## 一、第一部分，脉冲编码

1.1、纠正之前对论文论文中编码的误解，仔细观察图1中的图2，会发现论文中的编码结果是由01串组成的序列，也就是根据图3的流程图，而脉冲生成器处生成的是1序列，其他情况生成的是0序列。补充，论文中说明使用的是Temporal coding，查看资料https://en.wikipedia.org/wiki/Neural\_coding#Temporal\_coding。

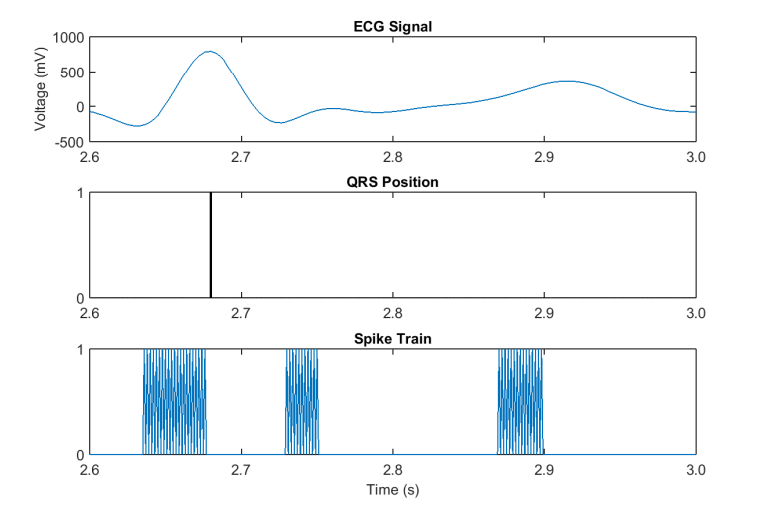


图1

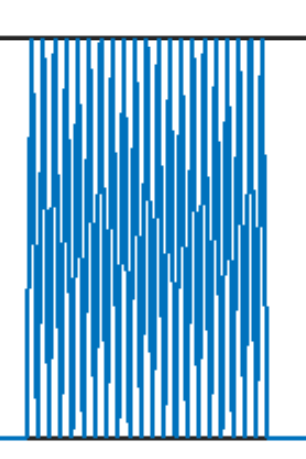


图2

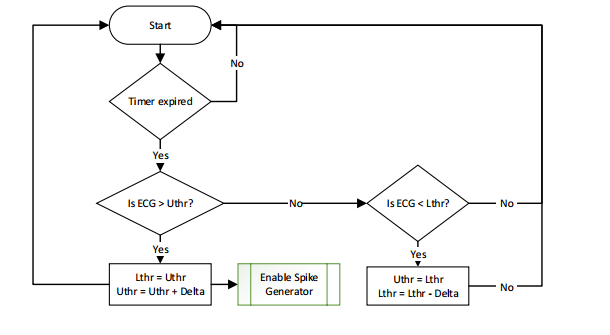


图3

1.2、根据论文中的编码规则，我编写了编码函数名为spike\_encoder，delta设置为20，生成了如图4、图5的结果，分别从不同精度观察，发现图5的结果并不理想，尝试过其他参数，目前这个应该是比较好的了。

