DATA_WALKTHROUGH

August 27, 2020

1 Release One Data Walkthrough

The CRyPTIC data tables and their associated schema is now quite complex. This jupyter notebook walks you through the identity of the tables. It should be read alongside `DATA_SCHEMA.pdf` which you can also find in the `cryptic-tables/` directory. New fields and tables are coloured red.

I'll first explain the tables containing phenotype (i.e. minimium inhibitory concentration) data before considering the genetic data.

This document is available as a PDF or as an interactive jupyter notebook which will let you run each cell containing Python code for yourself. Note that since you have read only access to this folder, if you want to run (i.e. alter) the notebook you will need to copy it to another location on your computer and change `TABLES_PATH` to point to the `cryptic-tables/` folder.

Philip W Fowler

27 August 2020

[1]:
```
import pandas, numpy
pandas.set_option('display.max_columns', 200)

%matplotlib inline
import matplotlib.pyplot as plt

TABLES_PATH="./"
```

2 A note on unique identifiers

CRyPTIC uses a hierarchical set of identifiers.

- `SITEID` a two-digit, left-padded number (`'%02d'`) uniquely identifying each site. Lookup table in `SITES.csv`
- `SUBJID` a string that uniquely identifies each patient in a specific site (might not be unique between sites). Some sites use a left-padded number (e.g. India uses `'%05d'` and China uses `'%04d'`) whilst others are alphanumerical strings. Sometimes country of origin can be inferred.

- LABID a string that uniquely identifies the clinical sample. In principle a SUBJID can therefore have multiple LABIDs associated with, but in practice there is usually only one. Some sites do not need this level of granularity and simply duplicated SUBJID in this field.
- ISOLATENO integer that identifies the particular isolate tested. Mostly 1. Some cases where the phenotype and genetic data uses different ISOLATENOs.
- SEQ_REPS a string associated with the genetics ('sequence repeats'). Again mostly '1', but in principle allows for sequencing to be repeated. Since sequencing appears to mostly fail due to coverage, it is common to see aggregation of short reads e.g. 1_2_3
- READING_DAY the number of days of incubation at which the plate was read. Specific to the phenotypic data. CRyPTIC uses 14 days unless there is poor growth when it uses 21 days, but there are a wide range of days.

CliRes uses USUBJID as its unique identifier which is SITEID+"-"+SUBJID. Many of the tables use UNIQUEID which is a concatenation of the first four identifiers above e.g.

site.02.subj.0003.lab.20142220007.iso.1

3 Phenotypic data

There are three levels to the phenotype data descending hierarchy in https://clires2.oucru.org.

1. SUBJECT data. This contains
2. SAMPLE
3. DST data.

These map onto SUBJID i.e. patient-level information, LABID i.e. sample-specific information, including other phenotypic tests such as MGIT that were run on this sample and lastly READINGDAY i.e. the MICs read from the 96-well plate after a specified number of days incubation.

All tables were downloaded and populated from https://clires2.oucru.org using their WebAPI via the Python zeep package. Hence if a CRyPTIC lab did not enter their data into CliRes, but instead provided a spreadsheet then they may only has DST entries in the UKMYC_PHENOTYPES table and no rows in SUBJECT or SAMPLES. This applies to CDC Atlanta (01), Italy (06) and Sweden (11) and possibly others.

The original numerical fields have been replaced by descriptive labels to aid interpretation using the CRyPTIC METADATA FILE SPECIFICATION . e.g. GENDER in SUBJECTS contains MALE/FEMALE/OTHER/UNKNOWN rather than 1/2/3/9. The version used was April 2018 v2.0. Note that there are some inconsistencies where the spec dictates the values should be 0/1 and they are 1/2.

3.1 SUBJECTS

This only contains 3 fields.

[2]:
```
SUBJECTS=pandas.read_pickle(TABLES_PATH+"SUBJECTS.pkl.gz")
SUBJECTS[:3]
```

```
[2]:                GENDER COUNTRY_OF_ORIGIN
     SITEID SUBJID
     02     0958       male               NaN
            0823    unknown               NaN
            0359    unknown               CHN
```

```
[3]: SUBJECTS.GENDER.value_counts(dropna=False).sort_index()
```

```
[3]: female     4916
     male       6733
     other         2
     unknown    3848
     Name: GENDER, dtype: int64
```

COUNTRY_OF_ORIGIN was not a mandatory field so there are >2,500 missing values.

```
[4]: SUBJECTS.COUNTRY_OF_ORIGIN.value_counts(dropna=False)[:5]
```

```
[4]: IND    5177
     PER    3454
     NaN    3269
     DEU     956
     CHN     896
     Name: COUNTRY_OF_ORIGIN, dtype: int64
```

These are described using the ISO 3166-1 alpha-3 country codes. To help with drawing maps later, there is also a lookup table containing all the 3 letter codes, a proper name and, crucially, the lattitude and longitude.

```
[5]: COUNTRIES_LOOKUP=pandas.read_csv("COUNTRIES_LOOKUP.csv")
     COUNTRIES_LOOKUP[:5]
```

```
[5]:      COUNTRY_NAME COUNTRY_CODE_2_LETTER COUNTRY_CODE_3_LETTER  \
     0     Afghanistan                    AF                   AFG
     1         Albania                    AL                   ALB
     2         Algeria                    DZ                   DZA
     3  American Samoa                    AS                   ASM
     4         Andorra                    AD                   AND

        COUNTRY_CODE_NUMERIC      LAT   LONG
     0                     4  33.0000   65.0
     1                     8  41.0000   20.0
     2                    12  28.0000    3.0
     3                    16 -14.3333 -170.0
     4                    20  42.5000    1.6
```

The COUNTRY_CODE_3_LETTER column (after appropriate renaming) can be used to join to other COUNTRY_OF_ORIGIN or COUNTRY_WHERE_SAMPLE_TAKEN to plot maps using (LAT,LONG)

```
[6]: df=SUBJECTS.reset_index()
     df=df.
      ↪merge(COUNTRIES_LOOKUP,left_on="COUNTRY_OF_ORIGIN",right_on="COUNTRY_CODE_3_LETTER",how='le
     df.COUNTRY_NAME.value_counts()[:5]
```

```
[6]: India      5177
     Peru       3454
     Germany     956
     China       896
     Taiwan      392
     Name: COUNTRY_NAME, dtype: int64
```

3.2 SAMPLES

```
[7]: SAMPLES=pandas.read_pickle(TABLES_PATH+"SAMPLES.pkl.gz")
     SAMPLES[:3]
```

```
[7]:                          COUNTRY_WHERE_SAMPLE_TAKEN           REGION  \
     SITEID SUBJID LABID
     02     0958   22A197                            CHN        CHONGQING
            0823   2013241494                        CHN          GUIZHOU
            0359   222018-14                         CHN  China ChongQing

                                        COLLECTION_DATE  \
     SITEID SUBJID LABID
     02     0958   22A197     2017-12-04 00:00:00+07:00
            0823   2013241494 2013-10-06 00:00:00+07:00
            0359   222018-14  2014-01-01 00:00:00+07:00

                               ISOLATE_COLLECTED_PROSPECTIVELY ANATOMICAL_ORIGIN  \
     SITEID SUBJID LABID
     02     0958   22A197                                False         not known
            0823   2013241494                            False         not known
            0359   222018-14                             False         not known

                              SMEAR_RESULT WGS_SEQUENCING_PLATFORM XPERT_MTB_RIF  \
     SITEID SUBJID LABID
     02     0958   22A197        not known                   HiSeq    not tested
            0823   2013241494    not known                   HiSeq    not tested
            0359   222018-14     not known                   HiSeq    not tested

                                 HAIN_RIF    HAIN_INH     HAIN_FL     HAIN_AM  \
     SITEID SUBJID LABID
     02     0958   22A197      not tested  not tested  not tested  not tested
            0823   2013241494  not tested  not tested  not tested  not tested
            0359   222018-14   not tested  not tested  not tested  not tested
```

```
                              HAIN_ETH     SMOKER INJECT_DRUG_USER  IS_HOMELESS  \
SITEID SUBJID LABID
02     0958   22A197      not tested  not known        not known        False
       0823   2013241494  not tested  not known        not known        False
       0359   222018-14   not tested  not known        not known        False

                            IS_IMPRISONED        HIV   DIABETES WHO_OUTCOME
SITEID SUBJID LABID
02     0958   22A197                False  not known  not known   not known
       0823   2013241494            False  not known  not known   not known
       0359   222018-14             False  not known  not known   not known
```

Usefully `COUNTRY_WHERE_SAMPLE_TAKEN` is a mandatory field. `REGION` is freeform text so will require cleaning if it is to be used.

[8]:
```
SAMPLES.COUNTRY_WHERE_SAMPLE_TAKEN.value_counts(dropna=False)[:6]
```

[8]:
```
IND    5109
PER    3450
ZAF    2267
CHN    1509
VNM    1112
DEU     851
Name: COUNTRY_WHERE_SAMPLE_TAKEN, dtype: int64
```

The `COLLECTION_DATE` might be useful, although beware samples from 1900 and 2201!

[9]:
```
SAMPLES[['COLLECTION_DATE']].groupby(SAMPLES.COLLECTION_DATE.dt.year).count()
```

[9]:
```
                 COLLECTION_DATE
COLLECTION_DATE
1900                           2
1986                           1
2001                           1
2003                          30
2004                          31
2005                          13
2006                           3
2007                         111
2008                          41
2009                         133
2010                          83
2011                         122
2012                         495
2013                        1360
2014                         856
2015                         923
```

```
2016                       1011
2017                       2825
2018                       4540
2019                       2921
2020                          5
2201                          1
```

[10]:
```
SAMPLES.ISOLATE_COLLECTED_PROSPECTIVELY.value_counts(dropna=False)
```

[10]:
```
False    12451
True      3057
Name: ISOLATE_COLLECTED_PROSPECTIVELY, dtype: int64
```

As one might expect the vast majority of samples are respiratory. Again this was not a required field so beware the large number of `'not known'` values.

[11]:
```
SAMPLES.ANATOMICAL_ORIGIN.value_counts(dropna=False).
 ↪sort_values(ascending=False)
```

[11]:
```
Respiratory                        7393
not known                          5306
Other known site                   1843
Lymph node                          461
CSF                                 344
Pleural                             134
Non-respiratory, site not known      22
Bone                                  5
Name: ANATOMICAL_ORIGIN, dtype: int64
```

There is also some smear data for some samples

[12]:
```
SAMPLES.SMEAR_RESULT.value_counts(dropna=False)
```

[12]:
```
not known    6388
Negative     3351
+            2619
+++          1340
++           1223
Scanty        587
Name: SMEAR_RESULT, dtype: int64
```

You can also check what sequencing platform was used (although this is, perhaps more correctly, recorded in the metadata spreadsheets sent to the EBI along with the `FASTQ` files)

[13]:
```
SAMPLES.WGS_SEQUENCING_PLATFORM.value_counts(dropna=False)
```

[13]:
```
NextSeq        6706
HiSeq          5499
```

```
Other           2173
MiSeq            969
NovaSeq6000      161
Name: WGS_SEQUENCING_PLATFORM, dtype: int64
```

Some samples have been on an Xpert MTB/RIF cartridge (note that there were only a handful of Ultra samples so these were discarded)

```
[14]: SAMPLES.XPERT_MTB_RIF.value_counts(dropna=False)
```

```
[14]: not tested           13859
      RIF susceptible        683
      test inconclusive      482
      RIF resistant          480
      test failed              4
      Name: XPERT_MTB_RIF, dtype: int64
```

A similar number also have Hain LPA results recorded for RIF, INH,

```
[15]: SAMPLES.HAIN_RIF.value_counts()
```

```
[15]: not tested           13983
      susceptible            930
      resistant              453
      test inconclusive      140
      test failed              2
      Name: HAIN_RIF, dtype: int64
```

```
[16]: SAMPLES.HAIN_INH.value_counts()
```

```
[16]: not tested           13983
      susceptible            865
      resistant              518
      test inconclusive      140
      test failed              2
      Name: HAIN_INH, dtype: int64
```

```
[17]: SAMPLES.HAIN_ETH.value_counts()
```

```
[17]: not tested           15331
      test inconclusive      140
      susceptible             30
      resistant                7
      Name: HAIN_ETH, dtype: int64
```

The below is understood to be a generic fluoroquinolone result

```
[18]: SAMPLES.HAIN_FL.value_counts()
```

[18]:
```
not tested            15331
test inconclusive       140
susceptible              22
resistant                15
Name: HAIN_FL, dtype: int64
```

The below is understood to be a generic aminoglycoside result

[19]:
```
SAMPLES.HAIN_AM.value_counts()
```

[19]:
```
not tested            15331
test inconclusive       140
susceptible              32
resistant                 5
Name: HAIN_AM, dtype: int64
```

Now we have some sparse lifestyle data. First is whether they smoked or not.

[20]:
```
SAMPLES.SMOKER.value_counts()
```

[20]:
```
not known          14260
no                   937
yes, currently       172
yes, previously      139
Name: SMOKER, dtype: int64
```

[21]:
```
SAMPLES.INJECT_DRUG_USER.value_counts()
```

[21]:
```
not known          14268
no                  1104
yes, previously      113
yes, currently        23
Name: INJECT_DRUG_USER, dtype: int64
```

[22]:
```
SAMPLES.IS_HOMELESS.value_counts()
```

[22]:
```
False    15490
True        18
Name: IS_HOMELESS, dtype: int64
```

[23]:
```
SAMPLES.IS_IMPRISONED.value_counts()
```

[23]:
```
False    15438
True        70
Name: IS_IMPRISONED, dtype: int64
```

[24]:
```
SAMPLES.HIV.value_counts()
```

[24]:
```
not known           12895
tested, negative     1938
tested, positive      669
not tested              6
Name: HIV, dtype: int64
```

[25]:
```
SAMPLES.DIABETES.value_counts()
```

[25]:
```
not known                   13421
tested, not diabetic         1892
tested, type 2 diabetes       107
tested, unknown type           68
tested, type 1 diabetes        20
Name: DIABETES, dtype: int64
```

Finally, a small number had a WHO outcome field recorded. These have been translated

[26]:
```
SAMPLES.WHO_OUTCOME.value_counts(dropna=False).sort_index()
```

[26]:
```
cured                              1567
died                                277
lost to follow-up or defaulted      265
not evaluated                        29
not known                         12531
treatment completed                 739
treatment failed                    100
Name: WHO_OUTCOME, dtype: int64
```

3.3 UKMYC_PLATES

UKMYC_PLATES contains one row per plate. It is a simplified view of the old PLATES and PLATE_MEASUREMENTS tables and hence contains 'the' reading and therefore the READINGDAY, which in most cases will be day 14. All other readings taken on other reading days are not shown in this view.

[27]:
```
UKMYC_PLATES=pandas.read_pickle(TABLES_PATH+"UKMYC_PLATES.pkl.gz")
UKMYC_PLATES[:3]
```

[27]:
```
                                     SITEID  SUBJID   LABID ISOLATENO  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1     11  MDR044  SWE-33         1
site.11.subj.MDR045.lab.SWE-34.iso.1     11  MDR045  SWE-34         1
site.11.subj.MDR046.lab.SWE-35.iso.1     11  MDR046  SWE-35         1

                                     READINGDAY  BELONGS_GPI PLATEDESIGN  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1         10        False      UKMYC5
site.11.subj.MDR045.lab.SWE-34.iso.1         10         True      UKMYC5
site.11.subj.MDR046.lab.SWE-35.iso.1         10         True      UKMYC5

                                                                TREE_PATH  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1  dat/CRyPTIC2/V2/11/MDR044/SWE-33/1/10/
site.11.subj.MDR045.lab.SWE-34.iso.1  dat/CRyPTIC2/V2/11/MDR045/SWE-34/1/10/
site.11.subj.MDR046.lab.SWE-35.iso.1  dat/CRyPTIC2/V2/11/MDR046/SWE-35/1/10/

                                     IMAGEFILENAME IMAGE_MD5SUM  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1           NaN          NaN
site.11.subj.MDR045.lab.SWE-34.iso.1           NaN          NaN
site.11.subj.MDR046.lab.SWE-35.iso.1           NaN          NaN

                                      DUPLICATED_IMAGE  IM_IMAGE_DOWNLOADED  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1             False                False
site.11.subj.MDR045.lab.SWE-34.iso.1             False                False
site.11.subj.MDR046.lab.SWE-35.iso.1             False                False

                                      IM_IMAGE_FILTERED  IM_WELLS_IDENTIFIED  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1              False                False
site.11.subj.MDR045.lab.SWE-34.iso.1              False                False
site.11.subj.MDR046.lab.SWE-35.iso.1              False                False

                                      IM_POS1GROWTH  IM_POS2GROWTH  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1            0.0            0.0
site.11.subj.MDR045.lab.SWE-34.iso.1            0.0            0.0
site.11.subj.MDR046.lab.SWE-35.iso.1            0.0            0.0

                                      IM_POS_AVERAGE  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1             NaN
site.11.subj.MDR045.lab.SWE-34.iso.1             NaN
site.11.subj.MDR046.lab.SWE-35.iso.1             NaN

                                      IM_DRUGS_INCONSISTENT_GROWTH  \
UNIQUEID
site.11.subj.MDR044.lab.SWE-33.iso.1                           NaN
site.11.subj.MDR045.lab.SWE-34.iso.1                           NaN
site.11.subj.MDR046.lab.SWE-35.iso.1                           NaN

                                      TRUST_PHENOTYPES
UNIQUEID
```

```
site.11.subj.MDR044.lab.SWE-33.iso.1             True
site.11.subj.MDR045.lab.SWE-34.iso.1             True
site.11.subj.MDR046.lab.SWE-35.iso.1             True
```

[28]:
```
UKMYC_PLATES.READINGDAY.value_counts(dropna=False)
```

[28]:
```
14    19834
21      723
10       80
28        0
7         0
Name: READINGDAY, dtype: int64
```

The BELONGS_GPI field is new, and tells us if this plate belong to the 'Geno-Pheno-Intersection' i.e. whether it was sequenced and successfully processed using Clockwork by the EBI.

[29]:
```
UKMYC_PLATES.BELONGS_GPI.value_counts()
```

[29]:
```
True     15039
False     5598
Name: BELONGS_GPI, dtype: int64
```

This is a bit less than the original number in the GPI of 15,211 since invalid plates (poor growth, contamination, problems with the control wells) have been excluded.

The PLATEDESIGN field is important since it tells us which antibiotics where on the plate, where they are located and their concentrations. You can look this up via

[30]:
```
PLATE_DESIGN=pandas.read_csv(TABLES_PATH+"PLATE_LAYOUT.csv.gz")
PLATE_DESIGN[:3]
```

[30]:
```
  PLATEDESIGN DRUG  DILUTION CONC  ROW  COL BINARY_PHENOTYPE
0      UKMYC5  AMI         7   >8  NaN  NaN                R
1      UKMYC5  AMI         6    8  1.0  1.0                R
2      UKMYC5  AMI         5    4  2.0  1.0                R
```

Note that, for ease, the binary phenotype for each MIC according to the 'current' CRyPTIC ECOFFs is also included in this table. The CRyPTIC ECOFFs may change slightly will change the BINARY_PHENOTYPE assignments – you'll be notified if this happens. We shall use this later

A 3-letter code is used in all the tables to identify all the drugs. Whilst these are mostly standard/obvious, there is a further lookup table you can use to get more information on what drug each 3-letter code describes.

[31]:
```
DRUG_LOOKUP=pandas.read_csv(TABLES_PATH+'DRUG_LOOKUP.csv.gz')
DRUG_LOOKUP[:3]
```

[31]:
```
  SAMPLES_COLUMN OTHER_PHENOTYPES_COLUMN DRUG_ABBREVATION
0     RIFAMPICIN                     RIF              RIF
```

```
1      RIFMETHOD                RIF_METHOD                RIF
2         RIFOTH                 RIF_OTHER                RIF
```

Note that CRyPTIC Release One is about 60% UKMYC6, at least when it comes to "samples on plates"

[32]:
```
UKMYC_PLATES.PLATEDESIGN.value_counts(dropna=False)
```

[32]:
```
UKMYC6    12672
UKMYC5     7965
Name: PLATEDESIGN, dtype: int64
```

Different to before, this table contains additional fields that allow you to retrieve raw files **directly** from the sharded data tree (assuming you have access). Central to this is the `TREE_PATH` which gives you the relative path to the leaf where the files (in this case images) are stored. For example, if you wanted to retrieve a list of raw images you could do (bit clunky but works)

[33]:
```
def return_good_images(row):
    print('/well/bag/pfowler/cryptic/
 ↪'+str(row['TREE_PATH'])+str(row['IMAGEFILENAME'])+'-'+row['PLATEDESIGN']+'-growth.
 ↪jpg')

GOOD_PLATE_IMAGES=UKMYC_PLATES.loc[(UKMYC_PLATES.IMAGEFILENAME.notna()) &␣
 ↪(UKMYC_PLATES.IM_IMAGE_FILTERED.notna()) & (~UKMYC_PLATES.DUPLICATED_IMAGE)␣
 ↪& (UKMYC_PLATES.TRUST_PHENOTYPES)]

a=GOOD_PLATE_IMAGES[:3].apply(return_good_images,axis=1)
```

```
/well/bag/pfowler/cryptic/dat/CRyPTIC2/V2/01/DR0013/DR0013/1/14/01-DR0013-DR0013
-1-14-UKMYC6-growth.jpg
/well/bag/pfowler/cryptic/dat/CRyPTIC2/V2/01/DR0018/DR0018/1/14/01-DR0018-DR0018
-1-14-UKMYC6-growth.jpg
/well/bag/pfowler/cryptic/dat/CRyPTIC2/V2/01/DR0025/DR0025/1/14/01-DR0025-DR0025
-1-14-UKMYC6-growth.jpg
```

[34]:
```
pandas.crosstab(UKMYC_PLATES.PLATEDESIGN,UKMYC_PLATES.BELONGS_GPI)
```

[34]:
```
BELONGS_GPI  False  True
PLATEDESIGN
UKMYC5         732  7233
UKMYC6        4866  7806
```

The ratio of plate designs in the `GPI` is more like 50:50 since we haven't received the FASTQ files for some of the newer UKMYC6 samples.

The MD5SUM of the image is recorded here so duplicates can be identified. (Note that if `True` all of the image related measurements are discarded and therefore these measurements can never have a `PHENOTYPE_QUALITY` of `HIGH`).

```
[35]: UKMYC_PLATES.DUPLICATED_IMAGE.value_counts()
```

```
[35]: False    20174
      True       463
      Name: DUPLICATED_IMAGE, dtype: int64
```

Sometimes the images are duplicated across READINGDAYs for the same sample, which is lazy but not too bad. However, there are also many instances where the same image has been associated with different samples.

```
[36]: UKMYC_PLATES[UKMYC_PLATES.
       ↪IMAGE_MD5SUM=="7e0d4fce8ecc9f2c9c08f87098c3c85f"][["DUPLICATED_IMAGE"]]
```

```
[36]:                                      DUPLICATED_IMAGE
      UNIQUEID
      site.04.subj.00033.lab.628880.iso.1              True
      site.04.subj.00200.lab.634474.iso.1              True
      site.04.subj.01246.lab.719263.iso.1              True
```

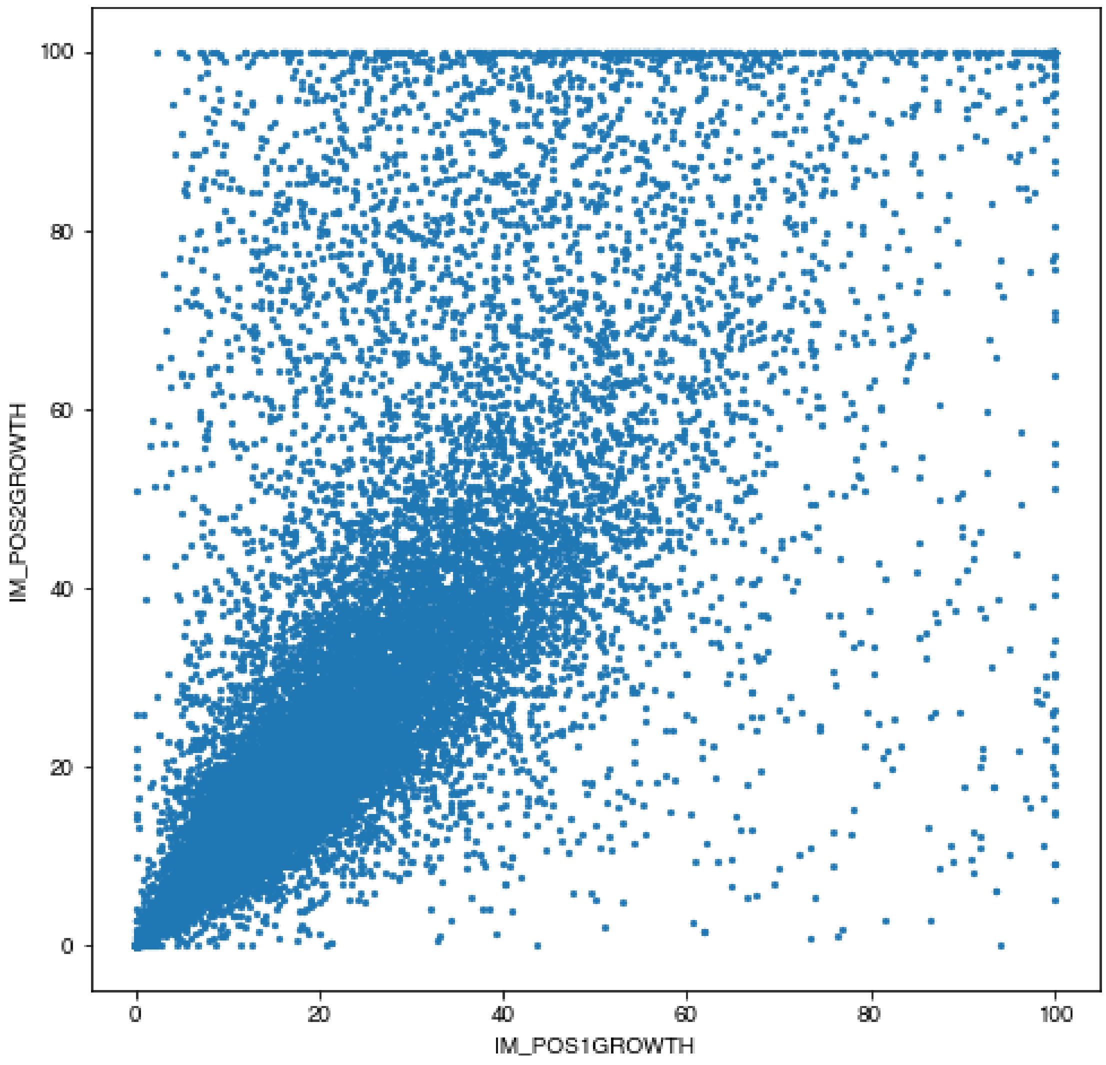
The IM_IMAGE_DOWNLOADED, IM_IMAGE_FILTERED and IM_WELLS_IDENTIFIED boolean fields tell you, respectively, if an image was downloaded, was successfully filtered by AMyGDA and whether 96 (and only 96!) wells were identified by AMyGDA. The last can fail if the image is improperly cropped or if the photo quality is so poor (e.g. washed out) that the algorithm cannot find the edges of the wells.

Note that this means having an image present, as indicated by IMAGEFILENAME does not guarantee that AMyGDA was also able to read it.

If AMyGDA was able to read the plate, then the growth in the two control wells (and their average for convenience) is recorded in the next 3 fields. As shown below the growth in the two control wells is correlated, but also truncated.

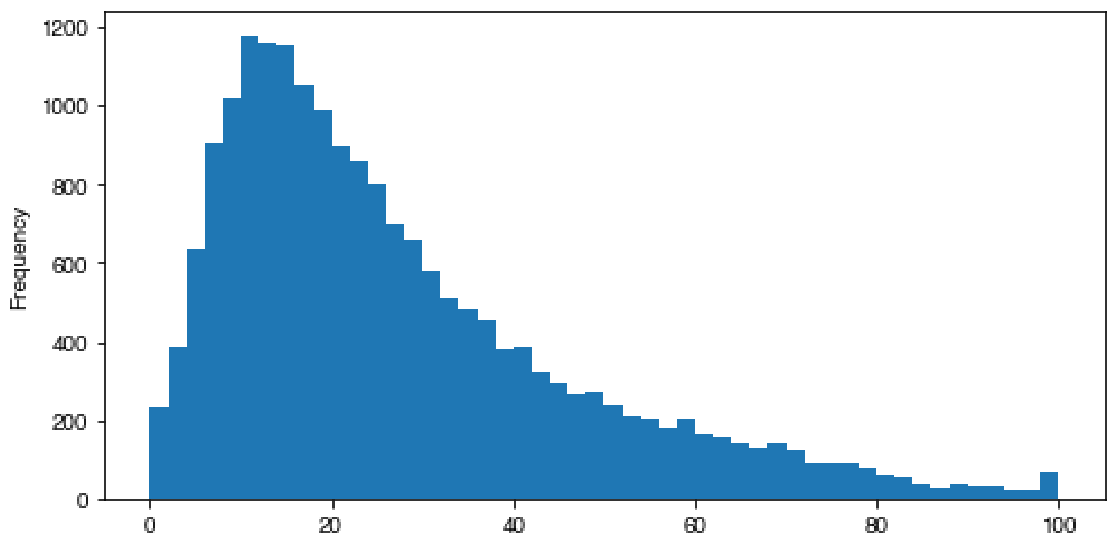
```
[37]: UKMYC_PLATES.plot.
       ↪scatter(x='IM_POS1GROWTH',y='IM_POS2GROWTH',figsize=(8,8),marker='.')
```

```
[37]: <matplotlib.axes._subplots.AxesSubplot at 0x12c99f190>
```



```
[38]: UKMYC_PLATES.IM_POS_AVERAGE.plot.hist(figsize=(8,4),bins=50)
```

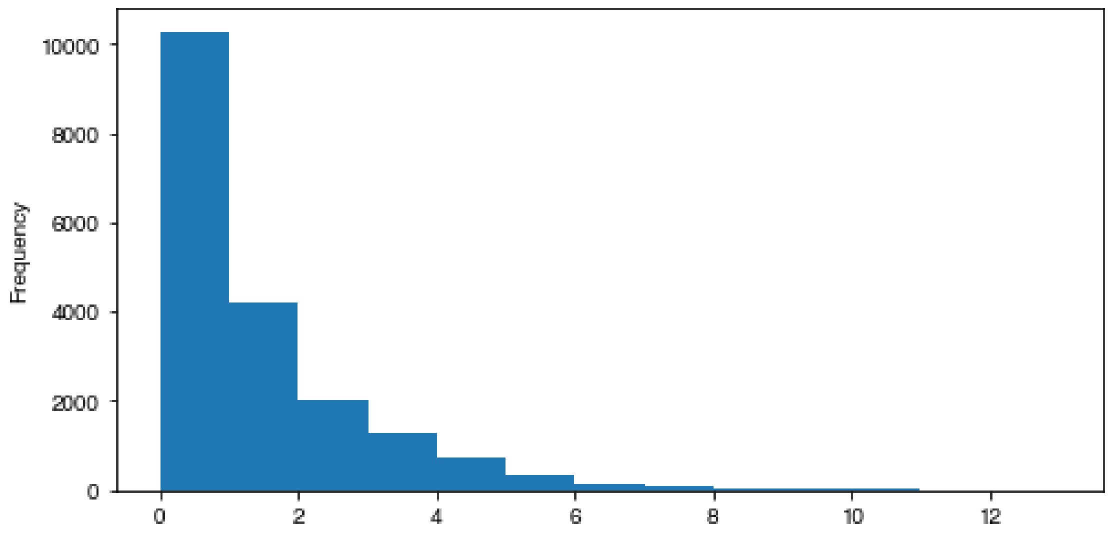
```
[38]: <matplotlib.axes._subplots.AxesSubplot at 0x12d5a9290>
```



The IM_DRUGS_INCONSISTENT_GROWTH field is the total number of drugs on this plate that AMyGDA was unable to read due to skipped wells etc. In theory a high number is a hint that this is a difficult plate to read. By definition, if AMyGDA cannot read the image, then that measurement will have been sent to BashTheBug for reading.

```
[39]: UKMYC_PLATES.IM_DRUGS_INCONSISTENT_GROWTH.plot.
       ↪hist(figsize=(8,4),bins=range(0,14))
```

```
[39]: <matplotlib.axes._subplots.AxesSubplot at 0x12d07c410>
```



Finally, the TRUST_PHENOTYPES column indicates which samples it is suspected are subject to systematic measurement error and therefore need to be excluded from the `UKMYC_PHENOTYPES` table. At present these samples are excluded * all of `SITEID=='13'` (very high number of discrepants recorded by Taiwan) * `02350>=SUBJID>='01575'` for `SITEID=='04'` (poor growth episode flagged by India) Note that `SUBJID>02350` for `SITEID==04` contains an elevated number of discrepancies and you may wish to also exclude these 1797 samples and see what the effect is on your analysis.

[40]:
```
UKMYC_PLATES.loc[~UKMYC_PLATES.TRUST_PHENOTYPES][:3]
```

[40]:
```
                                    SITEID SUBJID   LABID ISOLATENO  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1     04  01575  728967         1
site.04.subj.01576.lab.729539.iso.1     04  01576  729539         1
site.04.subj.01577.lab.725940.iso.1     04  01577  725940         1

                                    READINGDAY  BELONGS_GPI PLATEDESIGN  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1         14         True      UKMYC5
site.04.subj.01576.lab.729539.iso.1         14         True      UKMYC5
site.04.subj.01577.lab.725940.iso.1         14         True      UKMYC5

                                                               TREE_PATH  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1  dat/CRyPTIC2/V2/04/01575/728967/1/14/
site.04.subj.01576.lab.729539.iso.1  dat/CRyPTIC2/V2/04/01576/729539/1/14/
site.04.subj.01577.lab.725940.iso.1  dat/CRyPTIC2/V2/04/01577/725940/1/14/

                                            IMAGEFILENAME  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1  04-01575-728967-1-14
site.04.subj.01576.lab.729539.iso.1  04-01576-729539-1-14
site.04.subj.01577.lab.725940.iso.1  04-01577-725940-1-14

                                                        IMAGE_MD5SUM  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1  98e488b15b40f09bff1b84dbd3327d76
site.04.subj.01576.lab.729539.iso.1  57cd0d552bd53b48d7152428ada07c77
site.04.subj.01577.lab.725940.iso.1  4ce81fb9bf44d56d3b1fef2e0e504293

                                     DUPLICATED_IMAGE  IM_IMAGE_DOWNLOADED  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1             False                 True
site.04.subj.01576.lab.729539.iso.1             False                 True
site.04.subj.01577.lab.725940.iso.1             False                 True

                                     IM_IMAGE_FILTERED  IM_WELLS_IDENTIFIED  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1               True                 True
site.04.subj.01576.lab.729539.iso.1               True                    True
site.04.subj.01577.lab.725940.iso.1               True                    True

                                    IM_POS1GROWTH  IM_POS2GROWTH  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1          28.11          26.92
site.04.subj.01576.lab.729539.iso.1           7.62          10.57
site.04.subj.01577.lab.725940.iso.1           9.48          23.15

                                    IM_POS_AVERAGE  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1           27.52
site.04.subj.01576.lab.729539.iso.1            9.10
site.04.subj.01577.lab.725940.iso.1           16.31

                                    IM_DRUGS_INCONSISTENT_GROWTH  \
UNIQUEID
site.04.subj.01575.lab.728967.iso.1                           0.0
site.04.subj.01576.lab.729539.iso.1                           0.0
site.04.subj.01577.lab.725940.iso.1                           2.0

                                    TRUST_PHENOTYPES
UNIQUEID
site.04.subj.01575.lab.728967.iso.1            False
site.04.subj.01576.lab.729539.iso.1            False
site.04.subj.01577.lab.725940.iso.1            False
```

