

```
In [1]: ➜ import pandas as pd
         from math import sqrt
         from os import listdir
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
```

```
In [2]: ➜ def load_zscores(battery: str, filename: str):
             return pd.read_csv("zscores/" + battery + "/" + filename, header=None)

def calc_norm(zscores: list[float]) -> float:
    total = 0
    for zscore in zscores:
        total += zscore ** 2
    return sqrt(total)

class Test:
    def __init__(self, battery: str, name: str):
        self.name = name
        self.scores = list(load_zscores(battery, name)[0].values.tolist())
        self.norm = calc_norm(self.scores)
        self.max = max(self.scores)
        self.min = min(self.scores)

    def get_norm(self):
        return self.norm

def load_battery(battery: str) -> list[Test]:
    tests = []
    for file in listdir("zscores/" + battery):
        tests.append(Test(battery, file))
    return tests

def sort_battery(tests: list[Test]):
    return sorted(tests, key=Test.get_norm)

def print_battery(tests: list[Test]):
    for test in tests:
        print(f"battery: {test.battery} name: {test.name} norm: {test.norm}")
```

```
In [3]: def graph_test(battery: str, filename: str, ax, i: int, height):
    zscores = load_zscores(battery, filename)
    zscores[1] = filename
    zscores.plot(kind="scatter",
                 y=1,
                 x=0,
                 color="Red",
                 rot=0,
                 figsize=(20,height),
                 ax=ax,
                 alpha=0.15,
                 grid=True)

def graph_test_fst(battery: str, filename: str, height):
    zscores = load_zscores(battery, filename)
    zscores[1] = filename

    return zscores.plot(kind="scatter",
                         y=1,
                         x=0,
                         color="Red",
                         rot=0,
                         figsize=(20,height),
                         alpha=0.15,
                         grid=True)

def get_battery_min_max(battery: list[Test]) -> tuple[float, float]:
    min_b, max_b = 0, 0
    for test in battery:
        if test.min < min_b:
            min_b = test.min
        if test.max > max_b:
            max_b = test.max
    return min_b, max_b

def graph_battery(battery: str, height, left, right):
    directory = "zscores/" + battery + "/"
    i = 0
    for filename in listdir(directory):
        if i == 0:
            ax = graph_test_fst(battery, filename, height)
        else:
            graph_test(battery, filename, ax, i, height)
        i += 1
    plt.xlim([left, right])
    plt.show()

def graph_battery_sns(battery: list[Test], height: int) -> None:
    combined = pd.DataFrame()
    for test in battery:
        zscores = pd.DataFrame(test.scores)
        zscores[1] = test.name + " " + str(test.norm)

        combined = pd.concat([combined, zscores], ignore_index=True)

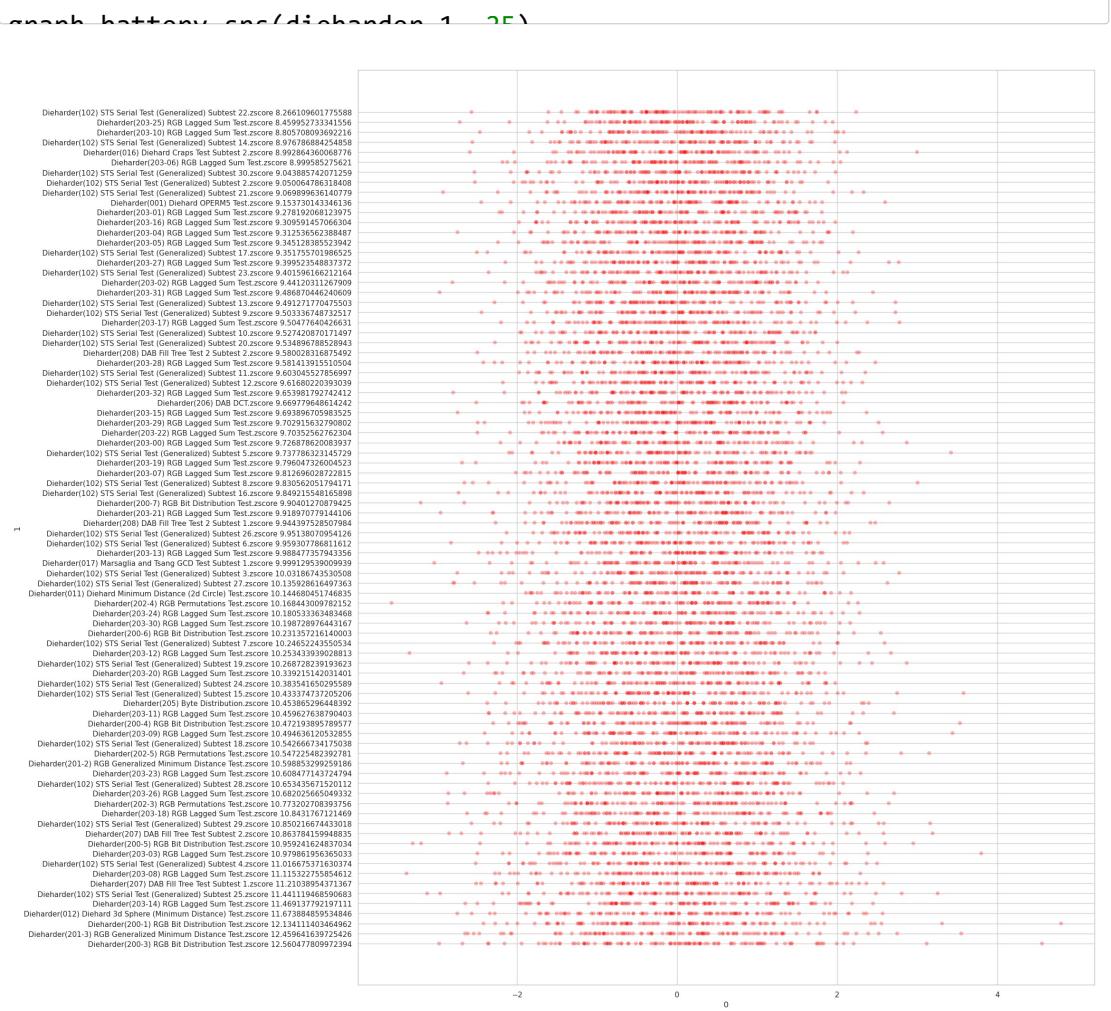
    sns.set(rc={"figure.figsize":(20, height)}, style="whitegrid") #width
```

```
p = sns.scatterplot(  
    data=combined,  
    x=0,  
    y=1,  
    hue=1,  
    palette={"red"},  
    legend=False,  
    alpha=0.3,  
)  
  
def split_battery(battery: list[Test]) -> tuple[list[Test]]:  
    low = []  
    high = []  
    for test in battery:  
        if test.norm < pivot:  
            low.append(test)  
        else:  
            high.append(test)  
    return low, high  
  
def slice_battery(battery: list[Test], low: float, high: float) -> list[Test]:  
    new = []  
    for test in battery:  
        if low <= test.norm <= high:  
            new.append(test)
```