spike\_encoder

delta **=** 20

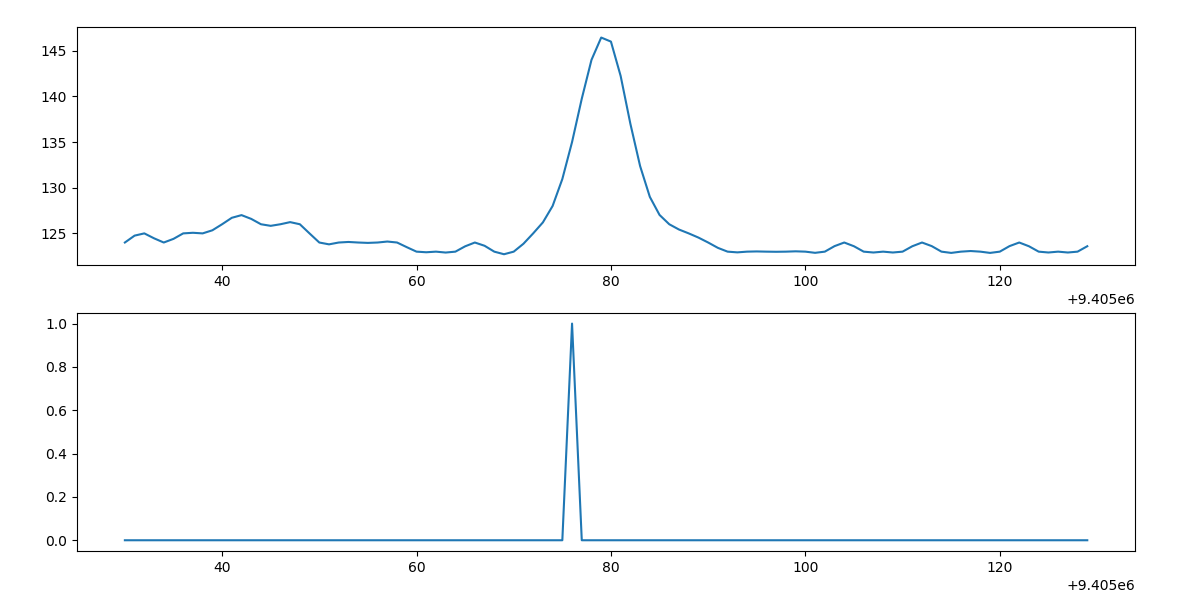


图4

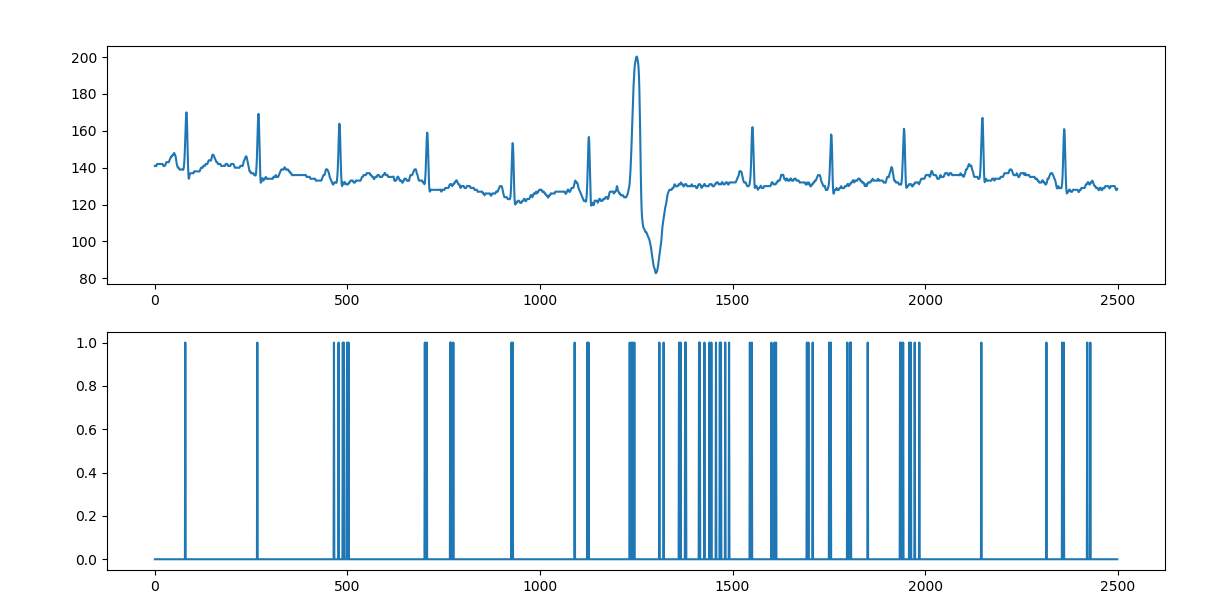


图5

1.3、由于第二部分的结果并不理想，通过观察心电数据，觉得如果直接将脉冲后一数值与前一数值进行比较可能会有更好的结果，即values