Although we start off with 15,211 in the GPI, we lose some since the plates are not readable. If we also exclude those which are under investigation due to high levels of discrepancies, then we reach 14,159!

```
[41]: pandas.crosstab(UKMYC_PLATES.TRUST_PHENOTYPES,UKMYC_PLATES.BELONGS_GPI)
```

```
[41]: BELONGS_GPI       False  True
      TRUST_PHENOTYPES
      False                62    880
      True               5536  14159
```

Because most of the plates excluded by `TRUST_PHENOTYPES` are UKMYC5, we end up at 45:55% for UKMYC5/6.

```
[42]: UKMYC_PLATES.loc[(UKMYC_PLATES.TRUST_PHENOTYPES) & (UKMYC_PLATES.BELONGS_GPI)].
       ↪PLATEDESIGN.value_counts()
```

```
[42]: UKMYC6    7804
      UKMYC5    6355
      Name: PLATEDESIGN, dtype: int64
```

Of these, 12,984 (92%) have images.

```
[43]: len(UKMYC_PLATES.loc[(UKMYC_PLATES.TRUST_PHENOTYPES) & (UKMYC_PLATES.
       ↪BELONGS_GPI) & (UKMYC_PLATES.IM_POS_AVERAGE.notna())])
```

```
[43]: 12984
```

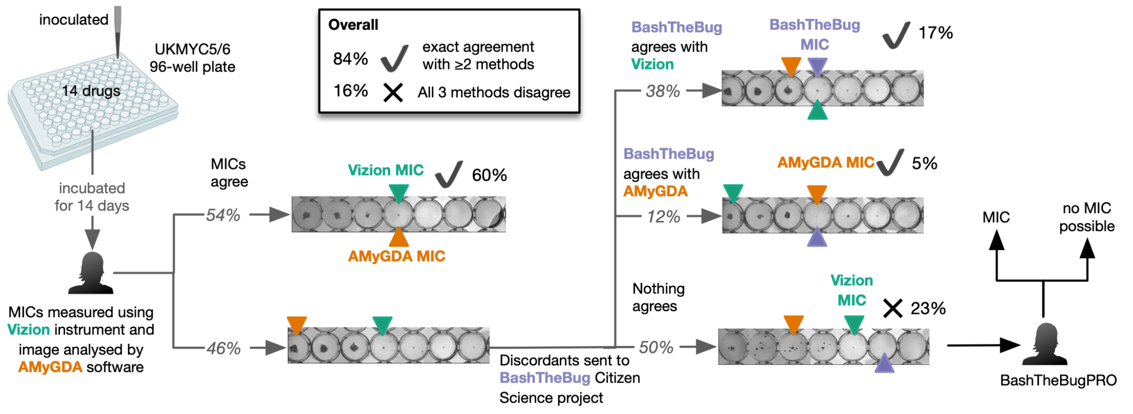
3.4 UKMYC_PHENOTYPES

This is the **core** phenotype table. It aggregates and summarises all the above and presents the current 'best valid reading' on each DRUG for each UNIQUEID.

```
[44]: UKMYC_PHENOTYPES=pandas.read_pickle(TABLES_PATH+"UKMYC_PHENOTYPES.pkl.gz")
      UKMYC_PHENOTYPES[:3]
```

```
[44]:                                              PLATEDESIGN  BELONGS_GPI SITEID  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD       UKMYC5        False     06
site.04.subj.00861.lab.713588.iso.1     ETH       UKMYC5         True     04
site.08.subj.24TB00-059.lab.2444.iso.1  RFB       UKMYC5         True     08

                                              DILUTION PHENOTYPE_QUALITY  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD        5.0            MEDIUM
site.04.subj.00861.lab.713588.iso.1     ETH        2.0              HIGH
site.08.subj.24TB00-059.lab.2444.iso.1  RFB        1.0              HIGH

                                             READINGDAY  PRIMARY_DILUTION  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD          14               5.0
site.04.subj.00861.lab.713588.iso.1     ETH          14               2.0
site.08.subj.24TB00-059.lab.2444.iso.1  RFB          14               1.0

                                             PRIMARY_METHOD  AMYGDA_DILUTION  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD              VZ              NaN
site.04.subj.00861.lab.713588.iso.1     ETH              VZ              2.0
site.08.subj.24TB00-059.lab.2444.iso.1  RFB              VZ              1.0

                                              BASHTHEBUG_DILUTION  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD                   NaN
site.04.subj.00861.lab.713588.iso.1     ETH                   3.0
site.08.subj.24TB00-059.lab.2444.iso.1  RFB                   1.0

                                              BASHTHEBUGPRO_DILUTION  \
UNIQUEID                                DRUG
```

```
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD                         NaN
site.04.subj.00861.lab.713588.iso.1     ETH                         NaN
site.08.subj.24TB00-059.lab.2444.iso.1  RFB                         NaN

                                             PHENOTYPE_DESCRIPTION  \
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD               VZ ONLY
site.04.subj.00861.lab.713588.iso.1     ETH           VZ,IM AGREE
site.08.subj.24TB00-059.lab.2444.iso.1  RFB           VZ,IM AGREE

                                              BASHTHEBUG_NUMBER_CLASSIFICATIONS
\
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD                                 NaN
site.04.subj.00861.lab.713588.iso.1     ETH                                11.0
site.08.subj.24TB00-059.lab.2444.iso.1  RFB                                11.0

                                                MIC  LOG2MIC BINARY_PHENOTYPE
UNIQUEID                                DRUG
site.06.subj.CL4441.lab.06MIL0824.iso.1 LZD     0.5    -1.00                S
site.04.subj.00861.lab.713588.iso.1     ETH     0.5    -1.00                S
site.08.subj.24TB00-059.lab.2444.iso.1  RFB  <=0.06    -4.06                S
```


inoculated
UKMYC5/6 96-well plate
14 drugs
incubated for 14 days
MICs measured using Vizion instrument and image analysed by AMyGDA software
Overall
84% ✓ exact agreement with ≥2 methods
16% ✗ All 3 methods disagree
MICs agree
54%
Vizion MIC ✓ 60%
AMyGDA MIC
46%
Discordants sent to BashTheBug Citizen Science project
BashTheBug agrees with Vizion
38%
BashTheBug MIC ✓ 17%
BashTheBug agrees with AMyGDA
12%
AMyGDA MIC ✓ 5%
Nothing agrees
50%
Vizion MIC ✗ 23%
MIC
no MIC possible
BashTheBugPRO

Each reading flows down the above quality assurance process; first a reading (PRIMARY_DILUTION) is recorded in the laboratory by the scientist using the PRIMARY_METHOD. This is almost always VZ i.e. Vizion, but some labs were only able to use Mirrored Box (MB) for some measurememts. This is almost always after 14 days of incubation (READINGDAY==14), but if a reading was not possible, we then consider the day 21 reading, if available.

Note that we use negative DILUTIONs to indicate there was a problem with that reading (the MIC will be NaN).

- -1 cannot read this particular drug for some reason (but usually can read the others off a plate). For AMyGDA (IM) this indicates the presence of one or more skip wells.
- -2 no or insufficient growth in one or both of the control wells

If no photograph of the plate was stored, or the image was subsequently duplicated (as indicated in UKMYC_PLATES), then only one reading is possible, the PRIMARY_DILUTION as done by the PRIMARY_METHOD. In these cases, the PHENOTYPE_QUALITY is left as the default, which is MEDIUM.

About 10.3% of all readings have no image.

```
[45]: len(UKMYC_PHENOTYPES.loc[UKMYC_PHENOTYPES.PHENOTYPE_QUALITY=="MEDIUM"])/
       ↪len(UKMYC_PHENOTYPES)
```

```
[45]: 0.1027750948734134
```

The remaining measurements do have a (not duplicated) image. Each image is analysed by AMyGDA and the dilution recorded in AMYGDA_DILUTION. In about 54% of cases, this exactly agrees with the PRIMARY_DILUTION and hence this measurement is marked as PHENOTYPE_QUALITY='HIGH'.

The remaining 46% are sent to BashTheBug for assessment by citizen scientists. Once BASHTHEBUG_NUMBER_CLASSIFICATIONS>=11 the media value is returned as the consensus and populated in BASHTHEBUG_DILUTION.

Of these, in about

- 38% of cases, BASHTHEBUG_DILUTION and PRIMARY_DILUTION are identical, suggesting that AMyGDA incorrectly read the plate (due to e.g. low growth or artefacts).
- 12% of cases BASHTHEBUG_DILUTION and AMYGDA_DILUTION are identical, suggesting that the laboratory scientist made a measurement or data entry error.
- 50% of cases, all three measurements are different.

If two measurements exactly agree, then the measurement is marked as PHENOTYPE_QUALITY='HIGH', otherwise if all three disgree, then PHENOTYPE_DESCRIPTION='ALL DISAGREE' and the PHENOTYPE_QUALTITY is marked as LOW.

It is recommended that, unless you have a good reason to the contrary, to only use readings where PHENOTYPE_QUALTIY is HIGH. We would be very interested in knowing what, if any, the effect of this QA workflow is, so would also be interested in seeing the effect of ignoring the PHENOTYPE_QUALITY i.e. just using the PRIMARY_DILUTION.

```
[46]: UKMYC_PHENOTYPES.PHENOTYPE_DESCRIPTION.value_counts()
```

```
[46]: VZ,IM AGREE     128038
      ALL DISAGREE     54419
      VZ,BB AGREE      40726
      VZ ONLY          26684
      BB,IM AGREE      12771
      BB RUNNING         344
      Name: PHENOTYPE_DESCRIPTION, dtype: int64
```

```
[47]: pandas.crosstab(UKMYC_PHENOTYPES.PHENOTYPE_DESCRIPTION,UKMYC_PHENOTYPES.
       ↪PHENOTYPE_QUALITY,margins=True)
```

[47]:
```
PHENOTYPE_QUALITY        HIGH    LOW  MEDIUM     All
PHENOTYPE_DESCRIPTION
ALL DISAGREE                0  54419       0   54419
BB RUNNING                  0      0     344     344
BB,IM AGREE             12771      0       0   12771
VZ ONLY                     0      0   26684   26684
VZ,BB AGREE             40726      0       0   40726
VZ,IM AGREE            128038      0       0  128038
All                    181535  54419   27028  262982
```

As mentioned above, the 344 rows where BashTheBug does not appear to have finished are glitches and will remain for the time being at least. 293 are for PAS which is excluded in all analyses.

That leaves 26,684 (10% of total) measurements where there is no image (or the image was a duplicate) and so only one measurement (usually `VZ`) is possible and hence these cannot progress any further than a `PHENOTYPE_QUALITY` of `MEDIUM`.

Of the remaining 233,948 measurements, 77% have two or more measurement methods (`VZ/IM/BB`) in exact concordance and therefore are classified as having a `HIGH PHENOTYPE_QUALITY`. In 70.8% of these, the AMyGDA measurement agreed with Vizion whilst in 22.2% and 7.0% of cases BashTheBug agreed with Vizion or AMyGDA, respectively. The latter set contain mistakes made by the laboratory scientist and therefore is an upper estimate of the laboratory reading error rate.

[48]:
```
UKMYC_PHENOTYPES.loc[UKMYC_PHENOTYPES.PHENOTYPE_QUALITY=='HIGH'].
 ↪PHENOTYPE_DESCRIPTION.value_counts(normalize=True)
```

[48]:
```
VZ,IM AGREE     0.705308
VZ,BB AGREE     0.224342
BB,IM AGREE     0.070350
VZ ONLY         0.000000
BB RUNNING      0.000000
ALL DISAGREE    0.000000
Name: PHENOTYPE_DESCRIPTION, dtype: float64
```

There are 53,129 rows where all three methods disagree. The last time the analysis was run 79% of these have been processed by `BashTheBugPRO`. (The figure stands now at 90% - 12 Aug 2020)

In future, the number of `HIGH` quality measurements will be increased by allowing their consensus to overrule i.e. they will arbitrate. The may choose to overrule the three methods and choose an MIC, or they may decide that the image is not readable.

[49]:
```
df=UKMYC_PHENOTYPES.loc[(UKMYC_PHENOTYPES.PHENOTYPE_QUALITY=="LOW")]


len(df.loc[UKMYC_PHENOTYPES.BASHTHEBUGPRO_DILUTION.notna()])/len(df)
```

[49]:
```
0.7882357264925853
```

Note also, that the volunteers also finished looking at the images where `PRIMARY_DILUTION` and `AMYGDA_DILUTION` agree and hence these rows will have a `BASHTHEBUG_DILUTION` reading even though it does not affect the final DILUTION. It is provided mainly for Machine Learning from the images using the classifications as input features.

3.5 GROWTH

A useful by-product of reading all the images of plates with AMyGDA is that we measure the percentage of growth in the centre of each well on every plate we have an image for. This is stored here in a (long) table.

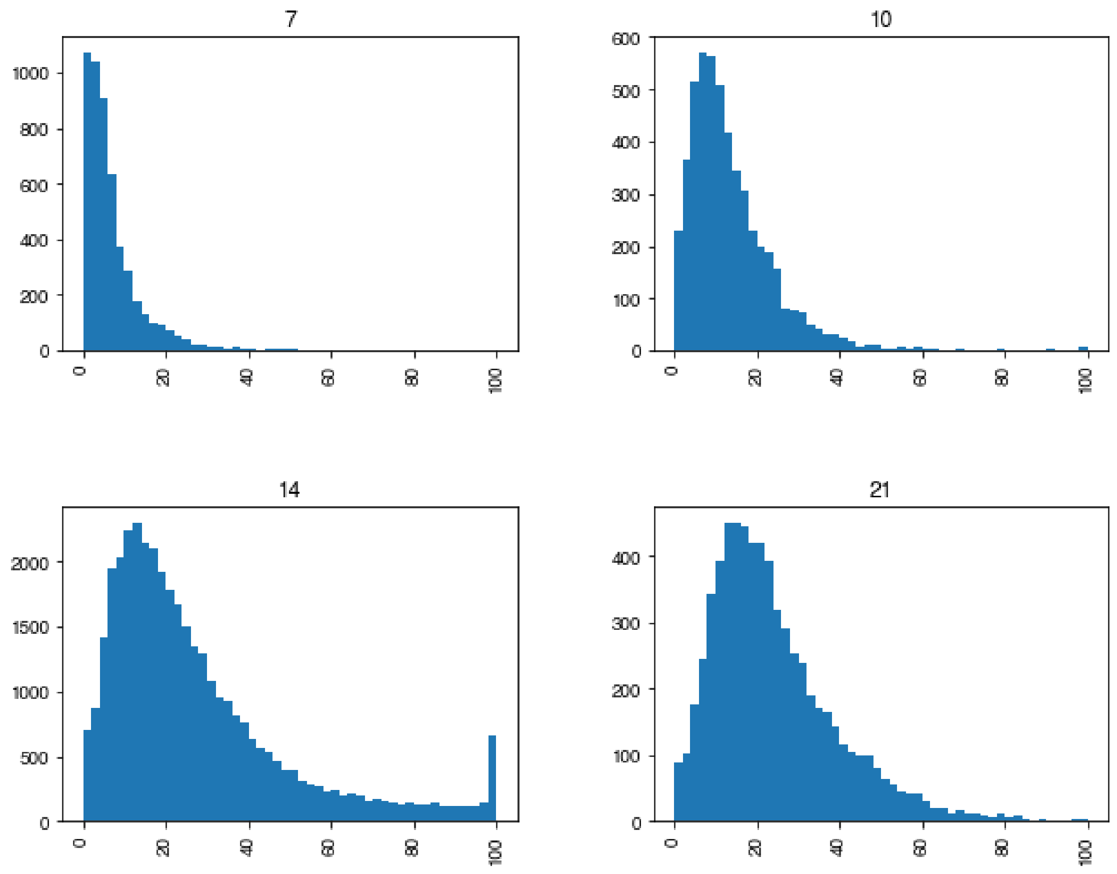
[50]:
```
GROWTH=pandas.read_pickle(TABLES_PATH+"UKMYC_GROWTH.pkl.gz")
print(len(GROWTH))
GROWTH[:3]
```

```
2609184
```

[50]:
```
                                                           PLATEDESIGN  \
UNIQUEID                           READINGDAY DRUG DILUTION
site.01.subj.DR0013.lab.DR0013.iso.1 14          AMI  1             UKMYC6
                                                      2             UKMYC6
                                                      3             UKMYC6

                                                           SITEID  \
UNIQUEID                           READINGDAY DRUG DILUTION
site.01.subj.DR0013.lab.DR0013.iso.1 14          AMI  1            01
                                                      2            01
                                                      3            01

                                                           WELL_CONC  \
UNIQUEID                           READINGDAY DRUG DILUTION
site.01.subj.DR0013.lab.DR0013.iso.1 14          AMI  1             0.25
                                                      2             0.50
                                                      3             1.00

                                                              GROWTH
UNIQUEID                           READINGDAY DRUG DILUTION
site.01.subj.DR0013.lab.DR0013.iso.1 14          AMI  1          9.554551
                                                      2          0.270088
                                                      3          0.129116
```

Let's quickly look at the distribution of measured growth in the control wells over time

[51]:
```
GROWTH.reset_index(inplace=True)
df=GROWTH.loc[(GROWTH.DRUG=="POS") & (GROWTH.DILUTION==0)]
a=df['GROWTH'].hist(by=df.READINGDAY,figsize=(10,8),bins=50)
```



Note that there is a bias here; it is likely labs only allowed plates to incubate to 21 days if their growth at 14 days was poor so you cannot directly compare the two. We obviously could only show histograms for plates that were read at day 21 but that isn't what is shown here.

4 Genotype data

In this section I will quickly run through the tables containing the genetics information.

4.1 `GENOMES`

These contain one row per VCF file, i.e. one row per successful Clockwork output. Successful means the cluster process did not fail and all quality control checks were passed. Hence some samples have been excluded since they did not pass the overall genetic quality control metrics.

Remember several sets of short-reads could have been combined and processed together if `seq_rep` is something like `1_2_3`. This table is a join of what used to be two separate tables (`GENOMES` and `VCF_FILES`).

```
[52]: GENOMES=pandas.read_pickle(TABLES_PATH+"GENOMES.pkl.gz")
      GENOMES[:3]
```

```
[52]:                                        SITEID SUBJID       LABID  ISOLATENO  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1         02   0958      22A197          1
      site.02.subj.0823.lab.2013241494.iso.1     02   0823  2013241494          1
      site.02.subj.0359.lab.222018-14.iso.1      02   0359   222018-14          1

                                               SEQREPS  BELONGS_GPI  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1           197         True
      site.02.subj.0823.lab.2013241494.iso.1    241494         True
      site.02.subj.0359.lab.222018-14.iso.1   14222018         True

                                              PER_SAMPLE_VCF_PRESENT  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1                        True
      site.02.subj.0823.lab.2013241494.iso.1                    True
      site.02.subj.0359.lab.222018-14.iso.1                     True

                                              REGENOTYPED_VCF_PRESENT  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1                         True
      site.02.subj.0823.lab.2013241494.iso.1                     True
      site.02.subj.0359.lab.222018-14.iso.1                      True

                                             CLOCKWORK_VERSION  TBI_INDEX  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1                  0.8.3       True
      site.02.subj.0823.lab.2013241494.iso.1              0.8.3       True
      site.02.subj.0359.lab.222018-14.iso.1               0.8.3       True

                                              KMER_COUNTS  SNP_DISTANCE_TO_H37rV  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1            False                 1154.0
      site.02.subj.0823.lab.2013241494.iso.1        False                  388.0
      site.02.subj.0359.lab.222018-14.iso.1         False                 1147.0

                                                      SPECIES LINEAGE_NAME  \
      UNIQUEID
      site.02.subj.0958.lab.22A197.iso.1      M. tuberculosis    Lineage 2
      site.02.subj.0823.lab.2013241494.iso.1  M. tuberculosis    Lineage 4
      site.02.subj.0359.lab.222018-14.iso.1   M. tuberculosis    Lineage 2

                                             SUBLINEAGE_NAME  LINEAGE_PERCENTAGE  \
      UNIQUEID
```

```
site.02.subj.0958.lab.22A197.iso.1                              71.283784
site.02.subj.0823.lab.2013241494.iso.1                          95.705521
site.02.subj.0359.lab.222018-14.iso.1                           95.608108

                                        N_NULL  N_SNP  N_INDEL  N_FILTER_FAIL  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1        4934      0     1154            104
site.02.subj.0823.lab.2013241494.iso.1    2250      0      388             43
site.02.subj.0359.lab.222018-14.iso.1     3578      0     1147            118

                                        N_REF  N_HET CATALOGUE_NAME  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1          0  14329        CRyPTIC
site.02.subj.0823.lab.2013241494.iso.1      0   9442        CRyPTIC
site.02.subj.0359.lab.222018-14.iso.1       0  13038        CRyPTIC

                                       CATALOGUE_VERSION TB_TYPE_1  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1                 v1.31       MDR
site.02.subj.0823.lab.2013241494.iso.1             v1.31       UNK
site.02.subj.0359.lab.222018-14.iso.1              v1.31       UNK

                                       WGS_PREDICTION_STRING  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1           RRURRRSSSSSSSSS
site.02.subj.0823.lab.2013241494.iso.1       UUSUSSSSSSSSSSS
site.02.subj.0359.lab.222018-14.iso.1        SUSSSSSSUSSSSSS

IMAGE_MD5SUM  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1      {'02-0958-22A197-1-14':
'a587bac9ad2a0ebd36274…
site.02.subj.0823.lab.2013241494.iso.1  {'02-0823-2013241494-1-14':
'698507bed7ff19268…
site.02.subj.0359.lab.222018-14.iso.1   {'02-0359-222018-14-1-14':
'39c28529c7564ce379…

                                                             FTP_PATH  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1      /well/bag/jeffk/release_staging/
site.02.subj.0823.lab.2013241494.iso.1  /well/bag/jeffk/release_staging/
site.02.subj.0359.lab.222018-14.iso.1   /well/bag/jeffk/release_staging/

FTP_FILENAME_VCF  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1
00/01/41/00/14100/site.02.iso.1.subject.0958.l…
```

```
site.02.subj.0823.lab.2013241494.iso.1
00/01/41/43/14143/site.02.iso.1.subject.0823.l…
site.02.subj.0359.lab.222018-14.iso.1
00/01/08/73/10873/site.02.iso.1.subject.0359.l…

TREE_PATH  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1
dat/CRyPTIC2/V2/02/0958/22A197/1/regenotyped/
site.02.subj.0823.lab.2013241494.iso.1
dat/CRyPTIC2/V2/02/0823/2013241494/1/regenotyped/
site.02.subj.0359.lab.222018-14.iso.1
dat/CRyPTIC2/V2/02/0359/222018-14/1/regenotyped/

TREE_FILENAME_VCF  \
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1
site.02.subj.0958.lab.22A197.iso.1.v0.8.3.rege…
site.02.subj.0823.lab.2013241494.iso.1
site.02.subj.0823.lab.2013241494.iso.1.v0.8.3…
site.02.subj.0359.lab.222018-14.iso.1
site.02.subj.0359.lab.222018-14.iso.1.v0.8.3.r…

                                       FASTQ_MD5SUMS
UNIQUEID
site.02.subj.0958.lab.22A197.iso.1
site.02.subj.0823.lab.2013241494.iso.1
site.02.subj.0359.lab.222018-14.iso.1
```

[53]:
```
len(GENOMES)
```

[53]:
```
70372
```

Since some of the later tables are VERY large, we also define a subset of 600 samples via this table. These were randomly chosen and so contain a mixture of sites as well as GPI/non-GPI etc.

[54]:
```
GENOMES_SAMPLE=pandas.read_pickle('GENOMES_SAMPLE.pkl.gz')
len(GENOMES_SAMPLE)
```

[54]:
```
600
```

As for `UKMYC_PLATES`, the hierarchical metadata fields `SITEID`, `SUBJID`, `LABID`, `ISOLATENO`, `SEQREPS` are included, the latter being specific to genetics.

The same Boolean flag, `BELONGS_GPI` is also included here. This was used to decided which samples should be regenotyped (and which should not). Not all samples could be regenotyped due to the large memory requirements of the process. Martin Hunt is attempting to regenotype all 62k samples but this may not work.

If a sample was regenotyped, a ‘normal’ per-sample VCF file was also generated and hence samples with `BELONG_GPI==True` have 2 VCF files. The presence or absence of both types of files is indicated with `PER_SAMPLE_VCF_PRESENT` and `REGENOTYPED_VCF_PRESENT`.

If a sample `BELONGS_GPI` then both VCF files are processed in the sharded data tree and both have mini `VARIANTS` and `MUTATIONS` tables stored (thereby enabling potential comparisons between the two approaches). When constructing these tables, we take the approach of using the regenotyped `VCF` for the `BELONGS_GPI` samples and the per-sample `VCF` data for the remainder.

The exception is the 17 quality control samples from Comas and Gagneux; these have only been through the regenotyping process and hence have no per-sample `VCF`. These are also the reason why there are 15228=15211+17 samples with `BELONGS_GPI` in the `GENOMES` table. They can be identified by `SITEID=='QC'`.

Since the regenotyped VCF files are large, they often have an attendant `.tbi` index file. The presence of this is noted with `TBI_INDEX`. Likewise the presence of `kmer-counts.txt.gz` files are noted with `KMER_COUNTS`. These files have now been sorted as requested by Alex Lachapelle.

Again, like in `UKMYC_PLATES`, the path to the correct folder in the sharded tree is given by `TREE_PATH` and the filename of the VCF file is stored in `TREE_FILENAME_VCF`. Using the above Boolean flags, one can also construct paths to all the other files (e.g. kmer counts for machine learning).

Finally, the clockwork version is stored and also the md5sums for the `FASTQ` files are stored as `JSON` in `FASTQ_MD5SUMS`. This was necessary since there could be multiple pairs if there are multiple `SEQ_REPS` for this sample. For information, the path to the original vcf file provided by Jeff can be parsed from `(FTP_PATH,FTP_FILENAME_VCF)`.

```
[55]: pandas.crosstab(GENOMES.BELONGS_GPI,GENOMES.REGENOTYPED_VCF_PRESENT)
```

```
[55]: REGENOTYPED_VCF_PRESENT  False  True
      BELONGS_GPI
      False                    55144      0
      True                         0  15228
```

```
[56]: pandas.crosstab(GENOMES.BELONGS_GPI,GENOMES.PER_SAMPLE_VCF_PRESENT)
```

```
[56]: PER_SAMPLE_VCF_PRESENT  False  True
      BELONGS_GPI
      False                       0  55144
      True                       17  15211
```

This is the number of VCFs we have per site

```
[57]: GENOMES.SITEID.value_counts().sort_index()
```

```
[57]: 00      1615
      01       136
      02      1090
      03      1837
      04      4393
```

```
05      2930
06      2368
07     10435
08      1291
10      2662
11       463
13       286
14       389
16        69
17        91
20       454
21         7
ENA    39839
QC        17
Name: SITEID, dtype: int64
```

All the `BELONGS_GPI` VCFs have a regenotyped VCF been passed through Sam Lipworth's SNPIT (this version) and hence this tables contains `SPECIES`, `LINEAGE_NAME` and, where provided (mostly for Lineage 4), `SUBLINEAGE_NAME`. Finally the `LINEAGE_PERCENTAGE` is also given. If no regenotyped VCF is present, then these are all nulls.

Pleasingly, all samples belong to the *M. tuberculosis* complex!

```
[58]: GENOMES.loc[GENOMES.BELONGS_GPI].SPECIES.value_counts().sort_index()
```

```
[58]:                        0
      M. bovis BCG           8
      M. bovis bovis         3
      M. bovis caprae        1
      M. orygis             10
      M. tuberculosis    15206
      Name: SPECIES, dtype: int64
```

Of those predicted to be *M. tuberculosis*, the majority are Lineage 2 and 4, as expected.

```
[59]: GENOMES[GENOMES.SPECIES=="M. tuberculosis"].LINEAGE_NAME.value_counts().
       ↪sort_index()
```

```
[59]:                0
      Lineage 1   1107
      Lineage 2   5527
      Lineage 3   1837
      Lineage 4   6724
      Lineage 5      2
      Lineage 6      9
      Name: LINEAGE_NAME, dtype: int64
```

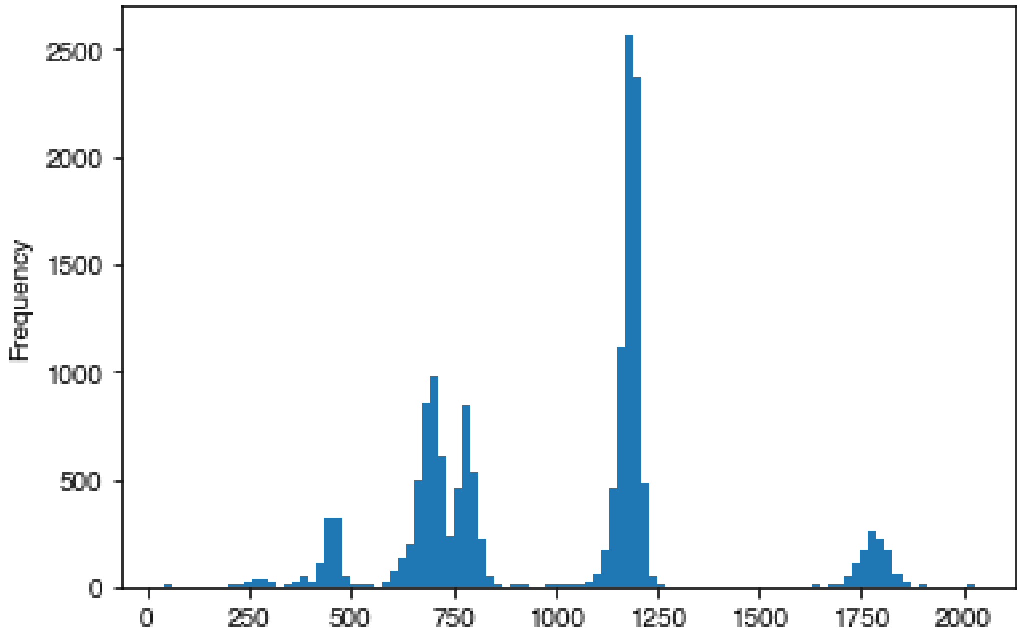
Some sublineage information is available for Lineage 4

```
[60]: GENOMES[GENOMES.LINEAGE_NAME=="Lineage 4"].SUBLINEAGE_NAME.value_counts().
       ↪sort_values(ascending=False)
```

```
[60]: LAM          2235
                   1554
      Haarlem      1344
      X-type        557
      S-type        524
      Tur           280
      Cameroon      118
      Ural           81
      Ghana          16
      Uganda         15
      Name: SUBLINEAGE_NAME, dtype: int64
```

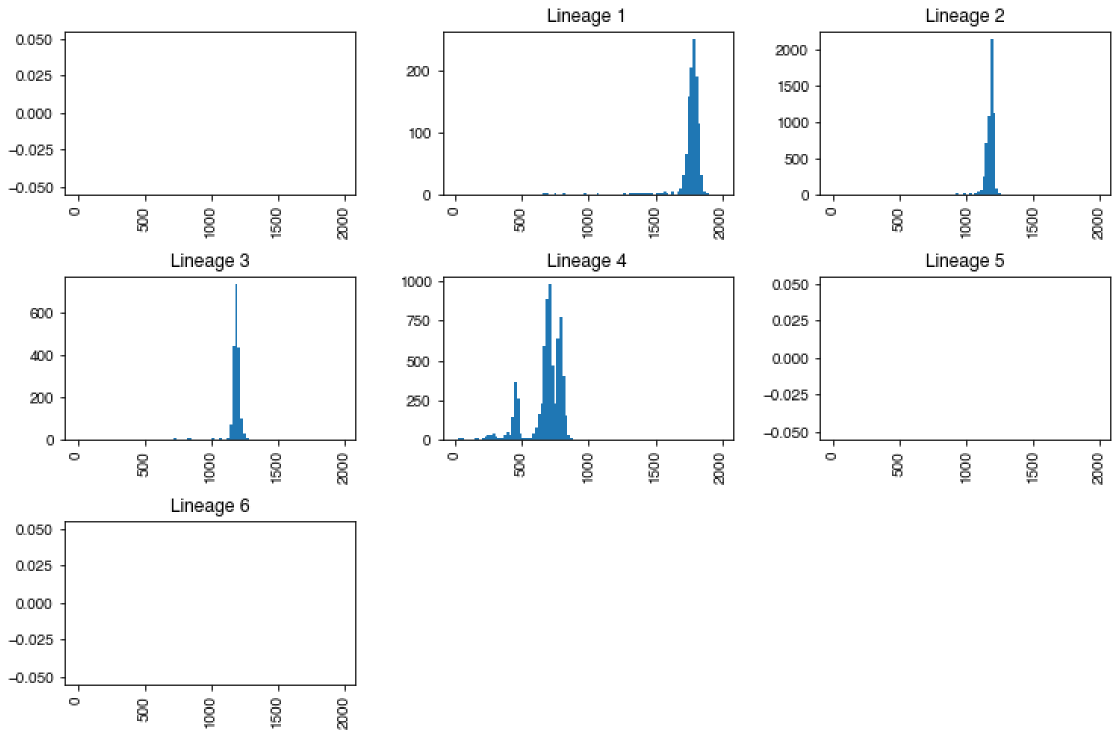
Again only for the `BELONGS_GPI` samples which have regenotyped VCFs, the number of SNPs to the H37rV version 3 reference is recorded in `SNP_DISTANCE_TO_H37rV` and we observe a number of peaks.

```
[61]: a=GENOMES[GENOMES.SPECIES=="M. tuberculosis"].SNP_DISTANCE_TO_H37rV.plot.
       ↪hist(bins=100)
```



Pleasingly, the different lineages explain the different peaks (although Lineages 2 & 3 overlap which just means they tend to be the same distance from the reference not that they are similar)

[62]:
```
df=GENOMES[(GENOMES.SPECIES=="M. tuberculosis") & (GENOMES.LINEAGE_NAME.
 ↪isin(['Lineage 1','Lineage 2','Lineage 3','Lineage 4']))]
a=df['SNP_DISTANCE_TO_H37rV'].hist(by=df.
 ↪LINEAGE_NAME,bins=range(0,2000,20),figsize=(12,8))
```



Then we record some very high-level information about the antibiogram predicted from the default genetic catalogue.

[63]:
```
GENOMES.CATALOGUE_NAME.value_counts()
```

[63]:
```
CRyPTIC    70372
Name: CATALOGUE_NAME, dtype: int64
```

[64]:
```
GENOMES.CATALOGUE_VERSION.value_counts()
```

[64]:
```
v1.31    70372
Name: CATALOGUE_VERSION, dtype: int64
```

- TB_TYPE_1 provides a high-level description of the degree of resistance, based on the genetics. It shouldn't be relied upon but gives you an idea of the level of resistance.
- SUS susceptible according to the Bayesian approach in NEJM2018
- XDR resistant to RIF, INH, one of LEV or MXF and one of AMI or KAN
- MDR resistant to RIF, INH

- RIF resistant to RIF, susceptible to INH
- UNK everything else

```
[65]: GENOMES.loc[GENOMES.BELONGS_GPI].TB_TYPE_1.value_counts()
```

```
[65]: SUS    6665
      MDR    5429
      UNK    1896
      XDR     680
      RIF     558
      Name: TB_TYPE_1, dtype: int64
```

`WGS_PREDICTION_STRING` is simply the genetic antibiogram as a single string. The order of drugs is currently

`["RIF","INH","PZA","EMB","AMI","KAN","LEV","MXF","ETH","PAS","RFB","LZD","BDQ","DLM","CFZ"]`

Hence you can use slice the first four characters to compare to the NEJM paper. This is only supposed to be a hint and isn't definitive. For that please use the `PREDICTIONS` and `EFFECTS` tables which are described later.

4.2 VARIANTS

`VARIANTS` is a very long table since it contains all SNPs and INDELs detected by Clockwork across the whole genome for all samples, including those retrieved from the ENA. It contains calls parsed from both regenotyped and per-sample `vcf` files depending on what is available for that sample.

There is an approximate 1:1 mapping between a row in the VCF file making a call, and a row in this table. There are occasions where a row in the VCF can parsed as containing e.g. two SNPs if `REF='cttcg'` and `ALT='ttttg'` and hence a single row in the VCF will map to two rows in the table.

It is not stored in `cryptic-tables/` since the gzipped `CSV` and `PKL` files are 2.8 GB and 1.6 GB respectively for the ~62k samples. We STRONGLY recommend you work with the `PKL` format since, unlike `CSV`, some of the fields are stored as categories which dramatically reduces the memory required. A table only containing the rows for the GPI samples is also included (`VARIANTS_GPI`: 18,159,378 rows : 326 MB and 623 MB).

Both tables are available on request, however for demonstration and testing purposes, the rows for the 600 randomly selected samples (1%) defined above in `GENOMES_SAMPLE` are stored in separate tables in `cryptic-tables/`. The whole table has currently (Aug 2020) 91,853,092 rows.

Since the regenotyped `VCF` files in particular yield large numbers of variants, we have adopted a 'mixed' approach; all SNPs and INDELs are recorded for all genes, however null calls and filter fails are only recorded for genes in the current resistance catalogue. To be clear a filter fail is a putative call that has not passed the quality and statistical thresholds set by Clockwork. The reason these are included for these genes is that one would like to know when predicting resistance e.g. one would treat a null or filter fail at position Ser450 in *rpoB* differently to a reference/wildtype call.

```
[66]: VARIANTS_SAMPLE=pandas.read_pickle(TABLES_PATH+"VARIANTS_SAMPLE.pkl.gz")
      VARIANTS_SAMPLE[:3]
```

```
[66]:                                                 IS_SNP REF ALT  GENOME_INDEX  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g    True   c   g          1565
                                            1977a>g    True   a   g          1977
                                            4013t>c    True   t   c          4013

                                                     GENE ELEMENT_TYPE  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g
                                            1977a>g  dnaN          GENE
                                            4013t>c  recF          GENE

                                                    MUTATION_TYPE  POSITION  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g           SNP       NaN
                                            1977a>g           SNP     -75.0
                                            4013t>c           SNP     245.0

                                                     NUCLEOTIDE_NUMBER  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g                NaN
                                            1977a>g              -75.0
                                            4013t>c              734.0

                                                     AMINO_ACID_NUMBER  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g                NaN
                                            1977a>g                NaN
                                            4013t>c              245.0

                                                     ASSOCIATED_WITH_GENE  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g                  False
                                            1977a>g                   True
                                            4013t>c                   True

                                                     IN_PROMOTER  IN_CDS  IS_INDEL  \
      UNIQUEID                             VARIANT
      site.02.subj.0345.lab.234051-15.iso.1 1565c>g        False   False     False
                                            1977a>g         True   False     False
                                            4013t>c        False    True     False

                                                     IS_HET  IS_NULL  \
      UNIQUEID                             VARIANT
```

```
site.02.subj.0345.lab.234051-15.iso.1 1565c>g   False    False
                                      1977a>g   False    False
                                      4013t>c   False    False

                                                 IS_FILTER_PASS  INDEL_LENGTH  \
UNIQUEID                              VARIANT
site.02.subj.0345.lab.234051-15.iso.1 1565c>g              True           NaN
                                      1977a>g              True           NaN
                                      4013t>c              True           NaN

                                                INDEL_1 INDEL_2     DP  \
UNIQUEID                              VARIANT
site.02.subj.0345.lab.234051-15.iso.1 1565c>g     None    None  238.0
                                      1977a>g     None    None  240.0
                                      4013t>c     None    None  212.0

                                                 COVERAGE     DPF  FRS  \
UNIQUEID                              VARIANT
site.02.subj.0345.lab.234051-15.iso.1 1565c>g         238  1.1944  1.0
                                      1977a>g         240  1.2044  1.0
                                      4013t>c         212  1.0639  1.0

                                                     GT_CONF  \
UNIQUEID                              VARIANT
site.02.subj.0345.lab.234051-15.iso.1 1565c>g    1870.410034
                                      1977a>g    1882.469971
                                      4013t>c    1712.030029

                                                 GT_CONF_PERCENTILE SITEID
UNIQUEID                              VARIANT
site.02.subj.0345.lab.234051-15.iso.1 1565c>g             88.739998     02
                                      1977a>g             89.720001     02
                                      4013t>c             67.720001     02
```

[67]:
```
len(VARIANTS_SAMPLE)
```

[67]:
```
676942
```

To help with some of the graphs below, let's join to the `GENOMES_SAMPLE` table so we can add the `BELONGS_GPI` column.

[68]:
```
def assign_gpi_description(row):
    if row['BELONGS_GPI']:
        return("GPI")
    else:
        return("NOT GPI")
```

```
GENOMES_SAMPLE['GPI_LABEL']=GENOMES_SAMPLE.apply(assign_gpi_description,axis=1)
GENOMES_SAMPLE[:3]

VARIANTS_SAMPLE.reset_index(inplace=True)
VARIANTS_SAMPLE.set_index('UNIQUEID',inplace=True)
VARIANTS_SAMPLE=VARIANTS_SAMPLE.join(GENOMES_SAMPLE[['GPI_LABEL']],how='left')
VARIANTS_SAMPLE.GPI_LABEL.value_counts()
```

[68]:
```
NOT GPI    515924
GPI        161018
Name: GPI_LABEL, dtype: int64
```

Genetic variants are by definition all located on the reference H37rV genome (version 3) using `GENOME_INDEX` and therefore this *always* contains a value.

[69]:
```
VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.GENOME_INDEX.isna()]
```

[69]:
```
Empty DataFrame
Columns: [VARIANT, IS_SNP, REF, ALT, GENOME_INDEX, GENE, ELEMENT_TYPE,
MUTATION_TYPE, POSITION, NUCLEOTIDE_NUMBER, AMINO_ACID_NUMBER,
ASSOCIATED_WITH_GENE, IN_PROMOTER, IN_CDS, IS_INDEL, IS_HET, IS_NULL,
IS_FILTER_PASS, INDEL_LENGTH, INDEL_1, INDEL_2, DP, COVERAGE, DPF, FRS, GT_CONF,
GT_CONF_PERCENTILE, SITEID, GPI_LABEL]
Index: []
```

A genetic 'variant' is either (i) a single nucleotide polymorphism or (ii) an insertion or deletion of a specified number of nucleotides or (iii) a null (which could be either a SNP or an INDEL or nothing, we don't know!). These can be identified via `MUTATION_TYPE` and also (redundantly) using the Booleans `IS_SNP` and `IS_INDEL`.

[70]:
```
pandas.crosstab(VARIANTS_SAMPLE.MUTATION_TYPE,VARIANTS_SAMPLE.IS_SNP)
```

[70]:
```
IS_SNP         False    True
MUTATION_TYPE
INDEL          57814       0
NULL           17385       0
SNP                0  601743
```

[71]:
```
pandas.crosstab(VARIANTS_SAMPLE.MUTATION_TYPE,VARIANTS_SAMPLE.IS_INDEL)
```

[71]:
```
IS_INDEL        False  True
MUTATION_TYPE
INDEL               0  57814
NULL            17385      0
SNP            601743      0
```

[72]:
```
pandas.crosstab(VARIANTS_SAMPLE.MUTATION_TYPE,VARIANTS_SAMPLE.IS_NULL)
```

```
[72]: IS_NULL          False   True
      MUTATION_TYPE
      INDEL            57814      0
      NULL                 0  17385
      SNP             601743      0
```

SNPs simply have the nucleotide of the reference genome in `REF` and the observed allele in `ALT`.

Note that the allowed `REF` bases are `[a,t,c,g]` but the allowed `ALT` bases are `[a,t,c,g,o,x,z]` where `o` indicates a `vcf` filter fail, `x` indicates a Null call and `z` a Heterogenous call. As we shall see later, Clockwork is not, at present, making any Het calls and therefore there are no `z`s but the code allows for them. There are the associated `IS_FILTER_PASS`,`IS_NULL` and `IS_HET` Boolean fields to help with identifying such variants.

```
[73]: df=VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.IS_SNP]
      pandas.crosstab(df.REF,df.ALT)
```

```
[73]: ALT       a      c       g    o       t
      REF
      a         0  26303  101066  225    5557
      c     35527      0   38980  274  100215
      g     87828  48653       0  326   33239
      t      3385  91772   28228  165       0
```

SNPs are simply described as `GENOME_INDEX REF>ALT` in the `VARIANT` field e.g. `1849c>a` whilst indels are simply noted e.g. `1849_indel`.

`(UNIQUEID,VARIANT)` is therefore the (unique) primary key.

```
[74]: VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.IS_INDEL][:3]
```

```
[74]:                                               VARIANT  IS_SNP  REF ALT  \
      UNIQUEID
      site.00.subj.LE10KTB_23.lab.7627572.iso.1  26747_indel   False   gc   g
      site.00.subj.LE10KTB_23.lab.7627572.iso.1  34568_indel   False   tc   t
      site.00.subj.LE10KTB_23.lab.7627572.iso.1  49690_indel   False  gcc   g

                                                 GENOME_INDEX     GENE ELEMENT_TYPE  \
      UNIQUEID
      site.00.subj.LE10KTB_23.lab.7627572.iso.1         26747  Rv0021c        LOCUS
      site.00.subj.LE10KTB_23.lab.7627572.iso.1         34568    bioF2         GENE
      site.00.subj.LE10KTB_23.lab.7627572.iso.1         49690  Rv0045c        LOCUS

                                                 MUTATION_TYPE  POSITION  \
      UNIQUEID
      site.00.subj.LE10KTB_23.lab.7627572.iso.1          INDEL     135.0
      site.00.subj.LE10KTB_23.lab.7627572.iso.1          INDEL     274.0
      site.00.subj.LE10KTB_23.lab.7627572.iso.1          INDEL     250.0
```

```
                                          NUCLEOTIDE_NUMBER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1              135.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              274.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              250.0

                                          AMINO_ACID_NUMBER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1               45.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1               92.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1               84.0

                                          ASSOCIATED_WITH_GENE  IN_PROMOTER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False

                                          IN_CDS  IS_INDEL  IS_HET  IS_NULL  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1   True      True   False    False
site.00.subj.LE10KTB_23.lab.7627572.iso.1   True      True   False    False
site.00.subj.LE10KTB_23.lab.7627572.iso.1   True      True   False    False

                                          IS_FILTER_PASS  INDEL_LENGTH  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1           True          -1.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1           True          -1.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1           True          -2.0

                                            INDEL_1      INDEL_2    DP  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  26747_del  26747_del_1  26.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1  34568_del  34568_del_1  22.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1  49690_del  49690_del_2  19.0

                                          COVERAGE     DPF     FRS  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1        23  1.1685  0.9231
site.00.subj.LE10KTB_23.lab.7627572.iso.1        22  0.9887  1.0000
site.00.subj.LE10KTB_23.lab.7627572.iso.1        19  0.8539  1.0000

                                             GT_CONF  GT_CONF_PERCENTILE  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  153.789993           43.340000
site.00.subj.LE10KTB_23.lab.7627572.iso.1  178.750000           56.919998
site.00.subj.LE10KTB_23.lab.7627572.iso.1  159.929993           46.880001
```

```
                                           SITEID GPI_LABEL
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1     00   NOT GPI
site.00.subj.LE10KTB_23.lab.7627572.iso.1     00   NOT GPI
site.00.subj.LE10KTB_23.lab.7627572.iso.1     00   NOT GPI
```

Whilst INDELs have one of more nucleotides from the reference in `REF` and one or more nucleotides in `ALT` and the first position is assumed to be the start of the INDEL, which may not be true, but is the most straightforward assumption and otherwise you get tangled up in dividing variants into one or more constituents (since INDEL and SNP are not an orthogonal basis set).

The length of the INDEL is also recorded; note that this is the *net* length i.e. `len(ALT)-len(REF)`

[75]:
```
VARIANTS_SAMPLE.INDEL_LENGTH.value_counts().sort_index()
```

[75]:
```
-29750.0    1
-12719.0    5
-7890.0     1
-6967.0     1
-6807.0     1
           ..
 6166.0     1
 8335.0     1
 11044.0    1
 13044.0    1
 14054.0    1
Name: INDEL_LENGTH, Length: 443, dtype: int64
```

Since the nomenclature for an INDEL forms a nature hierarchy and we've used the simplest descriptor in the `VARIANT` field, the descending levels are included in `INDEL_1` and `INDEL_2`.

[76]:
```
VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.
 ↪IS_INDEL][['VARIANT','INDEL_1','INDEL_2','INDEL_LENGTH']][:3]
```

[76]:
```
                                               VARIANT    INDEL_1  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  26747_indel  26747_del
site.00.subj.LE10KTB_23.lab.7627572.iso.1  34568_indel  34568_del
site.00.subj.LE10KTB_23.lab.7627572.iso.1  49690_indel  49690_del

                                               INDEL_2  INDEL_LENGTH
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  26747_del_1          -1.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1  34568_del_1          -1.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1  49690_del_2          -2.0
```

Each successive descriptor gives a little more information; first about whether it is an insertion or a deletion, and then how many bases are involved.

At present, there are no examples (beyond frameshifts) where different 'flavours' of INDELs at the same `GENOME_POSITION` need to be distinguished since they have been associated with different effects on drug resistance. This is likely to change (or at least be tested), hence the flexibility is built in here.

All SNP and INDEL variants may (or may not) fall within a coding region (gene) or its promoter. This can be identified by the Boolean `ASSOCIATED_WITH_GENE`. If `True`, then `ELEMENT_TYPE` can be used to distinguish the 'type' of coding region. Note that being in the 'promoter' is ill-defined and here is assumed to be 100 bases upstream of the start codon (or up to the next coding region, whichever comes sooner).

[77]:
```
VARIANTS_SAMPLE.ASSOCIATED_WITH_GENE.value_counts()
```

[77]:
```
True     620750
False     56192
Name: ASSOCIATED_WITH_GENE, dtype: int64
```

[78]:
```
pandas.crosstab(VARIANTS_SAMPLE.ELEMENT_TYPE,VARIANTS_SAMPLE.
 ↪ASSOCIATED_WITH_GENE)
```

[78]:
```
ASSOCIATED_WITH_GENE  False   True
ELEMENT_TYPE
                      56192       0
GENE                      0  339352
LOCUS                     0  279085
RNA                       0    2313
```

`GENE`, `LOCUS` and `RNA` are as defined in the H37rV Genbank file (`NC_000962.3.gbk`). The `RNA` genes found are, of course, ribosomal. Implicit, therefore, is the assumption that `GENE` and `LOCUS` code proteins, whilst `RNA` do not.

`GENE` encodes the name of the GENE or LOCUS as defined by the H37rV Genbank file. There are examples of *M. tuberculosis* genes that are referred to in the literature by one name, but are called something else in the GenBank file. Only using the latter makes any sense!

[79]:
```
VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.ELEMENT_TYPE=='RNA'].GENE.value_counts()
```

[79]:
```
rrl       1161
rrs       1149
rrf          3
zwf2         0
Rv2235       0
          …
fadD19       0
fadD18       0
fadD17       0
fadD15       0
             0
Name: GENE, Length: 3863, dtype: int64
```

If we only considering variants that are associated with a gene/locus, we now find that there are multiple ways of identifying the genetic position where the variant occurs; the position in the whole genome (`GENOME_INDEX`) but also the number of nucleotides since the start of the gene/locus (`NUCLEOTIDE_NUMBER` – this is negative by definition for promoters) and, if the gene/locus encodes protein, the amino acid number (`AMINO_ACID_NUMBER`).

Note that this also makes it clear why one must use the *M. tuberculosis* (i.e. reference) numbering, and why e.g. using *E. coli* numbering for *rpoB* is confusing and prevents matching to the GenBank reference.

Note also that if these cannot be defined for a variant (e.g. it isn't associated with a gene, or encodes RNA and therefore `AMINO_ACID_NUMBER` is nonsensical) then you'll find a `NaN`. It is a peculiarity of Pandas that only `floats` can hold `NaN`s, whilst `ints` cannot, and therefore these are all stored as `floats`.

- `NUCLEOTIDE_NUMBER` is simply the 1-based number of the base. A pecularity of genes is there is no `0`, so if there is a promoter, it will run `-3,-2,-1,1,2,3,4...`
- `AMINO_ACID_NUMBER` is the sequential number of the amino acid residue that the base belongs to/codes for. Hence it is only populated in the coding region of genes that code for protein (i.e. not RNA encoding genes like *rrs*). For the same sequence as above it will be, `NaN,NaN,NaN,1,1,1,2`

To faciliate joining between the `VARIANT` and `MUTATIONS` tables there is fourth aggregated location field called `POSITION`. If the variant occurs in the coding region of a gene that codes protein this is `AMINO_ACID_NUMBER`, otherwise it is simply `NUCLEOTIDE_NUMBER` and if the variant is not associated with a gene in any way, then it is a `NaN`.

[80]:
```
VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.
↪ASSOCIATED_WITH_GENE][["GENOME_INDEX","NUCLEOTIDE_NUMBER","AMINO_ACID_NUMBER",'POSITION']][
↪5]
```

[80]:
```
                                          GENOME_INDEX  NUCLEOTIDE_NUMBER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1         1977              -75.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1         3446              167.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1         4013              734.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1         7362               61.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1         7585              284.0

                                          AMINO_ACID_NUMBER  POSITION
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1               NaN     -75.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              56.0      56.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1             245.0     245.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              21.0      21.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              95.0      95.0
```

Two other Booleans are provided to help you; `IN_CDS` and `IN_PROMOTER` which do what you think they do!

Finally, each variant has stored the quality metrics used by Clockwork in deciding whether or not to make a call. I will describe these, but since I am not an expert at the meanings of these, please direct questions to Zam, Jeff or Martin.

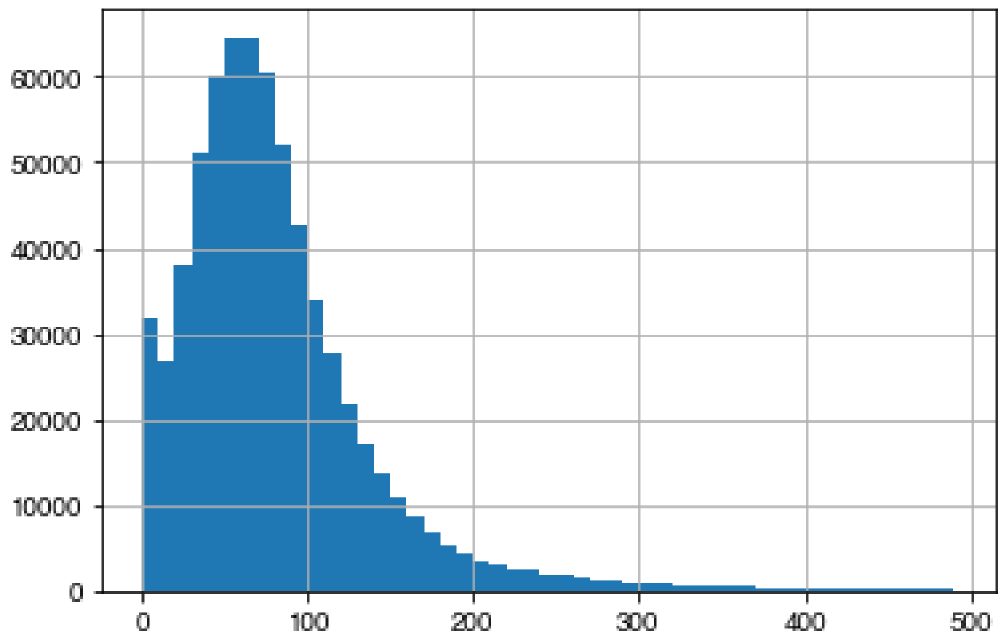
The total depth from the pile-up is stored in `DP` whilst the aggregate depth of the top two alleles is stored in `COVERAGE`. Most of the time these are identical, but on occasion `DP>COVERAGE` presumably because more than two bases were observed in pile-up.

[81]:
```
VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.DP>VARIANTS_SAMPLE.COVERAGE][:3]
```

[81]:
```
                                         VARIANT  IS_SNP REF ALT  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  69984c>a    True   c   a
site.00.subj.LE10KTB_23.lab.7627572.iso.1  69989g>a    True   g   a
site.00.subj.LE10KTB_23.lab.7627572.iso.1  72549g>a    True   g   a

                                          GENOME_INDEX    GENE ELEMENT_TYPE  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1         69984  Rv0064        LOCUS
site.00.subj.LE10KTB_23.lab.7627572.iso.1         69989  Rv0064        LOCUS
site.00.subj.LE10KTB_23.lab.7627572.iso.1         72549    icd2         GENE

                                         MUTATION_TYPE  POSITION  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1           SNP     455.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1           SNP     457.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1           SNP     655.0

                                          NUCLEOTIDE_NUMBER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1             1365.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1             1370.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1             1963.0

                                          AMINO_ACID_NUMBER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1              455.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              457.0
site.00.subj.LE10KTB_23.lab.7627572.iso.1              655.0

                                          ASSOCIATED_WITH_GENE  IN_PROMOTER  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False
site.00.subj.LE10KTB_23.lab.7627572.iso.1                 True        False

                                          IN_CDS  IS_INDEL  IS_HET  IS_NULL  \
UNIQUEID
```

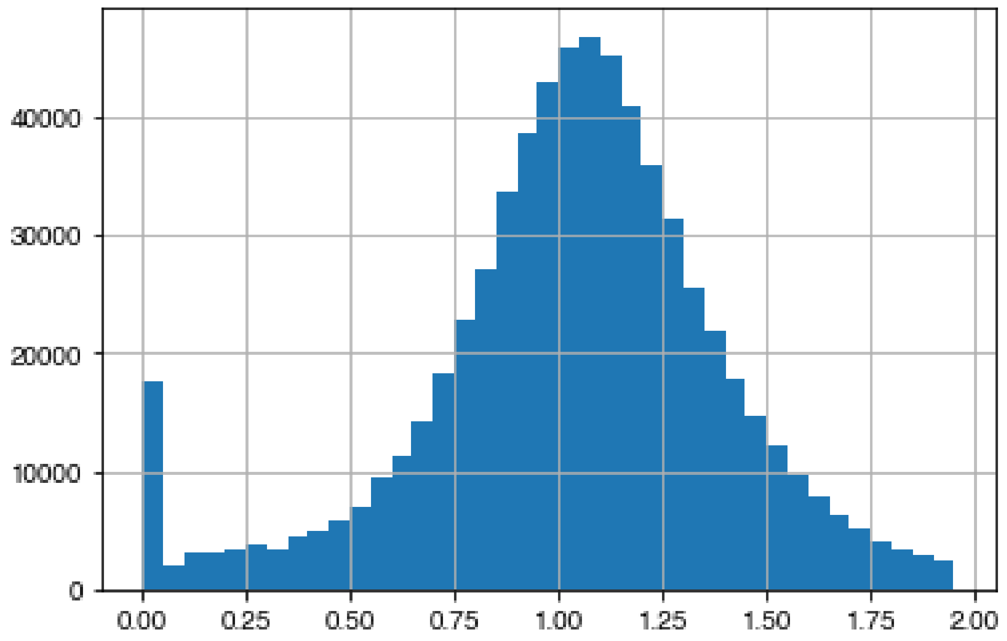
```
site.00.subj.LE10KTB_23.lab.7627572.iso.1    True     False    False    False
site.00.subj.LE10KTB_23.lab.7627572.iso.1    True     False    False    False
site.00.subj.LE10KTB_23.lab.7627572.iso.1    True     False    False    False

                                           IS_FILTER_PASS  INDEL_LENGTH  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1            True           NaN
site.00.subj.LE10KTB_23.lab.7627572.iso.1            True           NaN
site.00.subj.LE10KTB_23.lab.7627572.iso.1            True           NaN

                                          INDEL_1 INDEL_2    DP  COVERAGE  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1    None    None  32.0        30
site.00.subj.LE10KTB_23.lab.7627572.iso.1    None    None  32.0        30
site.00.subj.LE10KTB_23.lab.7627572.iso.1    None    None  25.0        24

                                              DPF   FRS     GT_CONF  \
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1  1.4381  1.00  220.899994
site.00.subj.LE10KTB_23.lab.7627572.iso.1  1.4381  1.00  220.899994
site.00.subj.LE10KTB_23.lab.7627572.iso.1  1.1235  0.96  181.759995

                                           GT_CONF_PERCENTILE SITEID GPI_LABEL
UNIQUEID
site.00.subj.LE10KTB_23.lab.7627572.iso.1           76.120003     00   NOT GPI
site.00.subj.LE10KTB_23.lab.7627572.iso.1           76.120003     00   NOT GPI
site.00.subj.LE10KTB_23.lab.7627572.iso.1           58.619999     00   NOT GPI
```

[82]:
```
a=VARIANTS_SAMPLE['DP'].hist(bins=numpy.arange(0,500,10),figsize=(6,4))
```



The depth at that position as a fraction of the average depth over the whole genome is stored in DPF. This is therefore a float. As you'd expect this is centred on unity, but there is also a peak centred on zero that are all null calls.

```
[83]: a=VARIANTS_SAMPLE['DPF'].hist(bins=numpy.arange(0,2,0.05),figsize=(6,4))
```

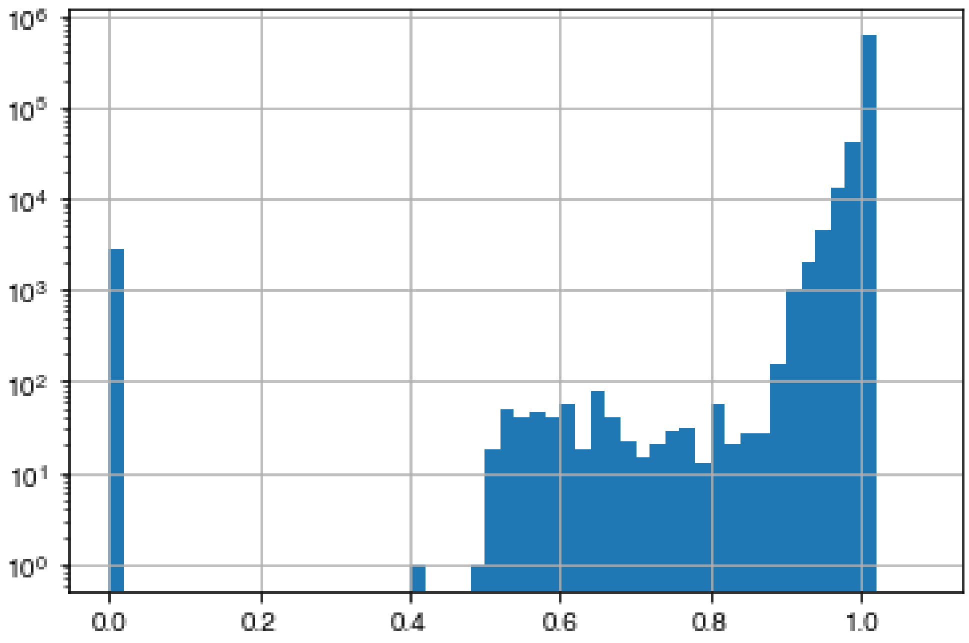


```
[84]: a=VARIANTS_SAMPLE.loc[VARIANTS_SAMPLE.DPF<0.01]
      pandas.crosstab(a.IS_NULL,a.IS_FILTER_PASS)
```

```
[84]: IS_FILTER_PASS  False  True
      IS_NULL
      False              23    413
      True             1118  13533
```

Then we have the Fraction of Read Support (`FRS`) which by definition has an upper bound of unity. This is the closest Clockwork gets to thinking about het calls at present.

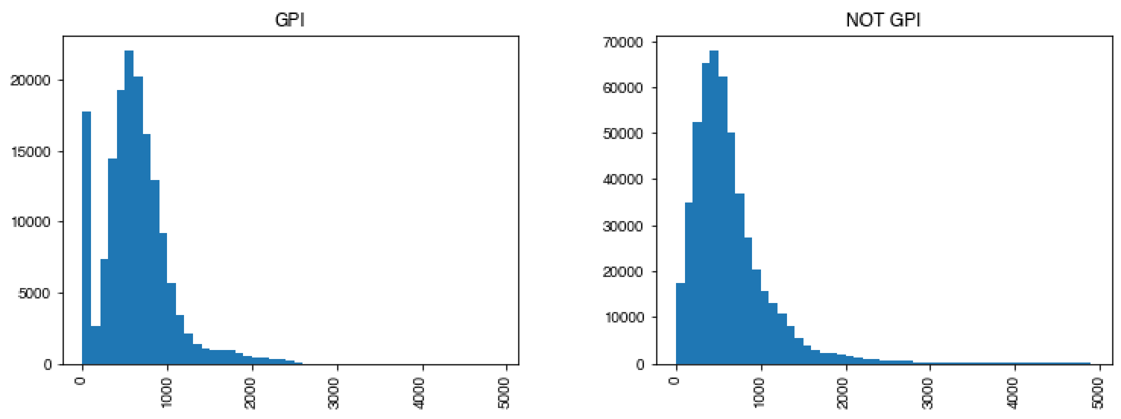
```
[85]: # a=VARIANTS_SAMPLE.FRS.hist(bins=numpy.arange(0,1.1,0.02))
      a=VARIANTS_SAMPLE['FRS'].hist(bins=numpy.arange(0.0,1.1,0.
       ↪02),figsize=(6,4),log=True)
```



Note that everything with `FRS<0.9` is either a NULL or FILTER_FAIL call.

Internally, Clockwork uses a model to predict the confidence of the call; this is stored in `GT_CONF`. Because the `GPI` samples are called together (by definition!) we have to start treating the GPI/NOT GPI calls separately from now on and as the quantities and thresholds are NOT equivalent.

```
[87]: a=VARIANTS_SAMPLE.GT_CONF.hist(by=VARIANTS_SAMPLE.
       ↪GPI_LABEL,bins=range(0,5000,100),figsize=(12,4))
```

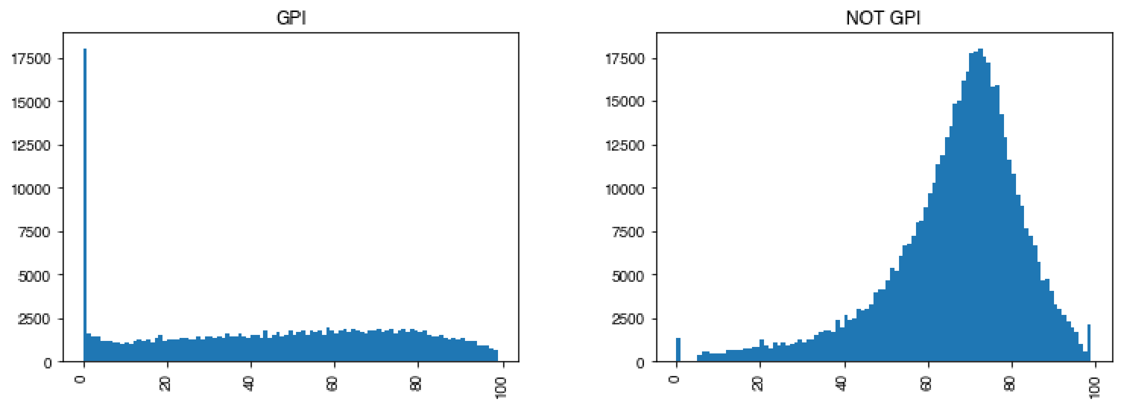


Since the threshold one might use to discard low confidence values is itself a function of depth, Clockwork creates a set of reads with the same average depth as the sample and then analyses these to create a `GT_CONF` distribution. It then uses this to convert the `GT_CONF` values for the actual sample into percentiles, which are stored as `GT_CONF_PERCENTILE` and the threshold is set as a percentile value.

Unfortunately `GT_CONF_PERCENTILE` cannot be compared between regenotyped and per-sample `vcf` files. For the latter a threshold of 5% was applied and everything below that was recorded as a filter fail. For the former a smaller threshold was applied by considering the precision/recall of the 17 high-quality `QC` Comas samples.

Hence `GT_CONF_PERCENTILE` is the preferred metric since it accounts for the depth in each sample and runs 0-100.

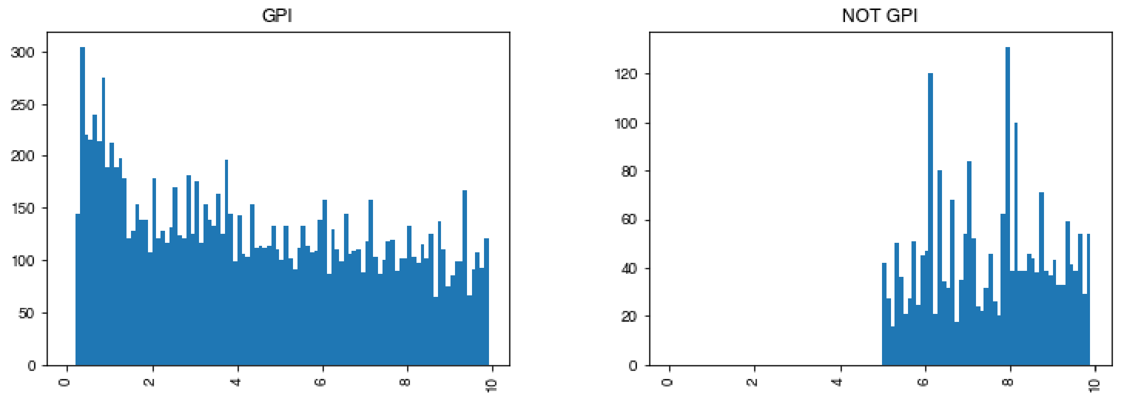
[88]:
```
a=VARIANTS_SAMPLE.GT_CONF_PERCENTILE.hist(by=VARIANTS_SAMPLE.
↪GPI_LABEL,bins=range(0,100,1),figsize=(12,4))
```



There are more nulls in the `GPI` simply because during the regenotyping each sample is examined at any and every position where there is evidence of a call in any sample, hence in many samples either a reference (wildtype) or null call is returned depending on the pile-up.

The GT_CONF_PERCENTILE thresholds for definite calls are 0.25% for the GPI and 5.0% for the non-GPI.

[89]:
```
df=VARIANTS_SAMPLE.loc[(VARIANTS_SAMPLE.IS_FILTER_PASS) & (~VARIANTS_SAMPLE.
↪IS_NULL)]
a=df.GT_CONF_PERCENTILE.hist(by=df.GPI_LABEL,bins=numpy.arange(0,10,0.
↪1),figsize=(12,4))
```



4.3 MUTATIONS

Although not as large as VARIANTS, the MUTATIONS table is still large at 82,290,716 rows (Aug 2020) and takes minutes to load on my workstation with 48 GB of memory. The compressed PKL and CSV versions take up 736 MB and 1.33 GB, respectively on disc. Again we STRONGLY recommend you work with the PKL version if you can to avoid memory issues. Like VARIANTS the GPI subset is stored. This is small enough when compressed (15,977,2222 rows: 124 MB / 275 MB) that it is stored in cryptic-tables/.

Finally, the rows relating to the 1% sample defined above in GENOMES_SAMPLE are also stored for testing and demonstration in MUTATIONS_SAMPLE.

[90]:
```
MUTATIONS=pandas.read_pickle(TABLES_PATH+"MUTATIONS_SAMPLE.pkl.gz")
MUTATIONS[:3]
```

[90]:
```
                                                   POSITION  \
UNIQUEID                           GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R       31.0
                                   PE1     L485L        485.0
                                   PE3     T14A          14.0

                                                   AMINO_ACID_NUMBER  \
UNIQUEID                           GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                31.0
                                   PE1     L485L               485.0
```

```
                                       PE3     T14A                   14.0

                                                           GENOME_INDEX  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                 NaN
                                       PE1     L485L                NaN
                                       PE3     T14A                 NaN

                                                           NUCLEOTIDE_NUMBER  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                      NaN
                                       PE1     L485L                     NaN
                                       PE3     T14A                      NaN

                                                           REF  ALT  IS_SNP  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R        caa  cga    True
                                       PE1     L485L       ctg  ttg    True
                                       PE3     T14A        acg  gcg    True

                                                           IS_INDEL  IN_CDS  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R           False    True
                                       PE1     L485L          False    True
                                       PE3     T14A           False    True

                                                           IN_PROMOTER  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R              False
                                       PE1     L485L             False
                                       PE3     T14A              False

                                                           IS_SYNONYMOUS  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                False
                                       PE1     L485L                True
                                       PE3     T14A               False

                                                           IS_NONSYNONYMOUS  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                    True
                                       PE1     L485L                  False
                                       PE3     T14A                    True

                                                           IS_HET  IS_NULL  \
UNIQUEID                               GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R         False    False
```

```
                                      PE1     L485L      False    False
                                      PE3     T14A       False    False

                                                         IS_FILTER_PASS  \
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                 True
                                      PE1     L485L                True
                                      PE3     T14A                 True

                                                        ELEMENT_TYPE  \
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R              GENE
                                      PE1     L485L             GENE
                                      PE3     T14A              GENE

                                                        MUTATION_TYPE  \
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                AAM
                                      PE1     L485L               AAM
                                      PE3     T14A                AAM

                                                         INDEL_LENGTH INDEL_1  \
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                NaN
                                      PE1     L485L               NaN
                                      PE3     T14A                NaN

                                                        INDEL_2 SITEID  \
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R                  02
                                      PE1     L485L                 02
                                      PE3     T14A                  02

NUMBER_NUCLEOTIDE_CHANGES
UNIQUEID                              GENE    MUTATION
site.02.subj.0345.lab.234051-15.iso.1 35kd_ag Q31R
1
                                      PE1     L485L

1
                                      PE3     T14A

1
```

The primary key (i.e. unique) is `UNIQUEID`,`GENE`,`MUTATION`. This protein level view (i.e. amino acids) needs to be separate from `VARIANTS` since you can have up to three SNPs in a single codon (and therefore three rows in `VARIANTS`) which would be represented by a single row here in `MUTATIONS`. This does mean, however, that if you want to find out the (min or max) `COVERAGE` in a codon, you need to join back to `VARIANTS`. This makes clear that the quality information only makes sense at the nucleotide level, not the codon level.

For more information on the grammar used to describe each mutation, head here.

Let's look at the mutations in the *rpoB* RRDR and look for any mutations where more than one base in the codon are different compared to the reference.

```
[91]: MUTATIONS.reset_index(inplace=True)
      df=MUTATIONS.loc[(MUTATIONS.GENE=='rpoB') & (MUTATIONS.POSITION>=428) &␣
       ↪(MUTATIONS.POSITION<=452) & (~MUTATIONS.IS_NULL)]
      pandas.crosstab(df.MUTATION,df.NUMBER_NUCLEOTIDE_CHANGES,margins=True)
```

```
[91]: NUMBER_NUCLEOTIDE_CHANGES    1  2  All
      MUTATION
      D435A                        1  0    1
      D435V                       11  0   11
      H445D                        9  0    9
      H445L                        1  0    1
      H445S                        0  2    2
      H445Y                        7  0    7
      L430P                        3  0    3
      L452P                        5  0    5
      Q429H                        1  0    1
      Q432P                        1  0    1
      S450F                        0  2    2
      S450L                       83  0   83
      S450Q                        4  0    4
      S450W                        1  0    1
      All                        127  4  131
```

All the mutations in our sample are SNPs in *rpoB* with 83 out of 131 being S450L, as expected and only four amino acid mutations involve more than 1 nucleotide change in the codon compared to reference.

Most of the remaining fields are there to help you select the mutations you want. These include a series of Booleans that are hopefully obvious: `IS_SNP`, `IS_INDEL`, `IN_CDS`, `IN_PROMOTER`, `IS_SYNONYMOUS`, `IS_NONSYNOYMOUS`, `IS_HET`, `IS_NULL`. And, yes, many of these are redundant e.g. `IN_CDS=~IN_PROMOTER` but it just makes life a bit easier.

`ELEMENT_TYPE` is the same as in `VARIANTS`, as are the additional descriptors for INDELS: `INDEL_LENGTH`, `INDEL_1` and `INDEL_2`.

`MUTATION_TYPE` is new and distinguishes between an amino acid mutation (`AAM`) e.g. a codon change which may or may not be synoymous and a nucleotide `SNP` e.g. in a promoter or an RNA gene, as well as `INDEL`s, which can be anywhere.

```
[92]: MUTATIONS.MUTATION_TYPE.value_counts()
```

```
[92]: AAM      526701
      INDEL     45175
```

```
SNP       38361
Name: MUTATION_TYPE, dtype: int64
```

Note that since FILTER_FAIL and NULL calls are included for genes in the resistance catalogue these need to be distinguished. This is done via the 'artificial' amino acids `O` and `X`, respectively. At present a NULL (i.e. `x`) in a codon trumps a FILTER_FAIL (i.e. `o`) . Hence an `ALT` of `aox` is translated to a `X` i.e. a NULL. There are only 8 instances in a our sample, however.

[94]:
```
MUTATIONS.loc[(MUTATIONS.ALT.str.contains('x')) & (MUTATIONS.ALT.str.
↪contains('o'))]
```

[94]:
```
                                       UNIQUEID   GENE MUTATION  POSITION  \
16247   site.05.subj.CA-1366.lab.CO-02570-19.iso.1   pepQ     A90X      90.0
79056          site.04.subj.01017.lab.720678.iso.1   embA    G149X     149.0
79432          site.04.subj.01017.lab.720678.iso.1   ethR    W145X     145.0
80045          site.04.subj.01017.lab.720678.iso.1  mmpL3    M211X     211.0
80085          site.04.subj.01017.lab.720678.iso.1  mmpL3    P867X     867.0
80199          site.04.subj.01017.lab.720678.iso.1   mshA    L277X     277.0
547769       site.03.subj.DR-8.lab.IML-01060.iso.1   ethA    V191X     191.0

        AMINO_ACID_NUMBER  GENOME_INDEX  NUCLEOTIDE_NUMBER  REF  ALT  IS_SNP  \
16247                90.0           NaN                NaN  gcc  xoc    True
79056               149.0           NaN                NaN  ggc  oxx    True
79432               145.0           NaN                NaN  tgg  oxg    True
80045               211.0           NaN                NaN  atg  otx    True
80085               867.0           NaN                NaN  ccg  xog    True
80199               277.0           NaN                NaN  ctg  cox    True
547769              191.0           NaN                NaN  gtg  xto    True

        IS_INDEL  IN_CDS  IN_PROMOTER  IS_SYNONYMOUS  IS_NONSYNONYMOUS  \
16247      False    True        False          False              True
79056      False    True        False          False              True
79432      False    True        False          False              True
80045      False    True        False          False              True
80085      False    True        False          False              True
80199      False    True        False          False              True
547769     False    True        False          False              True

        IS_HET  IS_NULL  IS_FILTER_PASS ELEMENT_TYPE MUTATION_TYPE  \
16247    False     True            True         GENE           AAM
79056    False     True            True         GENE           AAM
79432    False     True            True         GENE           AAM
80045    False     True            True         GENE           AAM
80085    False     True            True         GENE           AAM
80199    False     True            True         GENE           AAM
547769   False     True            True         GENE           AAM
```

```
        INDEL_LENGTH INDEL_1 INDEL_2 SITEID  NUMBER_NUCLEOTIDE_CHANGES
16247            NaN                     05                          2
79056            NaN                     04                          3
79432            NaN                     04                          2
80045            NaN                     04                          2
80085            NaN                     04                          2
80199            NaN                     04                          2
547769           NaN                     03                          2
```

4.4 EFFECTS

Now that we have a comprehensive view of all the genetic variants and their associated protein amino acid changes, we can apply one or more genetic resistance catalogues.

We are currently applying the `CRyPTICv1.31` catalogue which is a merged catalogue comprising NEJM2018 for the first-line compounds and ERJ2017 for the rest. It also includes all the genes identified by the Seq&Treat project as being of potential interest. As mentioned above all of these genes therefore have nulls and filter fails recorded in `VARIANTS` and `MUTATIONS` although since they only have default rows in the catalogue they can only ever cause a `U` or `S` to be returned.

[95]:
```python
EFFECTS=pandas.read_pickle(TABLES_PATH+"EFFECTS.pkl.gz")
EFFECTS[:10]
```

[95]:
```
                                                        SITEID  \
UNIQUEID                            DRUG GENE    MUTATION CATALOGUE_NAME
CATALOGUE_VERSION CATALOGUE_GRAMMAR
site.02.subj.0958.lab.22A197.iso.1 PZA  PPE35    A391O    CRyPTIC        v1.31
GARC1                       02
                                                 L896S    CRyPTIC        v1.31
GARC1                       02
                                   DLM  Rv1816   D70G     CRyPTIC        v1.31
GARC1                       02
                                   BDQ  Rv1816   D70G     CRyPTIC        v1.31
GARC1                       02
                                   CFZ  Rv1816   D70G     CRyPTIC        v1.31
GARC1                       02
                                   PAS  Rv1816   D70G     CRyPTIC        v1.31
GARC1                       02
                                   LZD  Rv1816   D70G     CRyPTIC        v1.31
GARC1                       02
                                   PZA  Rv3236c  T102A    CRyPTIC        v1.31
GARC1                       02
                                   AMI  aftB     D397G    CRyPTIC        v1.31
GARC1                       02
                                   KAN  aftB     D397G    CRyPTIC        v1.31
GARC1                       02
```

```
                                 PREDICTION
UNIQUEID                                  DRUG  GENE    MUTATION  CATALOGUE_NAME
CATALOGUE_VERSION CATALOGUE_GRAMMAR
site.02.subj.0958.lab.22A197.iso.1 PZA   PPE35   A391O     CRyPTIC        v1.31
GARC1                           U
                                                L896S     CRyPTIC        v1.31
GARC1                           U
                                   DLM   Rv1816  D70G      CRyPTIC        v1.31
GARC1                           U
                                   BDQ   Rv1816  D70G      CRyPTIC        v1.31
GARC1                           U
                                   CFZ   Rv1816  D70G      CRyPTIC        v1.31
GARC1                           U
                                   PAS   Rv1816  D70G      CRyPTIC        v1.31
GARC1                           U
                                   LZD   Rv1816  D70G      CRyPTIC        v1.31
GARC1                           U
                                   PZA   Rv3236c T102A     CRyPTIC        v1.31
GARC1                           U
                                   AMI   aftB    D397G     CRyPTIC        v1.31
GARC1                           U
                                   KAN   aftB    D397G     CRyPTIC        v1.31
GARC1                           U
```

`EFFECTS` contains one row per mutation in each catalogue gene per associated drug for a defined version of a single catalogue. Hence a single row in `MUTATIONS` e.g. `gyrA_A90V` may result in multiple rows in `EFFECTS` since not only can that mutation be associated with resistance to several fluroquinolones but also a range of different catalogues, perhaps also different versions of a single catalogue, may have been applied. In addition, there may be other `gyrA` mutations in the same sample, each of which will contribute one (or more) row to `EFFECTS`. Consider this sample which has 4 mutations in *gyrA*.

[96]:
```
EFFECTS.reset_index(inplace=True)
EFFECTS.loc[(EFFECTS.GENE=='gyrA') & (EFFECTS.UNIQUEID=="site.02.subj.0914.lab.
↪22A148.iso.1")]
```

[96]:
```
                                 UNIQUEID DRUG  GENE MUTATION CATALOGUE_NAME  \
11701  site.02.subj.0914.lab.22A148.iso.1  MXF  gyrA     E21Q        CRyPTIC
11702  site.02.subj.0914.lab.22A148.iso.1  OFX  gyrA     E21Q        CRyPTIC
11703  site.02.subj.0914.lab.22A148.iso.1  LEV  gyrA     E21Q        CRyPTIC
11704  site.02.subj.0914.lab.22A148.iso.1  MXF  gyrA     A90V        CRyPTIC
11705  site.02.subj.0914.lab.22A148.iso.1  OFX  gyrA     A90V        CRyPTIC
11706  site.02.subj.0914.lab.22A148.iso.1  LEV  gyrA     A90V        CRyPTIC
11707  site.02.subj.0914.lab.22A148.iso.1  MXF  gyrA     S95T        CRyPTIC
11708  site.02.subj.0914.lab.22A148.iso.1  OFX  gyrA     S95T        CRyPTIC
11709  site.02.subj.0914.lab.22A148.iso.1  LEV  gyrA     S95T        CRyPTIC
11710  site.02.subj.0914.lab.22A148.iso.1  MXF  gyrA    G668D        CRyPTIC
```

```
11711  site.02.subj.0914.lab.22A148.iso.1  OFX  gyrA    G668D         CRyPTIC
11712  site.02.subj.0914.lab.22A148.iso.1  LEV  gyrA    G668D         CRyPTIC

      CATALOGUE_VERSION CATALOGUE_GRAMMAR SITEID PREDICTION
11701             v1.31             GARC1     02          S
11702             v1.31             GARC1     02          U
11703             v1.31             GARC1     02          S
11704             v1.31             GARC1     02          R
11705             v1.31             GARC1     02          R
11706             v1.31             GARC1     02          R
11707             v1.31             GARC1     02          S
11708             v1.31             GARC1     02          S
11709             v1.31             GARC1     02          S
11710             v1.31             GARC1     02          S
11711             v1.31             GARC1     02          U
11712             v1.31             GARC1     02          S
```

The net result of all these predictions needs to sorted out; that is where the `PREDICTIONS` table comes in. This simply has one row per sample per drug per catalogue (version).

4.5 PREDICTIONS

[97]:
```
PREDICTIONS=pandas.read_pickle(TABLES_PATH+"PREDICTIONS.pkl.gz")
PREDICTIONS[:3]
```

[97]:
```
           SITEID  \
UNIQUEID                           DRUG CATALOGUE_NAME CATALOGUE_VERSION
CATALOGUE_GRAMMAR
site.02.subj.0958.lab.22A197.iso.1 CFZ  CRyPTIC        v1.31             GARC1
02
                                   RIF  CRyPTIC        v1.31             GARC1
02
                                   MXF  CRyPTIC        v1.31             GARC1
02

           PREDICTION  \
UNIQUEID                           DRUG CATALOGUE_NAME CATALOGUE_VERSION
CATALOGUE_GRAMMAR
site.02.subj.0958.lab.22A197.iso.1 CFZ  CRyPTIC        v1.31             GARC1
S
                                   RIF  CRyPTIC        v1.31             GARC1
R
                                   MXF  CRyPTIC        v1.31             GARC1
S

            DEFAULT_CATALOGUE
```

```
UNIQUEID                           DRUG CATALOGUE_NAME CATALOGUE_VERSION
CATALOGUE_GRAMMAR
site.02.subj.0958.lab.22A197.iso.1 CFZ  CRyPTIC        v1.31             GARC1
True
                                   RIF  CRyPTIC        v1.31             GARC1
True
                                   MXF  CRyPTIC        v1.31             GARC1
True
```

If we pull out the rows for the sample sample as above

[98]:
```
PREDICTIONS.reset_index(inplace=True)
PREDICTIONS.loc[(PREDICTIONS.UNIQUEID=="site.02.subj.0914.lab.22A148.iso.1") &␣
↪(PREDICTIONS.DRUG.isin(['LEV','MXF','OFX']))]
```

[98]:
```
                                UNIQUEID DRUG CATALOGUE_NAME  \
3956  site.02.subj.0914.lab.22A148.iso.1  LEV        CRyPTIC
3963  site.02.subj.0914.lab.22A148.iso.1  MXF        CRyPTIC
3966  site.02.subj.0914.lab.22A148.iso.1  OFX        CRyPTIC

     CATALOGUE_VERSION CATALOGUE_GRAMMAR SITEID PREDICTION  DEFAULT_CATALOGUE
3956             v1.31             GARC1     02          R               True
3963             v1.31             GARC1     02          R               True
3966             v1.31             GARC1     02          R               True
```

So they are all predicted to be resistant to the fluoroquinolones.

The logic is what you expect; if there is >0 rows in `EFFECTS` for a drug that predict resistance, then the sample is predicted to `R` regardless of what the other rows predict. If there are no rows in `EFFECTS` that predict resistance but >0 rows with `PREDICTION=='U'`, then the sample is predicted `U` for that sample regardless of the other rows. If there are 0 rows, or >0 rows which are all predicted to be `S`, then the sample is predicted to be `S`.

4.6 GPI_SNP_DISTANCES arrays

These are not tables, but instead are (NxN) `numpy` arrays of SNP distances between each GPI sample and every other GPI sample. The array is therefore symmetric with a leading diagonal of zeros.

For convenience, the labels are stored in a separate `numpy` array.

If you are not familiar with `numpy` please get in touch; keeping them in this form should save you a lot of time and effort.

[99]:
```
labels=numpy.load('GPI_SNP_DISTANCES_LABELS.npy')
labels
```

[99]:
```
array(['site.QA.subj.N0004.lab.N0004.iso.1',
       'site.QA.subj.N0031.lab.N0031.iso.1',
       'site.QA.subj.N0052.lab.N0052.iso.1', …,
       'site.21.subj.004.lab.MR253325K.iso.1',
       'site.21.subj.005.lab.MR311341D.iso.1',
       'site.21.subj.006.lab.MR304907G.iso.1'], dtype='<U57')
```

Note that the samples with `SITEID=='QA'` are the 17 Comas samples. 15211+17=15228

```
[100]: len(labels)
```

```
[100]: 15228
```

```
[101]: distances=numpy.load('GPI_SNP_DISTANCES_VALUES.npy')
       distances
```

```
[101]: array([[   0, 1118, 1140, …, 1198, 1157,  407],
              [1118,    0,  829, …, 1138,  847, 1108],
              [1140,  829,    0, …, 1166,  583, 1125],
              …,
              [1198, 1138, 1166, …,    0, 1196, 1192],
              [1157,  847,  583, …, 1196,    0, 1157],
              [ 407, 1108, 1125, …, 1192, 1157,    0]], dtype=int16)
```

Selecting the distances from one sample is easy with numpy fancy indexing. For example, to find all the SNP distances to the first sample we simply first create a array of Booleans telling us which column has the name of the first sample (the first one, funnily enough).

```
[102]: labels=='site.QA.subj.N0004.lab.N0004.iso.1'
```

```
[102]: array([ True, False, False, …, False, False, False])
```

Then we index the distances using that Boolean array

```
[103]: d=distances[labels=='site.QA.subj.N0004.lab.N0004.iso.1']
       d
```

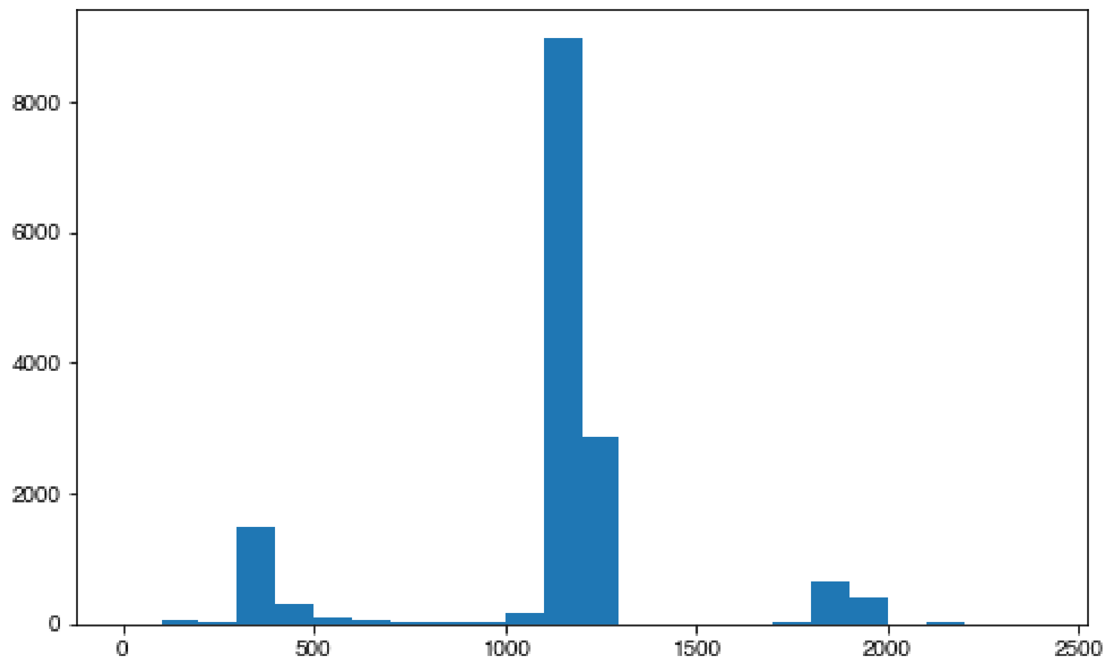
```
[103]: array([[   0, 1118, 1140, …, 1198, 1157,  407]], dtype=int16)
```

and we can simply calculate some statisitics and plot a histogram

```
[104]: numpy.average(d)
```

```
[104]: 1122.3733254531128
```

```
[105]: fig,axis=plt.subplots(1,1,figsize=(8,5))
       a=axis.hist(d.flatten(),bins=numpy.arange(0,2500,100))
```



Let's fish out all theGPI samples that are within 500 SNPs of this QA strain. This is where the power and simplicity of the `numpy` fancy indexing comes into its own!

The condition produces an array of Booleans..

```
[106]: close_samples=d<400
       close_samples
```

```
[106]: array([[ True, False, False, …, False, False, False]])
```

..which we can then use to pull out the distances

```
[107]: d[close_samples]
```

```
[107]: array([  0, 380, 346, …, 344, 349, 335], dtype=int16)
```

..and which samples they came from

```
[108]: labels[close_samples[0]]
```

```
[108]: array(['site.QA.subj.N0004.lab.N0004.iso.1',
              'site.QA.subj.N0054.lab.N0054.iso.1',
              'site.02.subj.0370.lab.222044-14.iso.1', …,
              'site.20.subj.SCH8604399.lab.YA00134971.iso.1',
              'site.21.subj.002.lab.MM011229E.iso.1',
              'site.21.subj.003.lab.MR428686L.iso.1'], dtype='<U57')
```

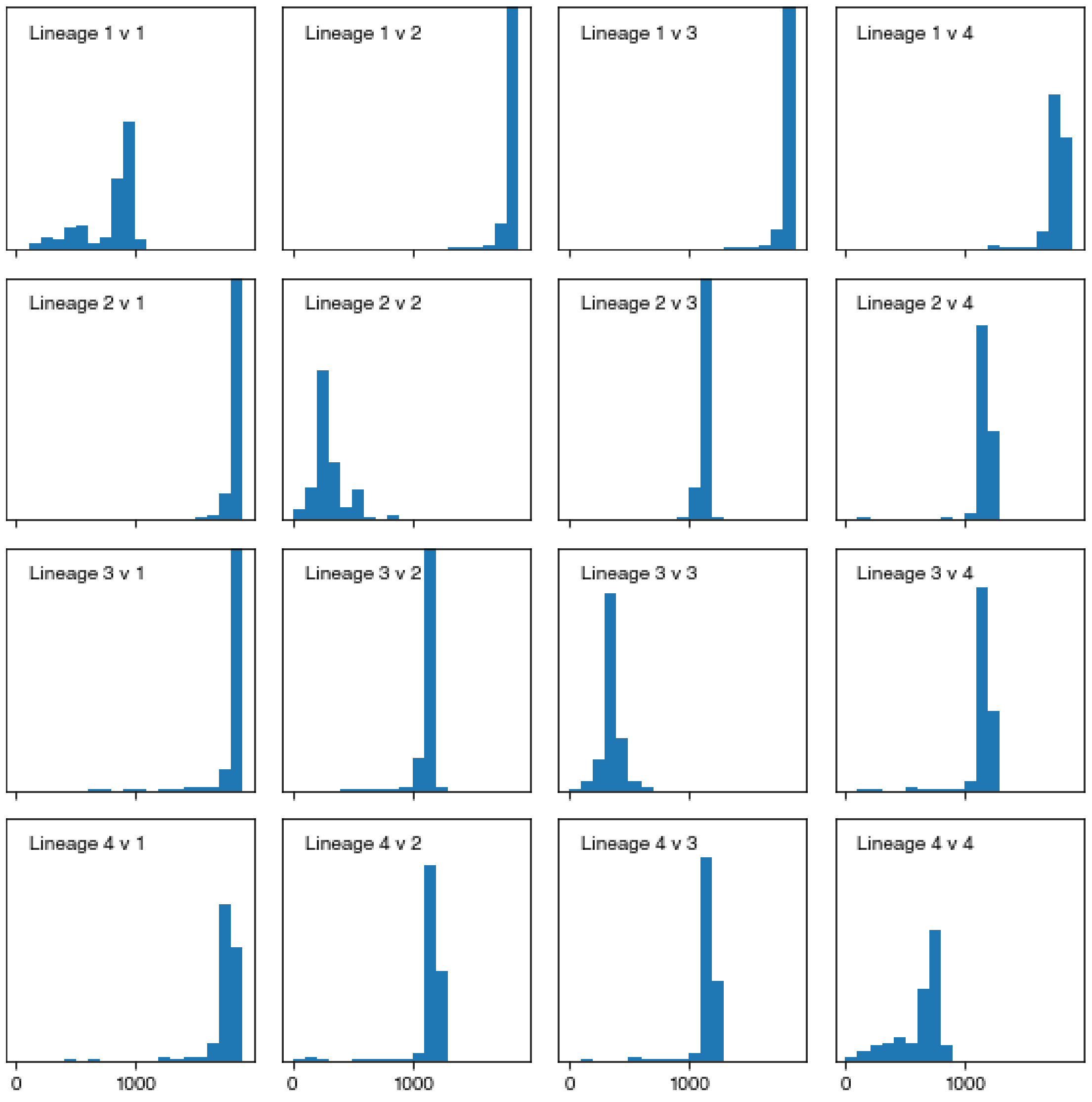
Finally, we can use this array to pull the rows in the GENOMES tables for these samples and then look which lineages they belong to

```
[109]: GENOMES.loc[GENOMES.index.isin(labels[close_samples[0]])].LINEAGE_NAME.
        ↪value_counts()
```

```
[109]: Lineage 3    1508
       Lineage 4      37
       Lineage 2       1
       Lineage 6       0
       Lineage 5       0
       Lineage 1       0
                       0
       Name: LINEAGE_NAME, dtype: int64
```

Let's try something more ambitious and calculate the SNP distributions for the distances between each and every set of lineages. (This will take about 30 sec to run)

```
[110]: l={}
       for lineage in [1,2,3,4]:
           l[lineage]=numpy.isin(labels,list(GENOMES.loc[GENOMES.
        ↪LINEAGE_NAME=='Lineage '+str(lineage)].index))

       d={}
       for lineage1 in [1,2,3,4]:
           for lineage2 in [1,2,3,4]:
               d[(lineage1,lineage2)]=distances[numpy.ix_(l[lineage1],l[lineage2])]

       fig,axes=plt.subplots(4,4,sharex=True,tight_layout=True,figsize=(8,8))

       for lineage1 in [1,2,3,4]:
           for lineage2 in [1,2,3,4]:
               axes[lineage1-1,lineage2-1].set_ylim([0,0.008])
               axes[lineage1-1,lineage2-1].axes.get_yaxis().set_visible(False)
               axes[lineage1-1,lineage2-1].hist(d[(lineage1,lineage2)].
        ↪flatten(),bins=numpy.arange(0,2000,100),density=True)
               axes[lineage1-1,lineage2-1].text(100,0.007,"Lineage "+str(lineage1)+" v␣
        ↪"+str(lineage2),horizontalalignment='left')
```



As expected, all samples that belong to a lineage are more similar to one another (generally < 1000 SNPs) than when samples belong to two different lineages are compared. The exception is when comparing Lineages 2 and 3 which are more alike than any of the other lineage pairs.

This result provides some comfort that (a) the SNP distance calculated by the Clockwork regenotyping process is correct and (b) the lineages called by SNP-IT are also good.

5 Analysis Examples

5.1 rpoB_S450L MIC distribution

[111]:
```
PHENOTYPES=pandas.read_pickle(TABLES_PATH+"UKMYC_PHENOTYPES.pkl.gz")

PHENOTYPES.reset_index(inplace=True)

PHENOTYPES=PHENOTYPES.loc[(PHENOTYPES.DRUG=="RIF") &\
                          (PHENOTYPES.PLATEDESIGN=="UKMYC5") &\
                          (PHENOTYPES.PHENOTYPE_QUALITY=="HIGH") &\
                          (PHENOTYPES.DILUTION>0)]

PHENOTYPES.set_index(["UNIQUEID"],inplace=True,verify_integrity=True)

PHENOTYPES[:3]
```

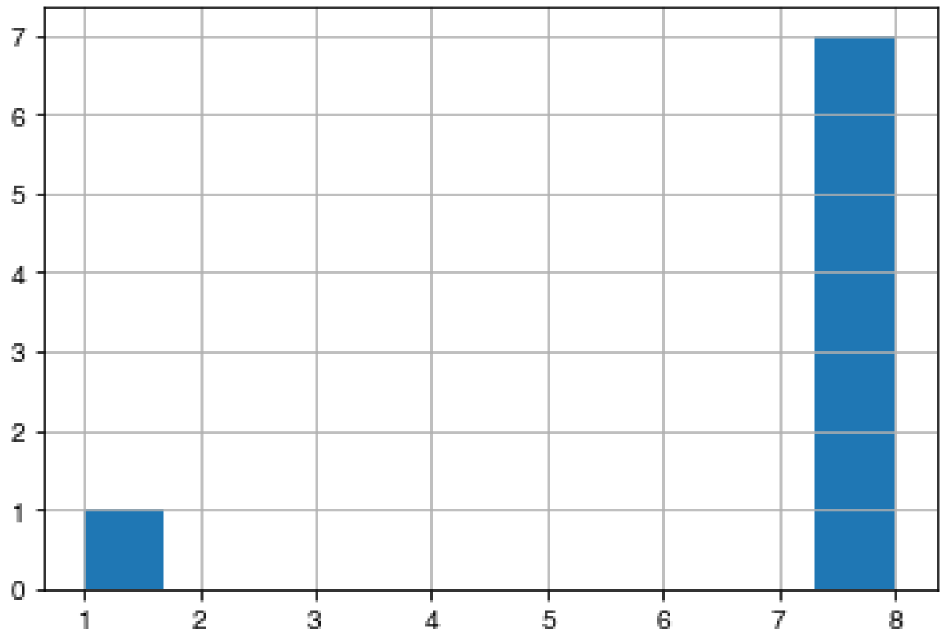
[111]:
```
                                             DRUG PLATEDESIGN  BELONGS_GPI  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1      RIF      UKMYC5         True
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1 RIF      UKMYC5         True
site.05.subj.PTAN-0394.lab.TAN-650.iso.1      RIF      UKMYC5         True

                                             SITEID  DILUTION  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1         05       8.0
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1     06       8.0
site.05.subj.PTAN-0394.lab.TAN-650.iso.1         05       2.0

                                             PHENOTYPE_QUALITY READINGDAY  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                  HIGH         14
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1              HIGH         14
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                  HIGH         14

                                              PRIMARY_DILUTION PRIMARY_METHOD  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                   8.0             VZ
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1               8.0             VZ
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                   2.0             VZ

                                              AMYGDA_DILUTION  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                  1.0
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1              8.0
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                  2.0
```

```
                                              BASHTHEBUG_DILUTION  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                      8.0
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1                  8.0
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                      2.0

                                              BASHTHEBUGPRO_DILUTION  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                         NaN
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1                     NaN
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                         NaN

                                             PHENOTYPE_DESCRIPTION  \
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1               VZ,BB AGREE
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1           VZ,IM AGREE
site.05.subj.PTAN-0394.lab.TAN-650.iso.1               VZ,IM AGREE

                                              BASHTHEBUG_NUMBER_CLASSIFICATIONS
\
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1                                   16.0
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1                               11.0
site.05.subj.PTAN-0394.lab.TAN-650.iso.1                                   11.0

                                               MIC  LOG2MIC BINARY_PHENOTYPE
UNIQUEID
site.05.subj.PSLM-0812.lab.SLM-072.iso.1        >4     3.00                R
site.06.subj.SSM_0145-14.lab.06MIL0268.iso.1    >4     3.00                R
site.05.subj.PTAN-0394.lab.TAN-650.iso.1      0.12    -3.06                S
```

[114]:
```
MUTATIONS=pandas.read_pickle(TABLES_PATH+"MUTATIONS_SAMPLE.pkl.gz")

MUTATIONS.reset_index(inplace=True)

MUTATIONS=MUTATIONS.loc[(MUTATIONS.MUTATION=="S450L") & (MUTATIONS.
 ↪GENE=="rpoB")]

MUTATIONS.set_index(["UNIQUEID"],inplace=True,verify_integrity=True)

MUTATIONS[:3]
```

[114]:
```
                                             GENE MUTATION  POSITION  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1   rpoB    S450L     450.0
site.05.subj.LI2076709.lab.15277_3_50.iso.1  rpoB    S450L     450.0
site.05.subj.PSLM-0779.lab.SLM-034.iso.1     rpoB    S450L     450.0

                                             AMINO_ACID_NUMBER  GENOME_INDEX  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1               450.0           NaN
site.05.subj.LI2076709.lab.15277_3_50.iso.1              450.0           NaN
site.05.subj.PSLM-0779.lab.SLM-034.iso.1                 450.0           NaN

                                             NUCLEOTIDE_NUMBER  REF  ALT  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1                 NaN  tcg  ttg
site.05.subj.LI2076709.lab.15277_3_50.iso.1                NaN  tcg  ttg
site.05.subj.PSLM-0779.lab.SLM-034.iso.1                   NaN  tcg  ttg

                                             IS_SNP  IS_INDEL  IN_CDS  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1     True     False    True
site.05.subj.LI2076709.lab.15277_3_50.iso.1    True     False    True
site.05.subj.PSLM-0779.lab.SLM-034.iso.1       True     False    True

                                             IN_PROMOTER  IS_SYNONYMOUS  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1         False          False
site.05.subj.LI2076709.lab.15277_3_50.iso.1        False          False
site.05.subj.PSLM-0779.lab.SLM-034.iso.1           False          False

                                             IS_NONSYNONYMOUS  IS_HET  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1               True   False
site.05.subj.LI2076709.lab.15277_3_50.iso.1              True   False
site.05.subj.PSLM-0779.lab.SLM-034.iso.1                 True   False

                                             IS_NULL  IS_FILTER_PASS  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1     False            True
site.05.subj.LI2076709.lab.15277_3_50.iso.1    False            True
site.05.subj.PSLM-0779.lab.SLM-034.iso.1       False            True

                                            ELEMENT_TYPE MUTATION_TYPE  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1          GENE           AAM
site.05.subj.LI2076709.lab.15277_3_50.iso.1         GENE           AAM
site.05.subj.PSLM-0779.lab.SLM-034.iso.1            GENE           AAM

                                             INDEL_LENGTH INDEL_1 INDEL_2  \
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1            NaN
site.05.subj.LI2076709.lab.15277_3_50.iso.1           NaN
site.05.subj.PSLM-0779.lab.SLM-034.iso.1              NaN

                                            SITEID  NUMBER_NUCLEOTIDE_CHANGES
UNIQUEID
site.05.subj.LR-2264.lab.FN-00887-17.iso.1      05                          1
site.05.subj.LI2076709.lab.15277_3_50.iso.1     05                          1
site.05.subj.PSLM-0779.lab.SLM-034.iso.1        05                          1
```

[115]:
```
MUTATIONS=MUTATIONS[["ALT"]]
df=PHENOTYPES.join(MUTATIONS,how="inner")
a=df.DILUTION.hist()
```



So as expected, the majority of samples with an `rpoB@S450L` mutation have growth in all wells on the UKMYC5 plate.