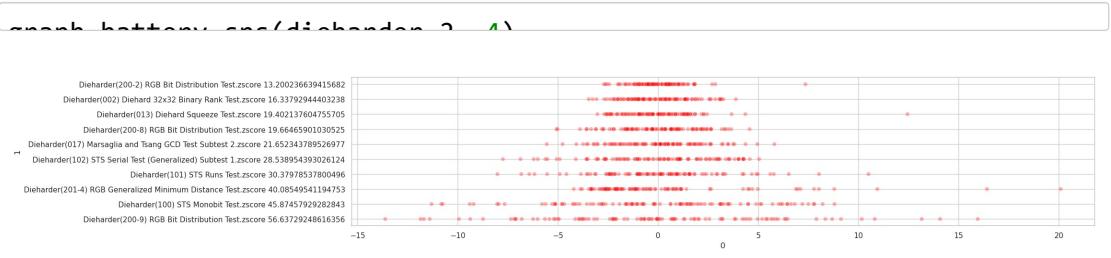
In [39]: dieharder = sort_battery(load_battery("dieharder"))

```
dieharder_1 = slice_battery(dieharder, 0, 13)  
dieharder_2 = slice_battery(dieharder, 13, 57)  
dieharder_3 = slice_battery(dieharder, 57, 100)  
dieharder_4 = slice_battery(dieharder, 100, 430)  
dieharder_5 = slice_battery(dieharder, 430, 1000)
```

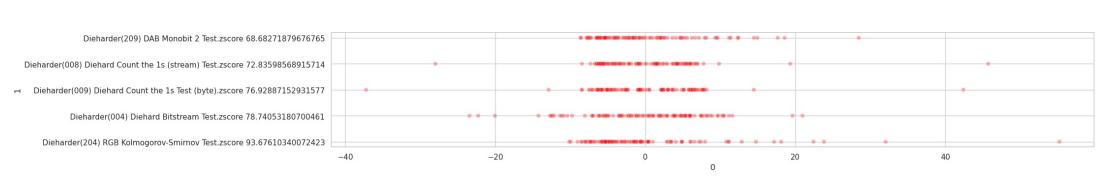
In [40]:



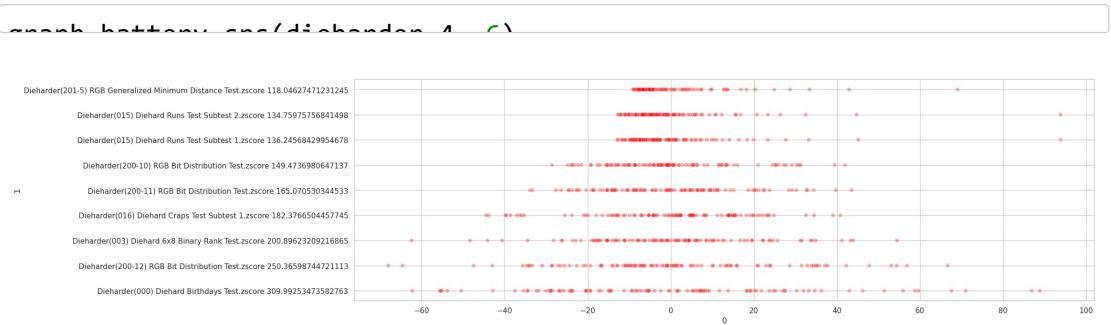
In [41]:



In [42]:



In [43]:



In [44]:



In [4]:

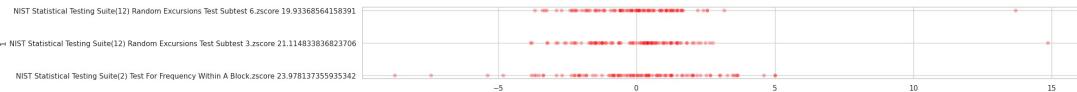
```
nist = sort_battery(load_battery("nist"))

nist_1 = slice_battery(nist, 0, 19.9)
nist_2 = slice_battery(nist, 19.9, 24)
nist_3 = slice_battery(nist, 24, 41)
nist_4 = slice_battery(nist, 41, 47)
nist_5 = slice_battery(nist, 47, 67 )
nist_6 = slice_battery(nist, 69, 170)
nist_7 = slice_battery(nist, 170, 1000)
```

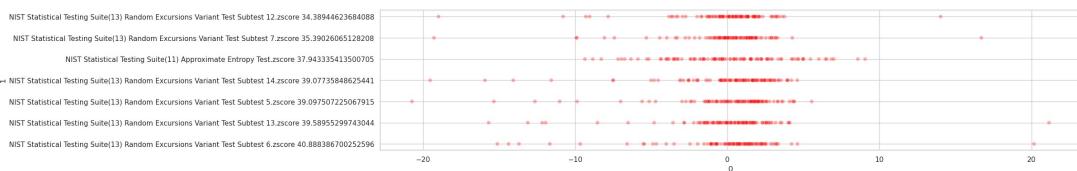
In [5]:



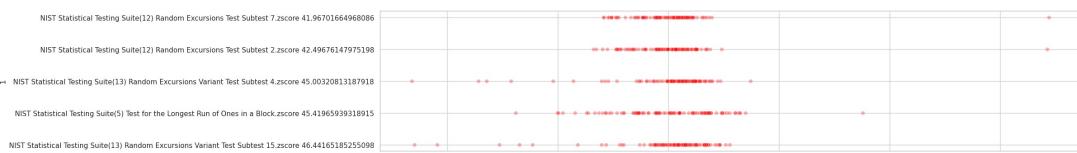
In [6]:



In [7]:

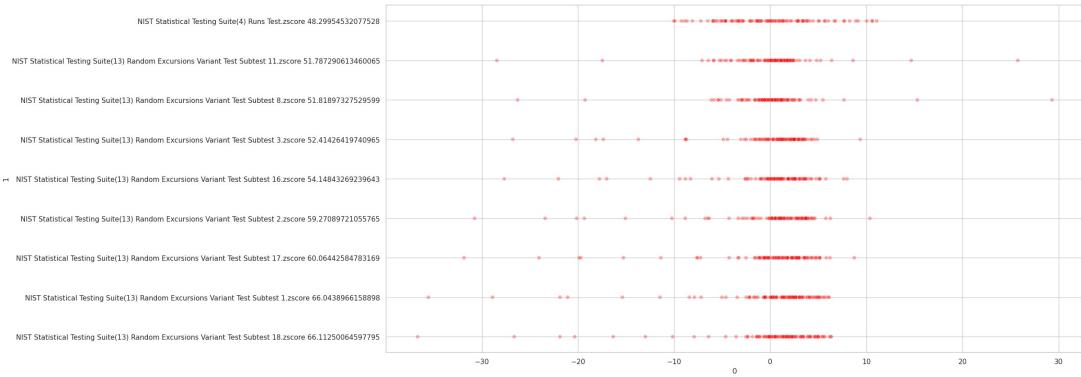


In [8]:



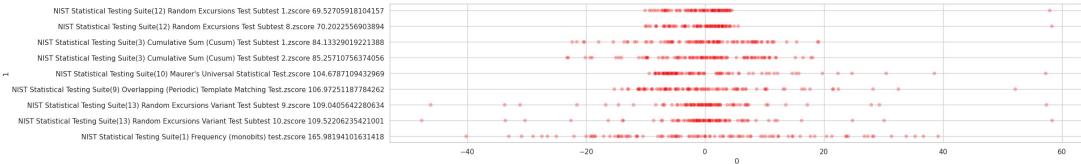
In [9]:

graph battery and nist 5-10



In [10]:

graph battery and nist 6-11



In [11]:

graph battery and nist 7-11



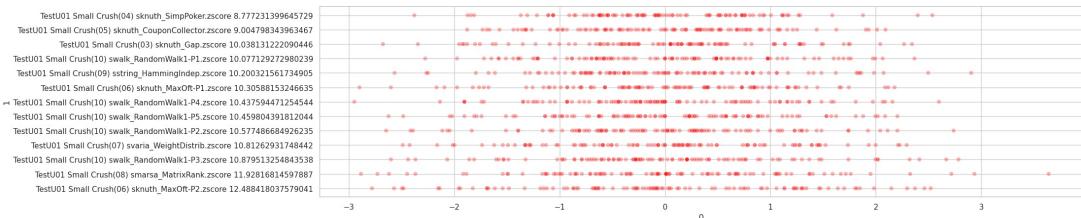
In [53]:

```
smallcrush = sort_battery(load_battery("testu01-smallcrush"))

smallcrush_1 = slice_battery(smallcrush, 0, 13)
smallcrush_2 = slice_battery(smallcrush, 13, 140)
smallcrush_3 = slice_battery(smallcrush, 140, 1000)
```

In [54]:

graph battery and smallcrush_1



In [55]:

graph battery and smallcrush_2



In [56]:

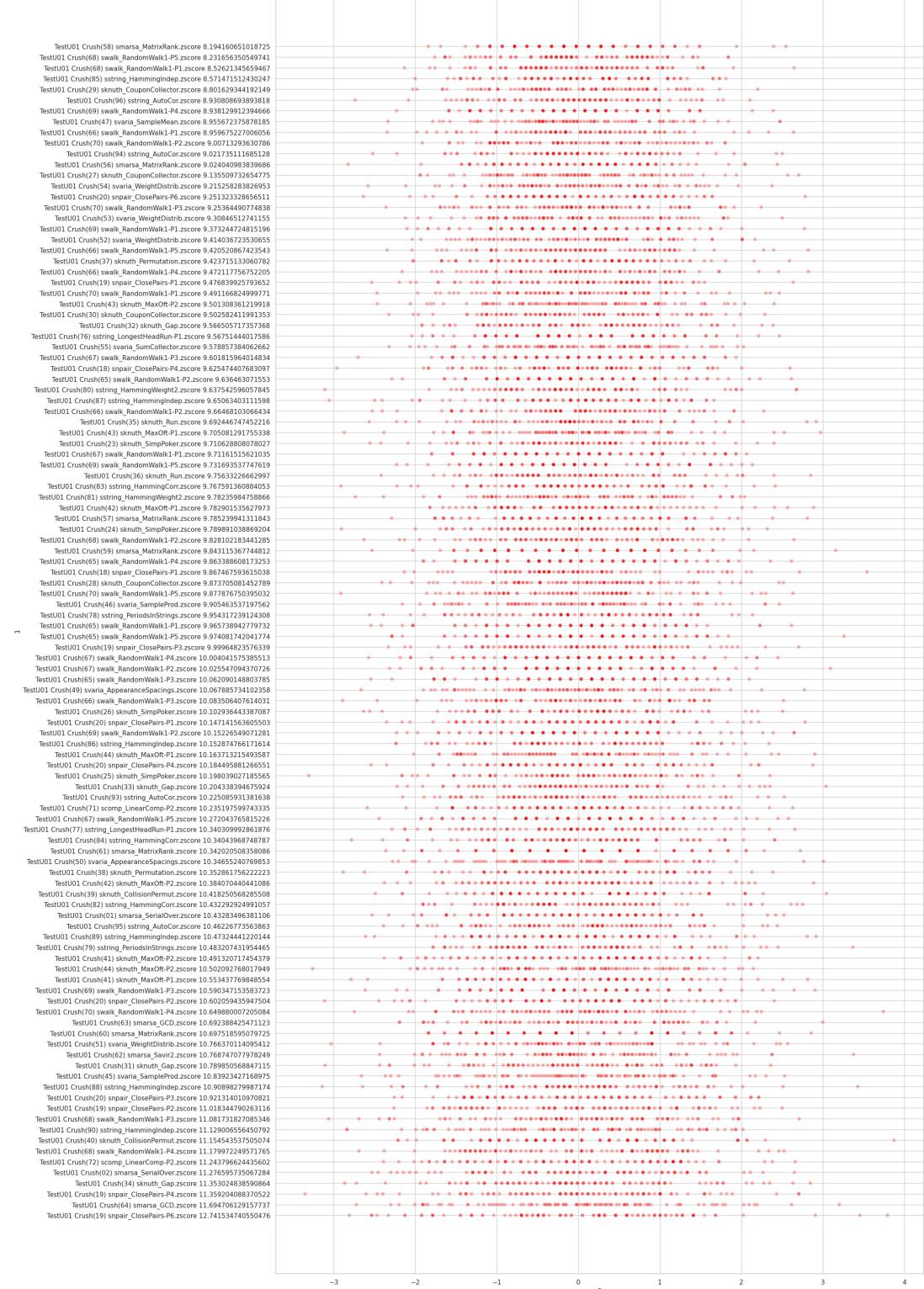
graph battery and smallcrush_3



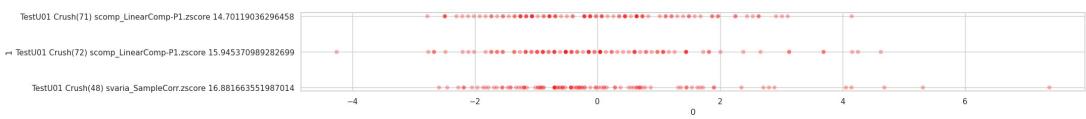
In [112]: crush = sort_battery(load_battery("testu01-crush"))

```
crush_1 = slice_battery(crush, 0, 14)
crush_2 = slice_battery(crush, 14, 17)
crush_3 = slice_battery(crush, 17, 50)
crush_4 = slice_battery(crush, 50, 100)
crush_5 = slice_battery(crush, 100, 160)
```

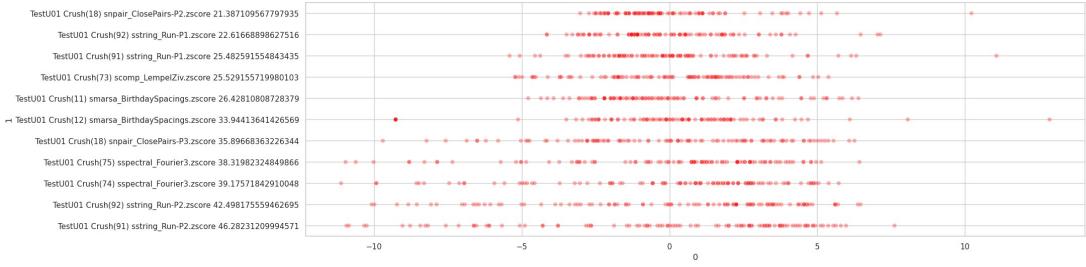
In [113]: crush_battery_and_crush_1[0]



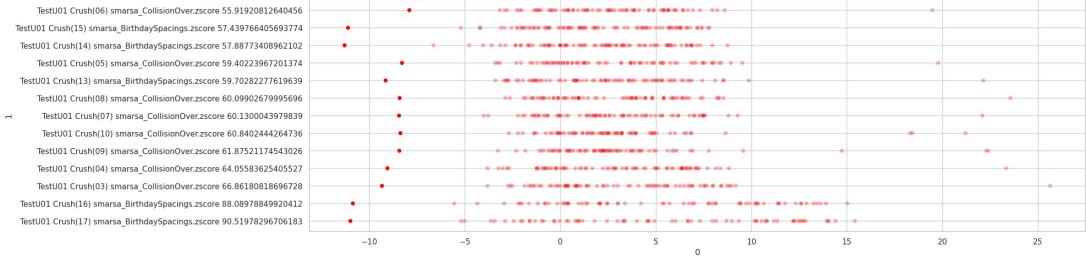
In [114]:



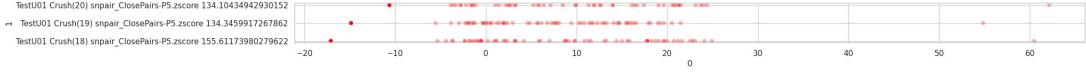
In [110]:



In [111]:



In [60]:



In [61]:

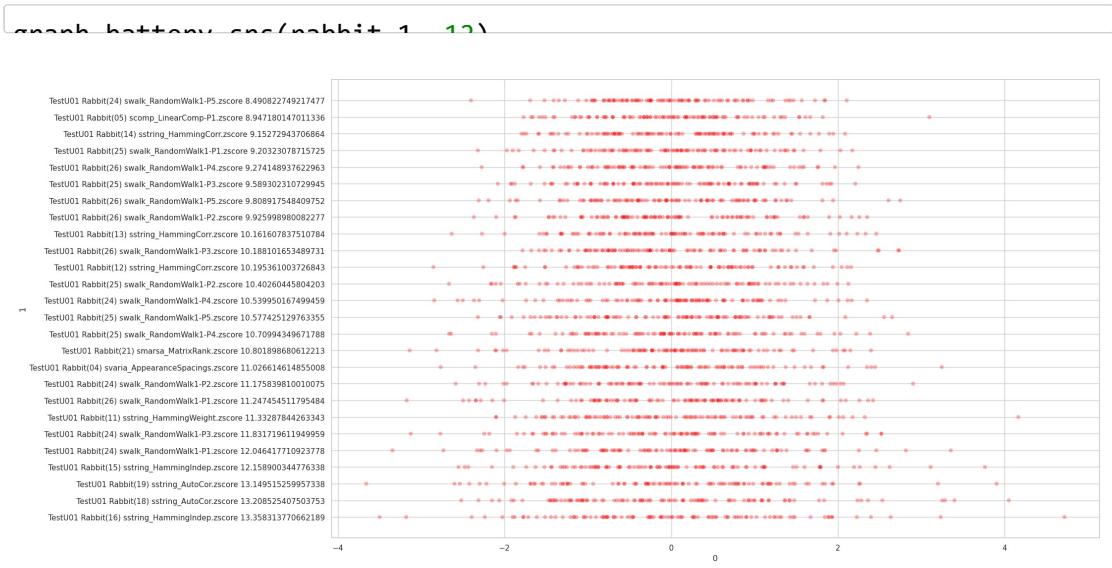


In [105]:

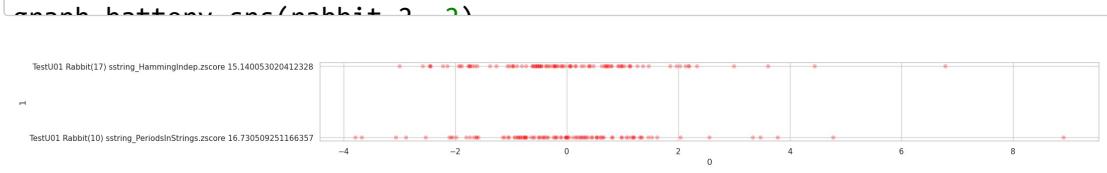
```
rabbit = sort_battery(load_battery("testu01-rabbit"))
```

```
rabbit_1 = slice_battery(rabbit, 0, 14)
rabbit_2 = slice_battery(rabbit, 14, 17)
rabbit_3 = slice_battery(rabbit, 17, 260)
rabbit_4 = slice_battery(rabbit, 260, 1000)
rabbit_5 = slice_battery(rabbit, 1000, 2000)
```

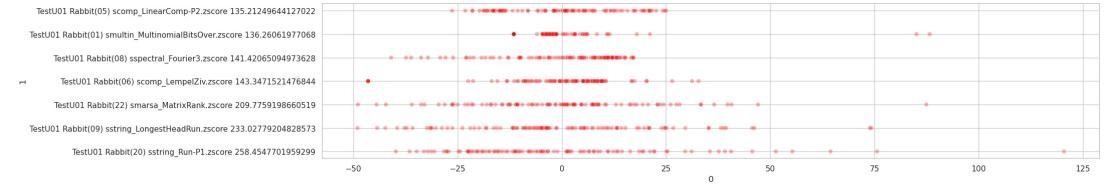
In [106]:



In [107]:



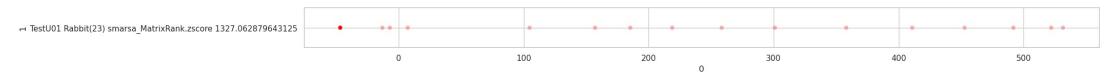
In [98]:



In [101]:



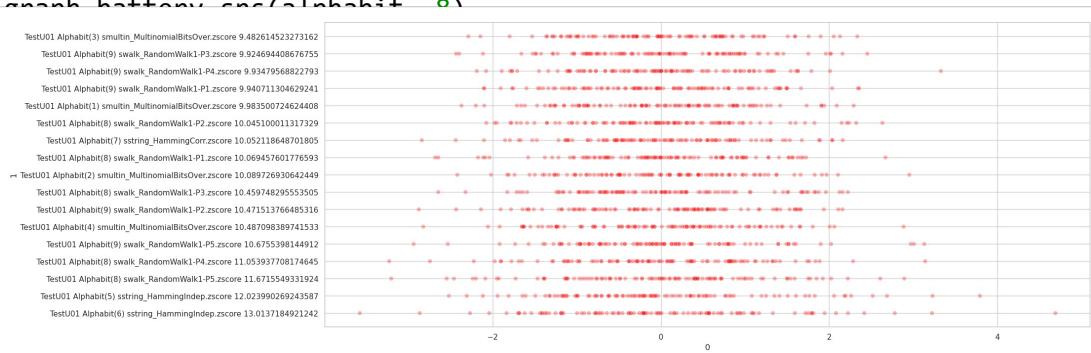
In [102]:



In [103]:



```
In [65]: ⏎ alphabit = sort_battery(load_battery("testU01-alphabit"))
```



In [66]: `blockalphabit = sort_battery(load_battery("testu01-blockalphabit"))`

```
TestU01 Block Alphabit(2-2) smutin_MultinomialBitsOver.zscore 8.581498529207954
TestU01 Block Alphabit(8-16) swalk_RandomWalk1-P5.zscore 8.774930429549238
TestU01 Block Alphabit(8-2) swalk_RandomWalk1-P3.zscore 9.030096758733212
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P5.zscore 9.0177425228957
TestU01 Block Alphabit(9-16) swalk_RandomWalk1-P3.zscore 9.10199059531798
TestU01 Block Alphabit(7-2) string_HammingCorr.zscore 9.144361264030881
TestU01 Block Alphabit(8-1) swalk_RandomWalk1-P2.zscore 9.187750660120239
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P3.zscore 9.19019799065611
TestU01 Block Alphabit(8-32) swalk_RandomWalk1-P1.zscore 9.276479571074388
TestU01 Block Alphabit(8-4) smutin_MultinomialBitsOver.zscore 9.3511928799561
TestU01 Block Alphabit(8-16) swalk_RandomWalk1-P1.zscore 9.354514906833456
TestU01 Block Alphabit(8-32) swalk_RandomWalk1-P1.zscore 9.369230034014013
TestU01 Block Alphabit(9-16) smutin_MultinomialBitsOver.zscore 9.410255371633733
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P4.zscore 9.469535862685407
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P1.zscore 9.5391695192632
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P2.zscore 9.541620500125267
TestU01 Block Alphabit(9-8) swalk_TandomWalk1-P4.zscore 9.541673272064173
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P4.zscore 9.557240996737608
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P4.zscore 9.60388789452168
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P2.zscore 9.649405080023055
TestU01 Block Alphabit(2-4) smutin_MultinomialBitsOver.zscore 9.657598846077272
TestU01 Block Alphabit(3-1) smutin_MultinomialBitsOver.zscore 9.6592690623558
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P2.zscore 9.70116513539043
TestU01 Block Alphabit(3-2) smutin_MultinomialBitsOver.zscore 9.70981263435675
TestU01 Block Alphabit(2-1) smutin_MultinomialBitsOver.zscore 9.719257047696367
TestU01 Block Alphabit(8-2) swalk_RandomWalk1-P5.zscore 9.720162270549265
TestU01 Block Alphabit(9-16) swalk_RandomWalk1-P1.zscore 9.73620336534092
TestU01 Block Alphabit(8-8) swalk_RandomWalk1-P3.zscore 9.737098039157063
TestU01 Block Alphabit(8-32) swalk_RandomWalk1-P3.zscore 9.73800405801164
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P1.zscore 9.755449452901162
TestU01 Block Alphabit(8-16) swalk_RandomWalk1-P2.zscore 9.75758184113789
TestU01 Block Alphabit(9-4) swalk_RandomWalk1-P3.zscore 9.76441425054065
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P4.zscore 9.778572288585506
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P5.zscore 9.780000000000000
TestU01 Block Alphabit(4-16) smutin_MultinomialBitsOver.zscore 9.844646215985981
TestU01 Block Alphabit(9-4) swalk_RandomWalk1-P5.zscore 9.84517293616111
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P3.zscore 9.893169795509317
TestU01 Block Alphabit(9-1) string_HammingCorr.zscore 9.916596717423215
TestU01 Block Alphabit(8-11) swalk_RandomWalk1-P3.zscore 9.946014303800515
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P4.zscore 10.011834877312698
TestU01 Block Alphabit(5-16) string_HammingIndep.zscore 10.033532372194907
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P3.zscore 10.045100011317338
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P1.zscore 10.060609438177141
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P3.zscore 10.0943478792794657
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P2.zscore 10.114304525673832
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P1.zscore 10.118532919614108
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P1.zscore 10.240516285768
TestU01 Block Alphabit(9-1) smutin_MultinomialBitsOver.zscore 10.281372449929656
TestU01 Block Alphabit(8-1) smutin_MultinomialBitsOver.zscore 10.33820306111732