[i+1] - values[i]与delta值进行比较，大于则返回1，否则返回0.得到实验结果如图6所示，个人感觉还算理想。

**spike\_encoder\_rela**

delta **=** 2

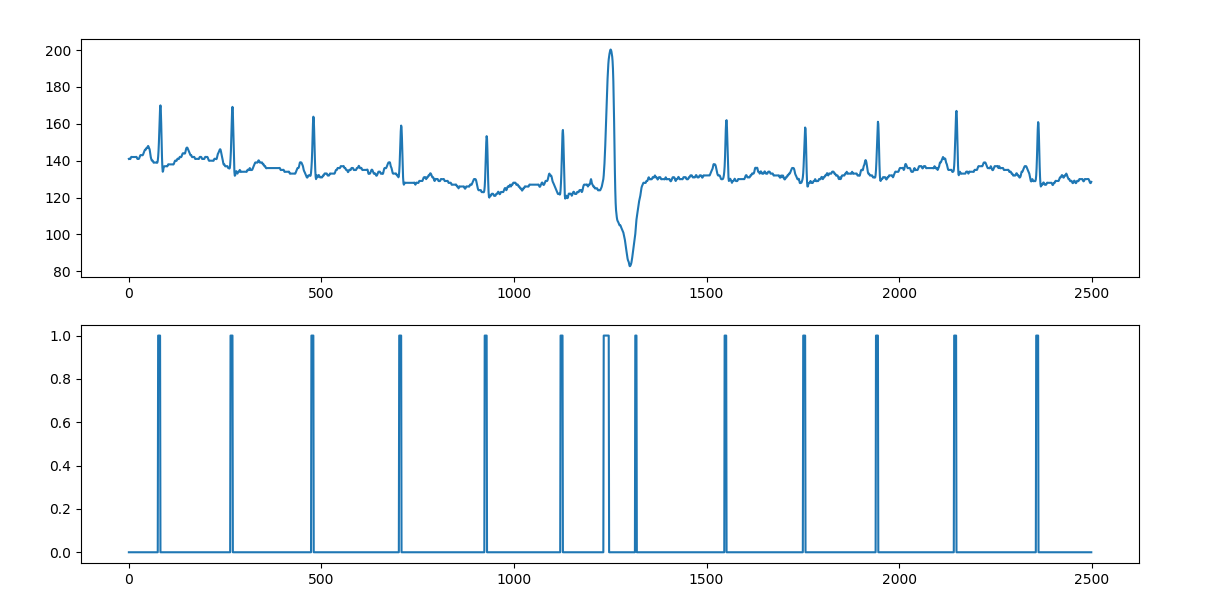
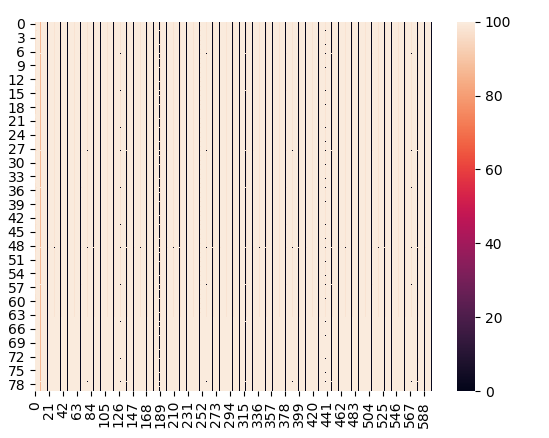


图6

## 第二部分，Spiking Neurons-Based Liquid

2.1、纵坐标：0-63是兴奋神经元，64-79是抑制神经元

横坐标：0-599是为一个HI中包含的SI



## 第三部分，心率计算

3.1、粒子群优化算法

self.iter\_max **=** 10000  
self.pop\_size **=** 100  
self.c1 **=** 0.5  
self.c2 **=** 0.5  
self.err\_crit **=** 0.00001

m **=** 2

实验结果：



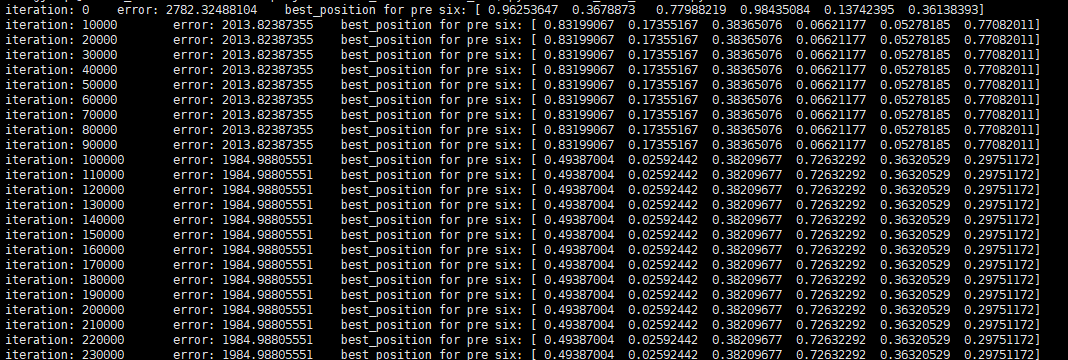
|  |
| --- |
| bestW:  [ 0.70719499 0.55913326 0.54627015 0.78311277 0.29201418 0.30269589  0.62586402 0.53227438 0.74411203 0.26339712 0.49347664 0.66977305  0.62549252 0.53440533 0.37368245 0.32775995 0.62887489 0.21518126  0.59002783 0.45102303 0.51020193 0.54710049 0.46742106 0.44543695  0.16429556 0.45457104 0.66456842 0.47430378 0.69206613 0.43596584  0.31787526 0.47925806 0.34089571 0.53416726 0.62517851 0.51425041  0.75580994 0.47099489 0.4279587 0.37400224 0.5707867 0.57985894  0.50738235 0.50405421 0.65688683 0.623082 0.38120706 0.39475517  0.49195961 0.82150626 0.52700137 0.35843311 0.52258836 0.50671968  0.49650391 0.4906488 0.50533753 0.41861932 0.59615013 0.36692056  0.36134885 0.38508584 0.51720272 0.44433742 0.51568019 0.59304941  0.56022518 0.63747234 0.48062421 0.49447626 0.25426249 0.25500756  0.42192658 0.4207336 0.51354338 0.80382532 0.51315667 0.45656621  0.38073997 0.58519543]  bestC0:  [ 0.44361765 0.30979802 0.53578847 0.25855301 0.58382585 0.50233993  0.33746202 0.57023936 0.62068357 0.49689578 0.48567533 0.36419003  0.52490458 0.5548483 0.51955577 0.32409643 0.35818595 0.59654864  0.507675 0.75688578 0.54084045 0.48936722 0.51503531 0.47508396  0.40058621 0.2845973 0.5124375 0.49712283 0.39519945 0.48632713  0.57725216 0.50349883 0.424744 0.42020696 0.62426475 0.61717168  0.50422802 0.66806082 0.51777882 0.58163098 0.48270555 0.75739144  0.51957989 0.46950811 0.39547536 0.64269275 0.54146908 0.54277411  0.65876711 0.54315973 0.51257148 0.44005153 0.57756037 0.47504042  0.44963852 0.48809323 0.44650127 0.63274356 0.52655762 0.363768  0.59105431 0.56289586 0.37580704 0.41706183 0.38643491 0.4467732  0.55565945 0.39195601 0.45028997 0.59835921 0.56179822 0.45356523  0.50719835 0.40694899 0.44866177 0.69345102 0.42346504 0.30797696  0.49140156 0.63368719]  bestC1:  [ 0.49557154 0.57321908 0.51947924 0.46142281 0.55306544 0.68494054  0.35167619 0.39824476 0.57338853 0.2477024 0.50918107 0.58930002  0.28578359 0.68603331 0.77413212 0.66360923 0.75912646 0.36698124  0.62116726 0.59594272 0.63207276 0.47209492 0.32472334 0.60563965  0.42890983 0.45402997 0.56842292 0.72242612 0.62133639 0.47591532  0.38442991 0.48217868 0.55296688 0.59674764 0.38726471 0.35181571  0.34424868 0.51699509 0.47435005 0.3985522 0.67360077 0.47706571  0.18323016 0.45488331 0.50000204 0.41360152 0.46719147 0.60593923  0.25875239 0.59624391 0.41246122 0.60762179 0.51063556 0.67208955  0.5869858 0.37972101 0.38673208 0.35128535 0.52769131 0.57824482  0.56772493 0.40480348 0.28937375 0.32981836 0.21359861 0.58354089  0.46083639 0.37068724 0.3895056 0.38001773 0.68413226 0.37969742  0.47163689 0.44822092 0.43292607 0.53315531 0.50361656 0.49611957  0.6632969 0.52842343]  bestError:  906.698511095  probability: [[ 0.49987267 0.50012733]  [ 0.59825229 0.40174771]  [ 0.59729603 0.40270397]  ...,  [ 0.59856535 0.40143465]  [ 0.59856535 0.40143465]  [ 0.4990071 0.5009929 ]] |

3.2、monte carlo算法

self.n **=** 1000000

m **=** 2

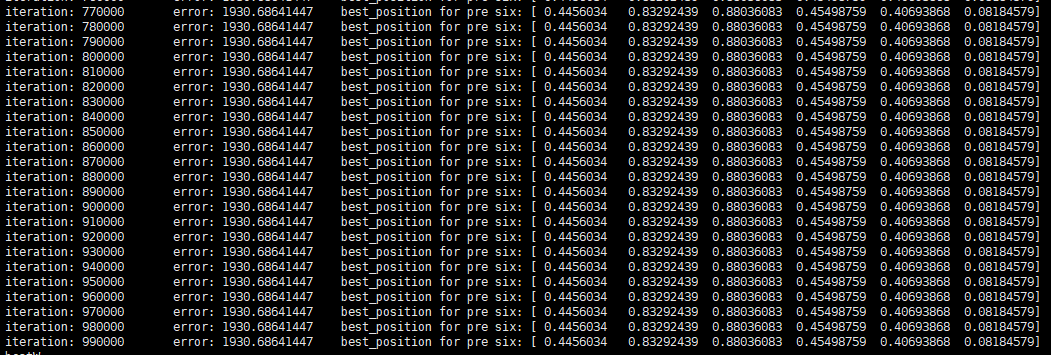
实验结果：



。

。

。

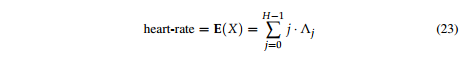


|  |
| --- |
| bestW:  [ 0.4456034 0.83292439 0.88036083 0.45498759 0.40693868 0.08184579  0.20287661 0.79834511 0.83113546 0.99484338 0.23587952 0.42748531  0.14570807 0.76572051 0.23701308 0.91379232 0.24018558 0.88508707  0.15729886 0.02097906 0.15714828 0.03791852 0.28573519 0.51225047  0.27411533 0.05588212 0.37920595 0.59618636 0.26001664 0.77725402  0.78180211 0.38369227 0.80203494 0.7372773 0.25729429 0.70071894  0.71366857 0.61604309 0.19027265 0.74570746 0.04397352 0.3478282  0.0886744 0.75634859 0.73291849 0.16237241 0.58380705 0.23503365  0.42859336 0.93373272 0.39001024 0.39115629 0.38849346 0.14471402  0.94366234 0.20154129 0.12861059 0.13510793 0.41852152 0.0397039  0.68066928 0.85691268 0.41501202 0.82110077 0.62401373 0.04113166  0.45460682 0.59653335 0.22703703 0.25140446 0.80308711 0.28876807  0.94212366 0.70947901 0.46026122 0.31675508 0.84867625 0.22102777  0.39546811 0.68022422]  bestC0:  [ 0.09639289 0.68192458 0.62503778 0.81978425 0.38385173 0.58560844  0.26269317 0.60161044 0.51475004 0.45806363 0.97596772 0.98031736  0.21595128 0.43466293 0.68775008 0.71046936 0.32789382 0.55291257  0.11936863 0.58589454 0.42171723 0.53281791 0.20498419 0.07581633  0.67362884 0.79370107 0.39994043 0.81832719 0.02816592 0.87888155  0.62193821 0.24457678 0.23357448 0.72327009 0.2059735 0.70250942  0.35353001 0.02720154 0.38962177 0.31138173 0.62580568 0.43202479  0.36985618 0.79379353 0.54969286 0.26204451 0.46190561 0.38049531  0.48526024 0.81619956 0.2247762 0.872335 0.33912192 0.63763831  0.76797151 0.1258977 0.27553342 0.10057003 0.49082604 0.35698178  0.62940428 0.61776957 0.2260349 0.67010261 0.88977476 0.15498448  0.17884704 0.44217879 0.42760122 0.42634787 0.36444814 0.43544327  0.40218067 0.75831974 0.14245582 0.22469147 0.57568625 0.01946993  0.59722413 0.66590977]  bestC1:  [ 0.36746165 0.25543005 0.55015459 0.85967222 0.3360151 0.44589494  0.43892114 0.42088049 0.13299453 0.66960343 0.75849098 0.99734688  0.87950747 0.2794367 0.94792903 0.53300332 0.21639268 0.2983807  0.94301783 0.37945471 0.3755877 0.40398231 0.63070886 0.433729  0.26007475 0.12157884 0.39825782 0.12565716 0.4460984 0.67593343  0.1096745 0.54822733 0.81741133 0.3528361 0.25884693 0.54364714  0.1325785 0.44521485 0.41681076 0.48432383 0.58809202 0.16911657  0.4863418 0.57104265 0.51562928 0.20083693 0.88532477 0.23931121  0.22994939 0.13098728 0.00971296 0.30515659 0.78705731 0.9707151  0.11018894 0.15305266 0.44816313 0.18693677 0.11264172 0.44597617  0.60501458 0.39693641 0.69189575 0.16549273 0.32574217 0.34234195  0.05016712 0.45679267 0.14568117 0.09226096 0.73896547 0.72179214  0.51399748 0.32344328 0.53063026 0.27301085 0.05741056 0.42069437  0.20062151 0.19816433]  bestError:  1930.68641447  probability: [[ 0.45800006 0.54199994]  [ 0.57952073 0.42047927]  [ 0.58172257 0.41827743]  ...,  [ 0.58288945 0.41711055]  [ 0.58288945 0.41711055]  [ 0.45875907 0.54124093]] |

3.3、predict heart rate

论文中使用的心率预测函数如下：





但由于该算法的时间复杂度至少为2^H次方，而H为1分钟最多心跳数，本程序中默认100，时间复杂度太高。

因此目前使用的是



实验结果如下：

heartRate: 600

3.4、actual heart rate

读取标签中心跳发生的脉冲下标，

heartRate:77

# 准确率评估

使用方法：

