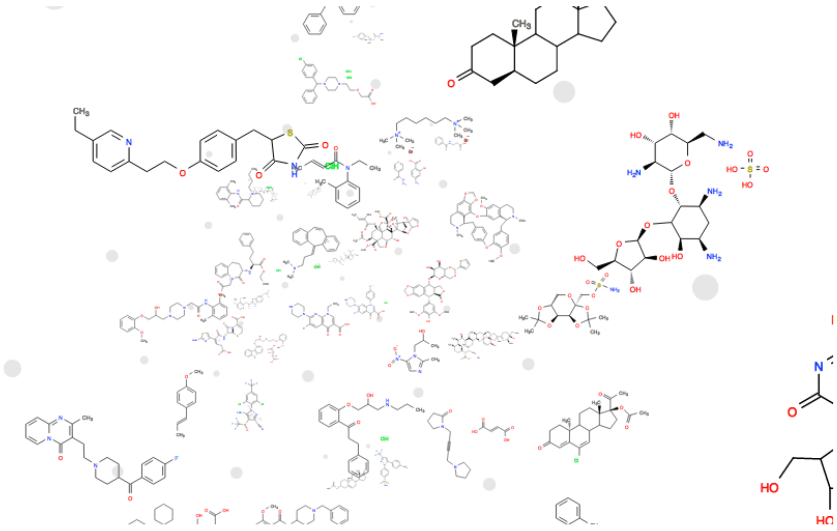
It uses viewbox which makes it easier to us to scale. I basically deleted all non-needed part and replace svg label to symbol:

```
<symbol width='100' height='100' x='0' y='0' viewBox='0 0 184.286 280.083'
font-family='sans-serif' stroke='rgb(0,0,0)' stroke-width='2'  stroke-linecap='round'>
<line x1='74.6' y1='73.0' x2='74.6' y2='100.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='85.8' y1='53.5' x2='97.9' y2='46.5' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='61.8' y1='56.1' x2='49.7' y2='49.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='64.8' y1='50.9' x2='52.8' y2='43.9' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='180.1' x2='74.6' y2='220.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='160.0' x2='133.0' y2='173.6' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='76.1' y1='222.7' x2='52.8' y2='236.2' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='73.1' y1='217.5' x2='49.7' y2='231.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='220.1' x2='97.9' y2='233.6' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='160.0' x2='74.6' y2='180.1' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='100.3' y1='156.8' x2='76.1' y2='170.8' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='180.1' x2='40.0' y2='160.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='40.0' y1='160.0' x2='40.0' y2='120.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='47.2' y1='154.0' x2='47.2' y2='126.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='40.0' y1='120.0' x2='74.6' y2='100.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='74.6' y1='100.0' x2='109.1' y2='120.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='76.1' y1='109.3' x2='100.3' y2='123.3' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<line x1='109.1' y1='120.0' x2='109.1' y2='160.0' stroke='rgb(0,0,0)'  stroke-width='2.0'/>
<text x='79.360492' y='55.975205' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >+</text>
<text x='68.560492' y='68.007205' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >N</text>
<text x='113.920984' y='25.888000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >_</text>
<text x='103.120984' y='48.000000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='34.000000' y='48.000000' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='34.000000' y='248.083457' fill='rgb(255,12,12)'  stroke='rgb(255,12,12)' stroke-width='1' font-
size='16' >O</text>
<text x='105.120984' y='248.083457' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='16' >NH</text>
<text x='129.120984' y='251.763457' fill='rgb(12,12,255)'  stroke='rgb(12,12,255)' stroke-width='1' font-
size='13' >2</text>
<text x='138.286200' y='188.058990' fill='rgb(30,239,30)'  stroke='rgb(30,239,30)' stroke-width='1' font-
size='16' >Cl</text>
</symbol>
```

Now I can use <use> tag to draw the symbol at any scale:



After feeding svgs to a force layout (with collision detection), it looks like this:



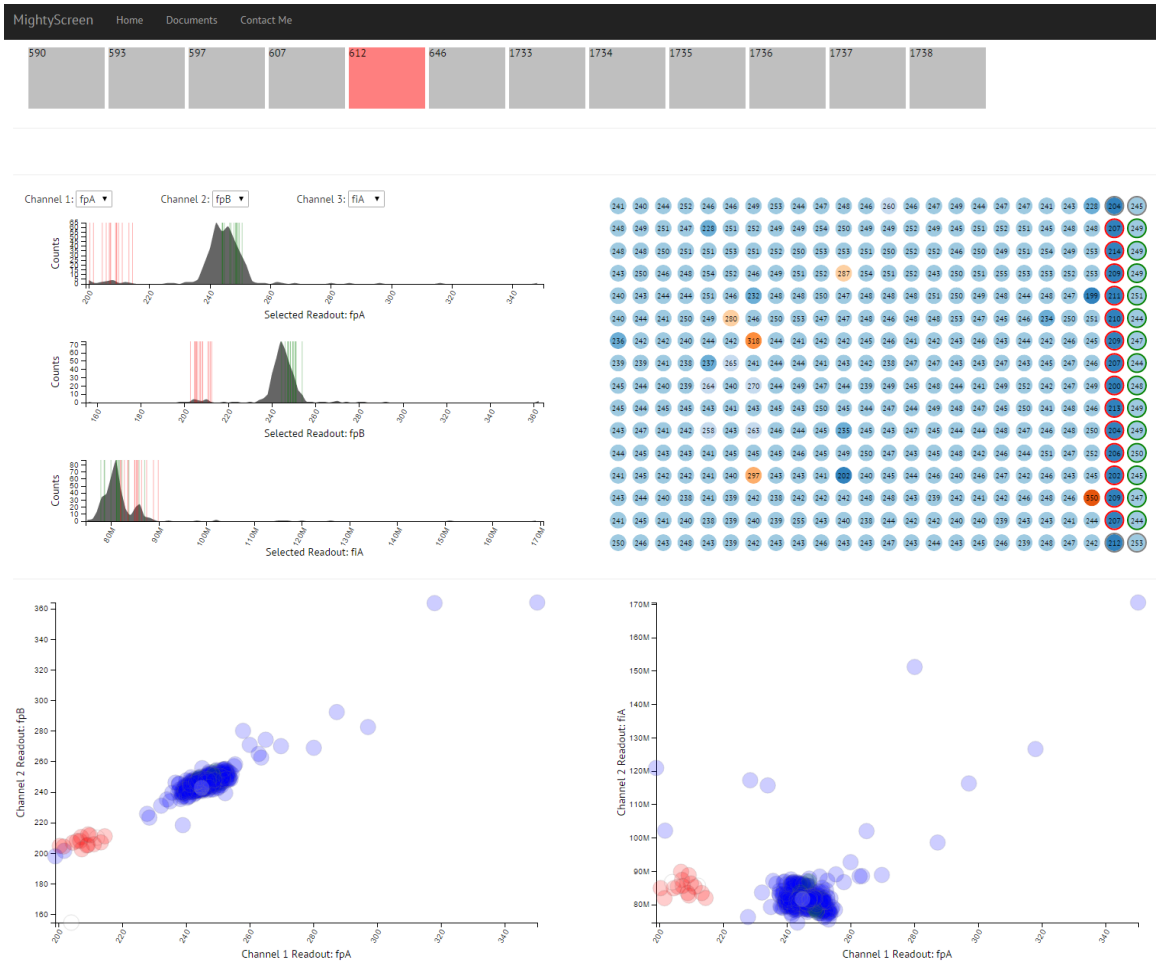
Current Issues:

A) It is hard to drag molecules. Possible solution is to draw a handle to each molecule.
B) Collision detection is kind of messy. Molecules overlap with each other.

After some discussion with Hendrik, we found although this plot looks fancy but it doesn't really make selection easier, so we decided to take it out from the final presentation. It might be useful for some other visualization.

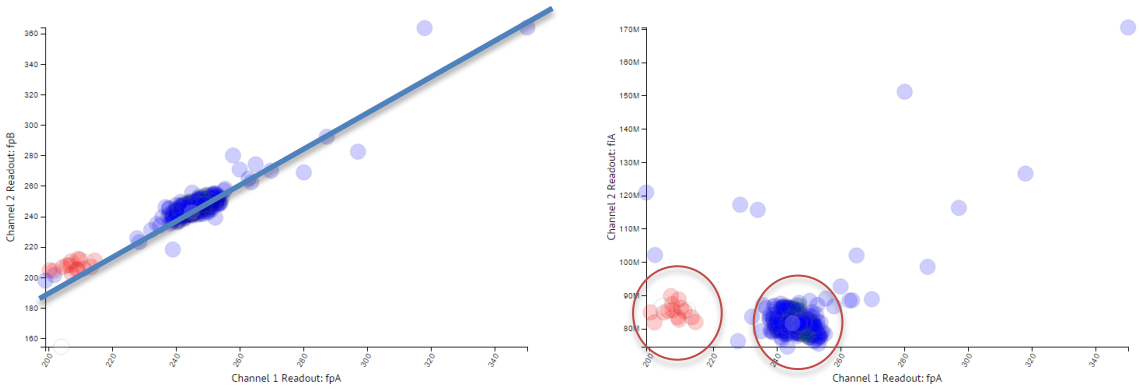
Implementation and Evaluation

Now the whole visualization looks like this.

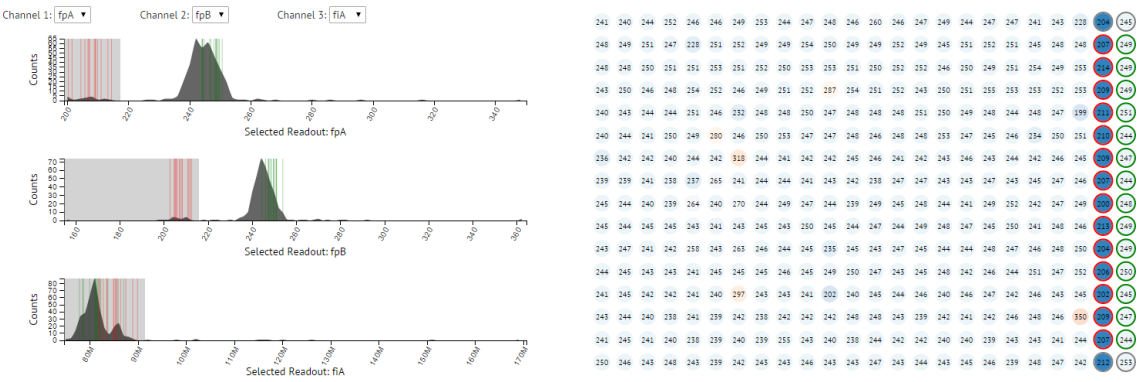


Now let's try to find a hit using all the visualization I built.

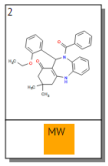
First of all, it is obvious that the data quality is really great on this plate. On scatterplot 1, duplicate readings are really well correlated (as shown by the line). On scatterplot 2, the positive controls and chemicals/negative controls is really well separated.

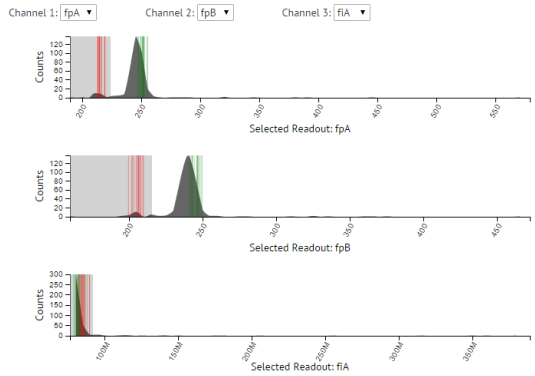


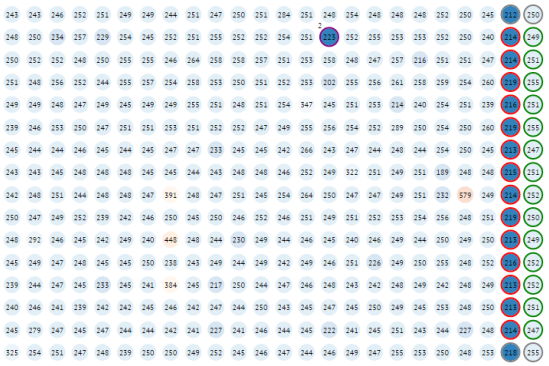
Then I will brush the channel 1, 2 to set it only to show fp lower than positive controls. On channel 3, I'll brush to cover all controls just to exclude auto-fluorescence chemicals.



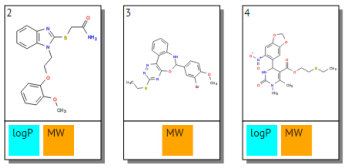
The result is that only positive controls are left on the plate, meaning there is no hit on this plate. Now let's move on to next plate.

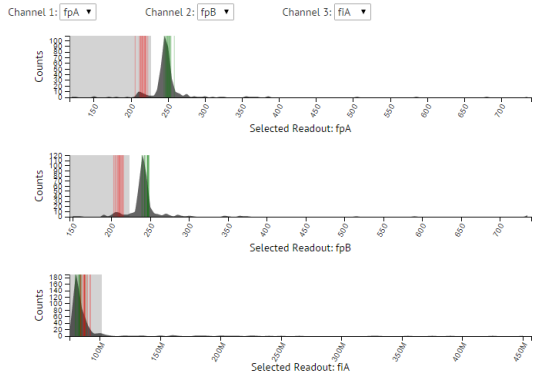


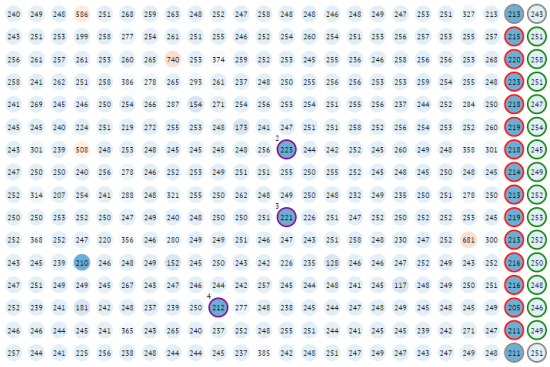




On this plate, using the same method, I am able to identify one small molecule. I immediately know that the molecular weight of it is less than 500 but the logP value is not optimal as a drug candidate.







On another plate, I finally got chemicals that have both optimal logP and molecular weight as drug candidates.

Summary and future plan

In this project, I designed and implemented a tool for HTS hit selection. Using this tool, user can directly see their data, play with their data and make decisions based on the visualizations. I think this tool will be very useful for myself and other researchers who deal with HTS data. The overall design is to break down a complicated data into important features.
One future plan is to add a new function that can do small molecule structure based clustering.