TestU01 Block Alphabit(9-32) smutin_MultinomialBitsOver.zscore 10.17452032837408
TestU01 Block Alphabit(9-32) swalk_HammingCorr.zscore 10.205253924251098
TestU01 Block Alphabit(9-3) swalk_RandomWalk1-P3.zscore 10.240530400707992
TestU01 Block Alphabit(3-4) smutin_MultinomialBitsOver.zscore 10.25032538020849
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P2.zscore 10.26742986679923
TestU01 Block Alphabit(9-4) swalk_RandomWalk1-P4.zscore 10.283321757192365
TestU01 Block Alphabit(9-4) swalk_RandomWalk1-P1.zscore 10.2930025288014721
TestU01 Block Alphabit(8-2) swalk_RandomWalk1-P4.zscore 10.29947503338571
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P2.zscore 10.3139212420288457
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P5.zscore 10.340321969716988
TestU01 Block Alphabit(3-16) smutin_MultinomialBitsOver.zscore 10.340763379239773
TestU01 Block Alphabit(2-16) smutin_MultinomialBitsOver.zscore 10.348061706743636
TestU01 Block Alphabit(9-8) swalk_RandomWalk1-P4.zscore 10.380597068389861
TestU01 Block Alphabit(9-16) swalk_RandomWalk1-P5.zscore 10.38225682914664
TestU01 Block Alphabit(9-2) smutin_MultinomialBitsOver.zscore 10.411203370601328
TestU01 Block Alphabit(1-1) smutin_MultinomialBitsOver.zscore 10.436401543273687
TestU01 Block Alphabit(9-4) swalk_RandomWalk1-P2.zscore 10.458848278751079
TestU01 Block Alphabit(9-3) swalk_RandomWalk1-P5.zscore 10.492038274713806
TestU01 Block Alphabit(9-32) smutin_MultinomialBitsOver.zscore 10.522001318809
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P5.zscore 10.523200130604465
TestU01 Block Alphabit(9-2) swalk_RandomWalk1-P5.zscore 10.60472334691097
TestU01 Block Alphabit(7-10) string_HammingCorr.zscore 10.6591722155975
TestU01 Block Alphabit(4-4) string_HammingCorr.zscore 10.746127769810889
TestU01 Block Alphabit(4) smutin_MultinomialBitsOver.zscore 10.730685014576846
TestU01 Block Alphabit(6-32) string_HammingIndep.zscore 10.785745223978218
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P5.zscore 10.789816189921341
TestU01 Block Alphabit(8-1) swalk_RandomWalk1-P1.zscore 10.79916402348215
TestU01 Block Alphabit(8-4) swalk_RandomWalk1-P4.zscore 10.803862507772748
TestU01 Block Alphabit(8-4) swalk_RandomWalk1-P2.zscore 10.856198104316218
TestU01 Block Alphabit(8-4) swalk_RandomWalk1-P1.zscore 10.96038554468454
TestU01 Block Alphabit(5-4) string_HammingIndep.zscore 10.972196995964555
TestU01 Block Alphabit(9-32) swalk_RandomWalk1-P2.zscore 10.919375653623017
TestU01 Block Alphabit(6-16) string_HammingIndep.zscore 11.02084635564276
TestU01 Block Alphabit(4-4) smutin_MultinomialBitsOver.zscore 11.024565297836812
TestU01 Block Alphabit(2-32) smutin_MultinomialBitsOver.zscore 11.041629462145137
TestU01 Block Alphabit(8-8) swalk_RandomWalk1-P1.zscore 11.15383377986334
TestU01 Block Alphabit(9-1) swalk_RandomWalk1-P4.zscore 11.155146927412748
TestU01 Block Alphabit(7-8) string_HammingCorr.zscore 11.157010506454086
TestU01 Block Alphabit(6-4) string_HammingIndep.zscore 11.236609203958274
TestU01 Block Alphabit(6-1) string_HammingIndep.zscore 11.238933165107045
TestU01 Block Alphabit(1-1) smutin_MultinomialBitsOver.zscore 11.592986278990691
TestU01 Block Alphabit(5-2) string_HammingIndep.zscore 11.80605704710731
TestU01 Block Alphabit(7-1) string_HammingCorr.zscore 11.412354024521774
TestU01 Block Alphabit(9-16) swalk_RandomWalk1-P2.zscore 11.542717688362468
TestU01 Block Alphabit(8-8) smutin_MultinomialBitsOver.zscore 11.551462947721202
TestU01 Block Alphabit(8-4) swalk_RandomWalk1-P3.zscore 11.627052122988623
TestU01 Block Alphabit(4-2) smutin_MultinomialBitsOver.zscore 11.77011375045788
TestU01 Block Alphabit(5-32) string_HammingIndep.zscore 11.785276828869554
TestU01 Block Alphabit(6-2) string_HammingIndep.zscore 12.141292911582346
TestU01 Block Alphabit(5-8) string_HammingIndep.zscore 12.40033673039414
TestU01 Block Alphabit(6-8) string_HammingIndep.zscore 12.4004278919888
```



In []: