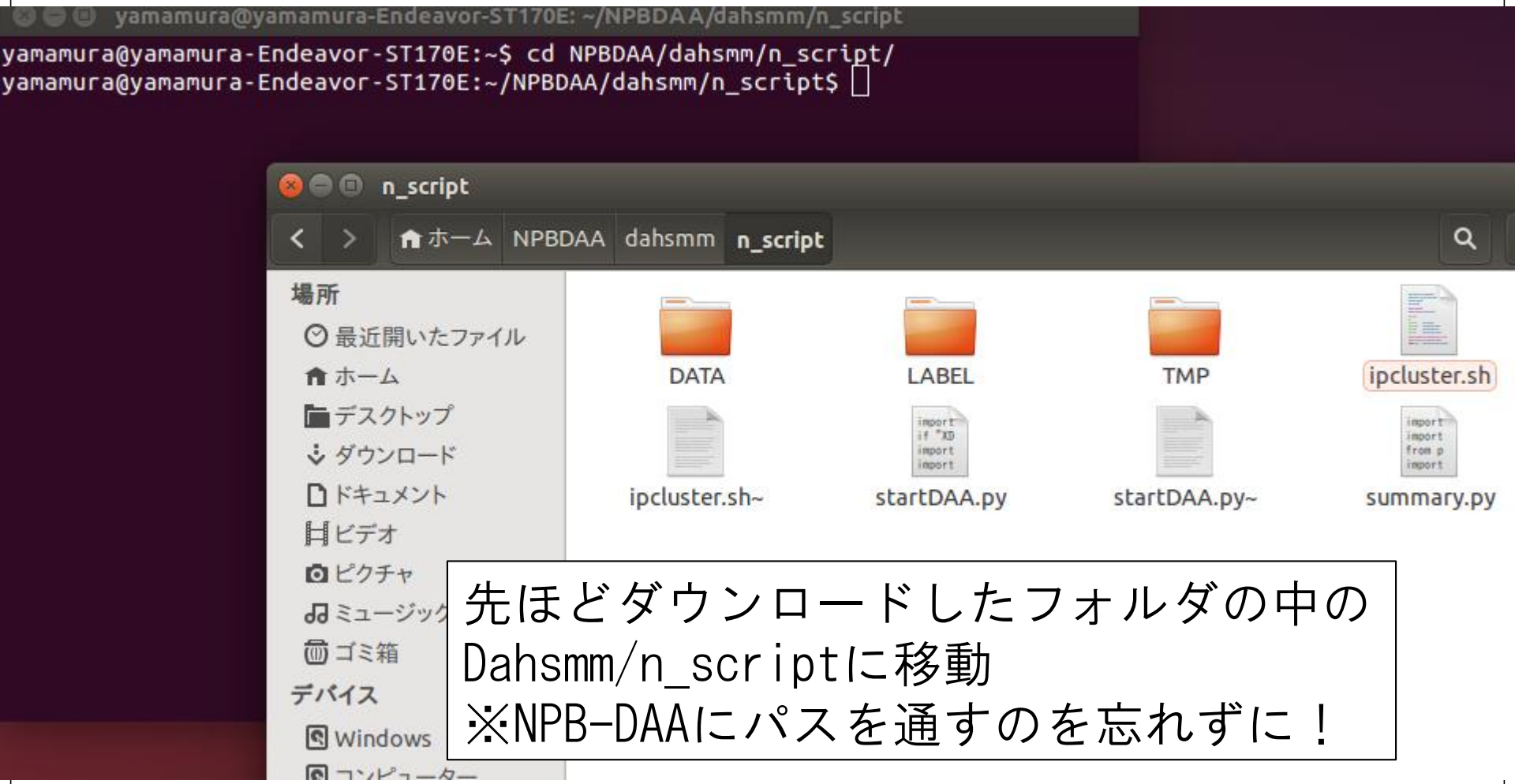
LST勉強会　第２回

NPBDAAの起動

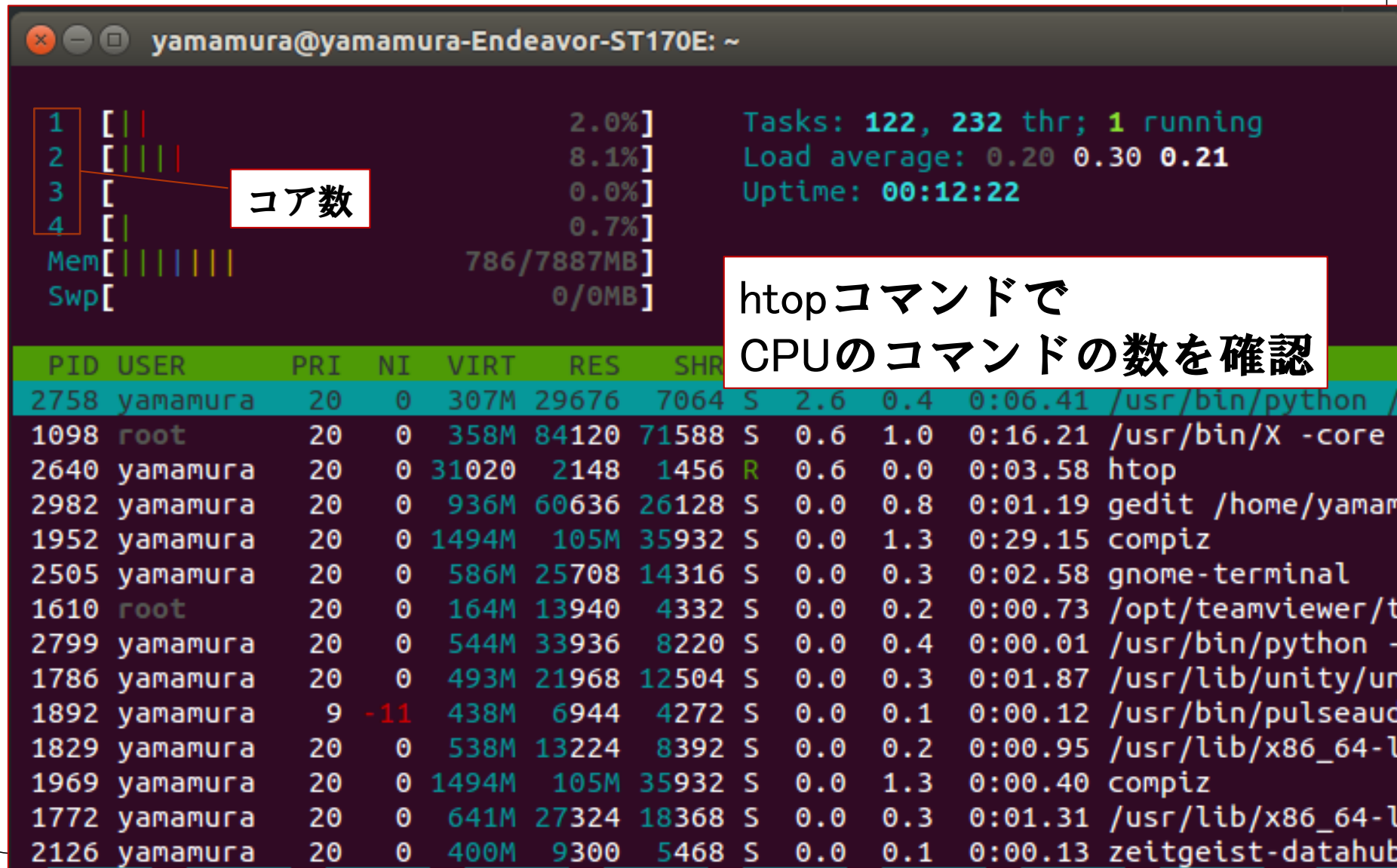
Step1 googleドライブからDAAをダウンロード

Step2 カレントディレクトリの移動

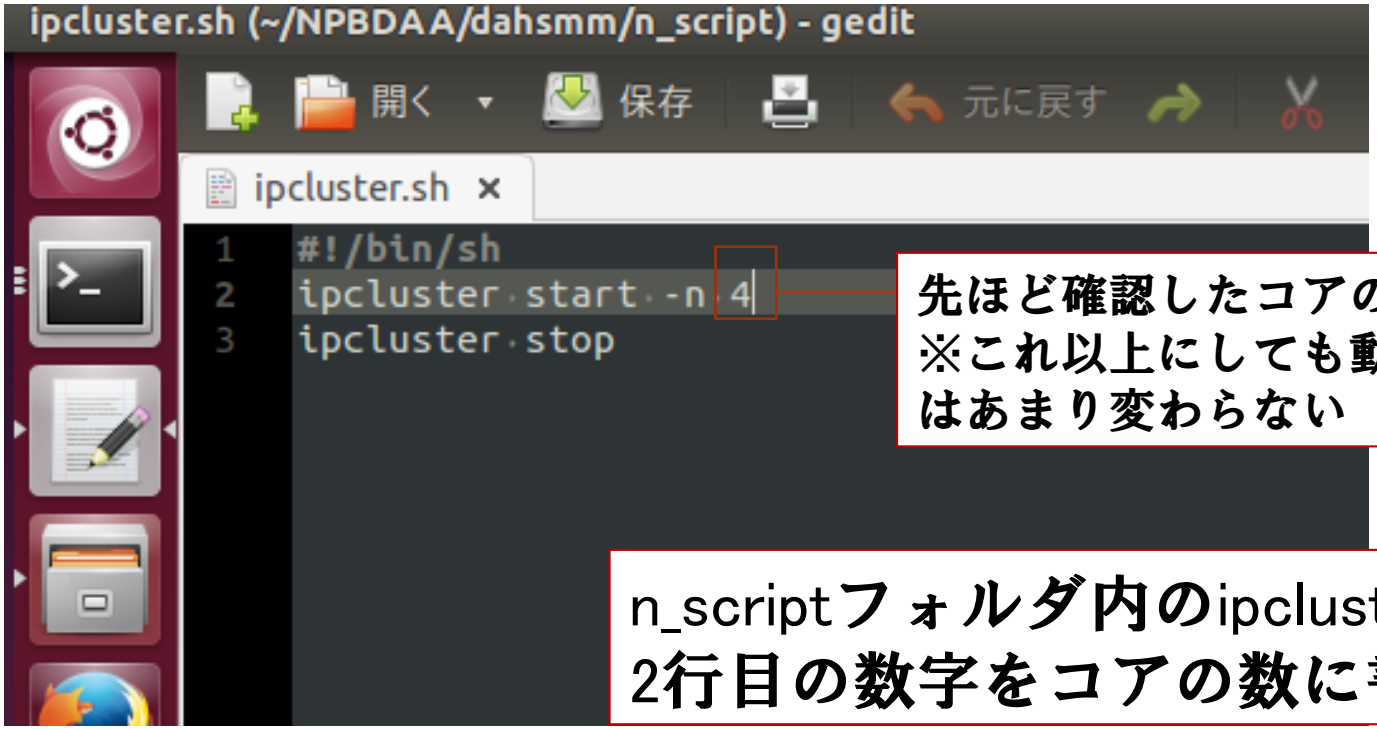


先ほどダウンロードしたフォルダの中の
Dahsmm/n_scriptに移動
※NPB-DAAにパスを通すのを忘れずに！

Step3 コアの数の設定



Step3 コアの数の設定



先ほど確認したコアの数に書き換える
※これ以上にしても動くがパフォーマンスはあまり変わらない

n_scriptフォルダ内のipcluster.shを開き
2行目の数字をコアの数に書き換える

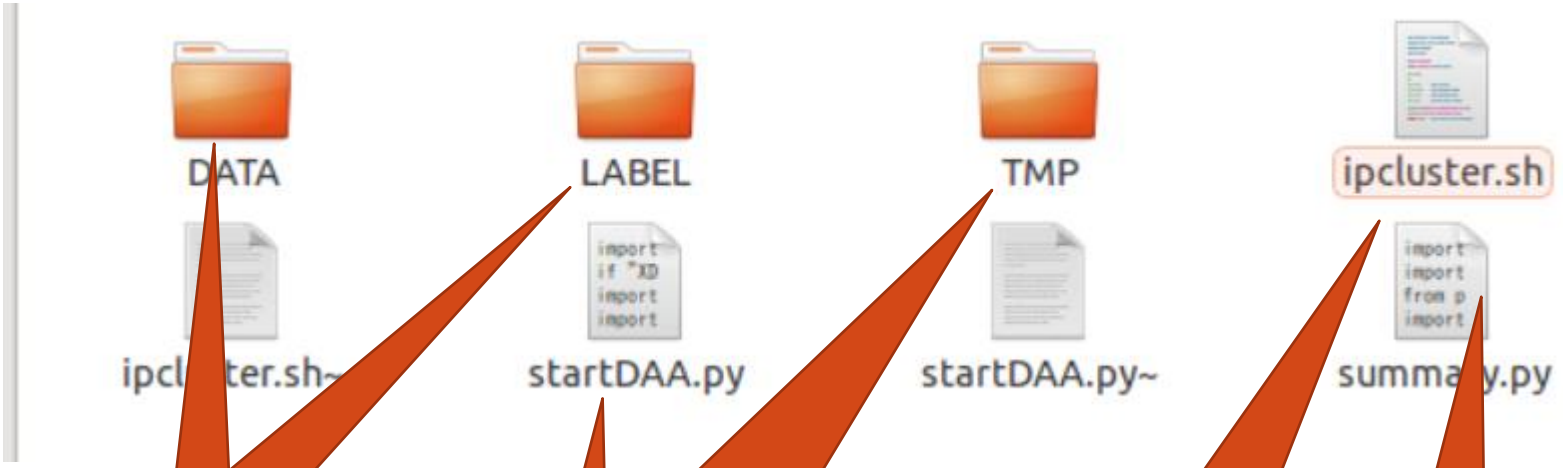
Step3 コアの数の設定

```
yamamura@yamamura-Endeavor-ST170E:~/NPBDAA/dahsmm/n_script$ sh ipcluster.sh
2015-07-29 15:37:22.793 [IPClusterStart] Using existing profile dir: u'/home/yam
amura/.config/ipython/profile_default'
2015-07-29 15:37:22.842 [IPClusterStart] Removing pid file: /home/yamamura/.conf
ig/ipython/profile_default/pid/ipcluster.pid
2015-07-29 15:37:22.842 [IPClusterStart] Starting ipcluster with [daemon=False]
2015-07-29 15:37:22.843 [IPClusterStart] Creating pid file: /home/yamamura/.conf
ig/ipython/profile_default/pid/ipcluster.pid
2015-07-29 15:37:22.843 [IPClusterStart] Starting Controller with LocalControlle
rLauncher
2015-07-29 15:37:23.843 [IPClusterStart] Starting 4 Engines with LocalEngineSetL
auncher
```

sh コマンドでipcluster.shを起動

この時コアが全て動いていたらOK

Step3.5 NPBDAAの起動に必要なファイル



真ラベルデータフォルダ（60データ

音声MFCC→DSAE3次元データ（60データ

データ読み込み再利用用オブジェクト保存フォルダ

NPB-DAA実行スクリプト

ipcluster用（並列処理

結果図，単語リストなど出力ファイル生成スクリプト

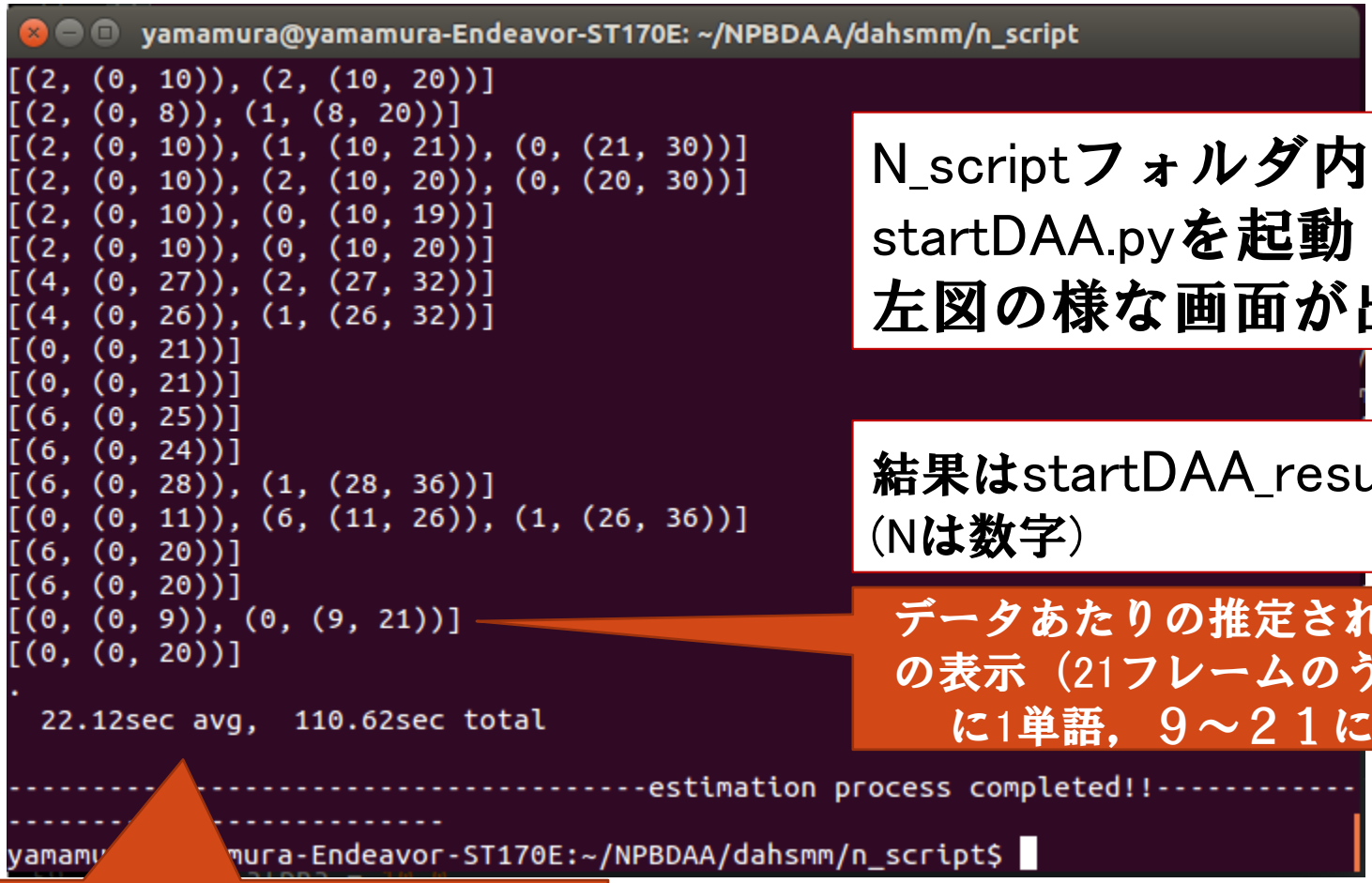
Step4 NPBDAAの起動

startDAA.py

```
19  LETTER_N
20  WORD_N
21  DATA_N
22  model_hypparams
23  word_model_params
24  obs_hypparams
25  dur_hypparams
26  filename'''.split('\n')
27  #-----------------------------------multi process function-----------------------------
28  def multi_dump_object(data,model,f,count):
29      print f," dumping..."
30      fp = open("TMP/"+f+".dump", 'w')
31      pickle.dump(HSMMState(data,model), fp)
32      fp.close()
33      count.value = count.value + 1
34      print f, 'dump finish-->count:',count.v
35  #-------------------------------------main
36  def main(result_dir = None):
37      if result_dir == None:
38          result_dir = get_result_dir(__file_
39      os.mkdir(result_dir)
40      param_path = os.path.join(result_dir, 'parameter.json')
41      fig_title_path = os.path.join(result_dir, 'fig_title.json')
42  #initialize model params#
43      #O(T*L_max*W_max^2*d_max^3)
44      ITER_N = 5
45      LETTER_N = 7
46      WORD_N = 7
47      DATA_N = 60
48      obs_dim = 3
49      model_hypparams = {'state_dim': WORD_N, 'alpha': 10.0, 'gamma': 10.0}
50      word_model_params = {'letter_type': LETTER_N, 'rho': 10}
51      obs_hypparams = {'mu_0': np.zeros(obs_dim), 'sigma_0': np.eye(obs_dim) ,'kappa_0': 0.01,
52      dur_hypparams = {'alpha_0': 50.0, 'beta_0': 10.0}
53      length_dist = pyhsmm.distributions.PoissonDuration(alpha_0 = 30, beta_0 = 10, lmbda = 3)
54  #setting#
```

StartDAA.pyを開き
44行目のITER_Nの数字を１００から５に変更
(ITER_Nはイテレーションの回数
今回は試験的に起動するため小さく設定)

Step4 NPBDAAの起動



N_scriptフォルダ内の
startDAA.pyを起動
左図の様な画面が出ればOK

結果はstartDAA_result_N に保存
(Nは数字)

データあたりの推定された単語境界の表示（21フレームのうち，０～９に1単語，９～２１に一単語）

トータルの時間と
1イテレーションごとの時間

※Keyerror がでた場合はもう一度起動しなおすとうまくいく

Step4.　5 出力ファイル説明

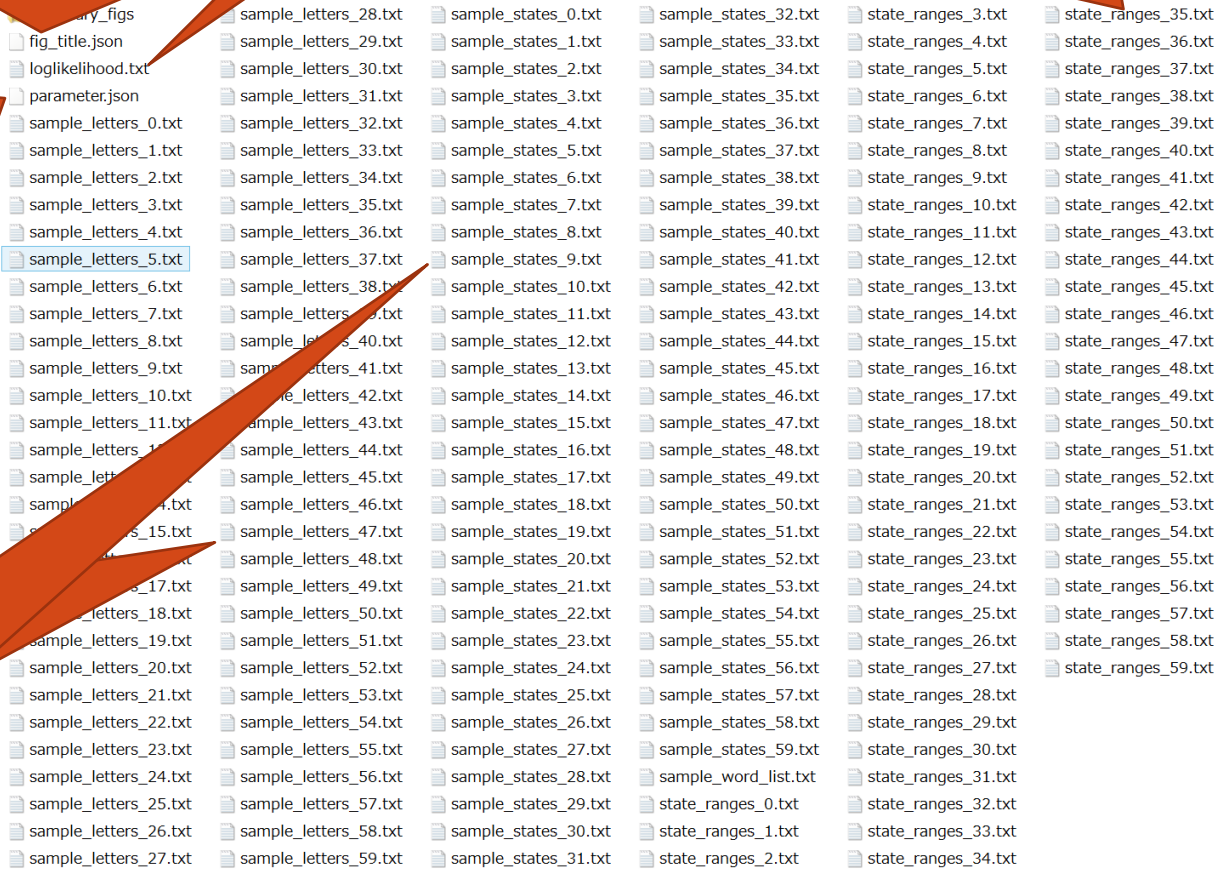
["aioi_aioi", "aioi_aioi2", "aioi_ao", …, "uo_ie", "uo_ie2", "uo_uo", "uo_uo2"]
データ名と順番を保存するファイル

モデルの尤度をイテレーション毎に保存するファイル

単語境界をオブジェクト保存するファイル（機械言語でかかれているので見てもわからない）

{"LETTER_N": 7, "model_hypparams": {"state_dim": 7, "gamma": 10.0, "alpha": 10.0}, "ITER_N": 100, "DATA_N": 60,…, "word_model_params": {"letter_type": 7, "rho": 10}}
設定したパラメータ，実験条件などを保存しておくファイル

sample_letters_*.txt，sample_states_*.txtは*番目のデータにおいてイテレーション毎に推定されたフレームごとの音素,単語ラベルが保存されている



Step4.　5 出力ファイル説明

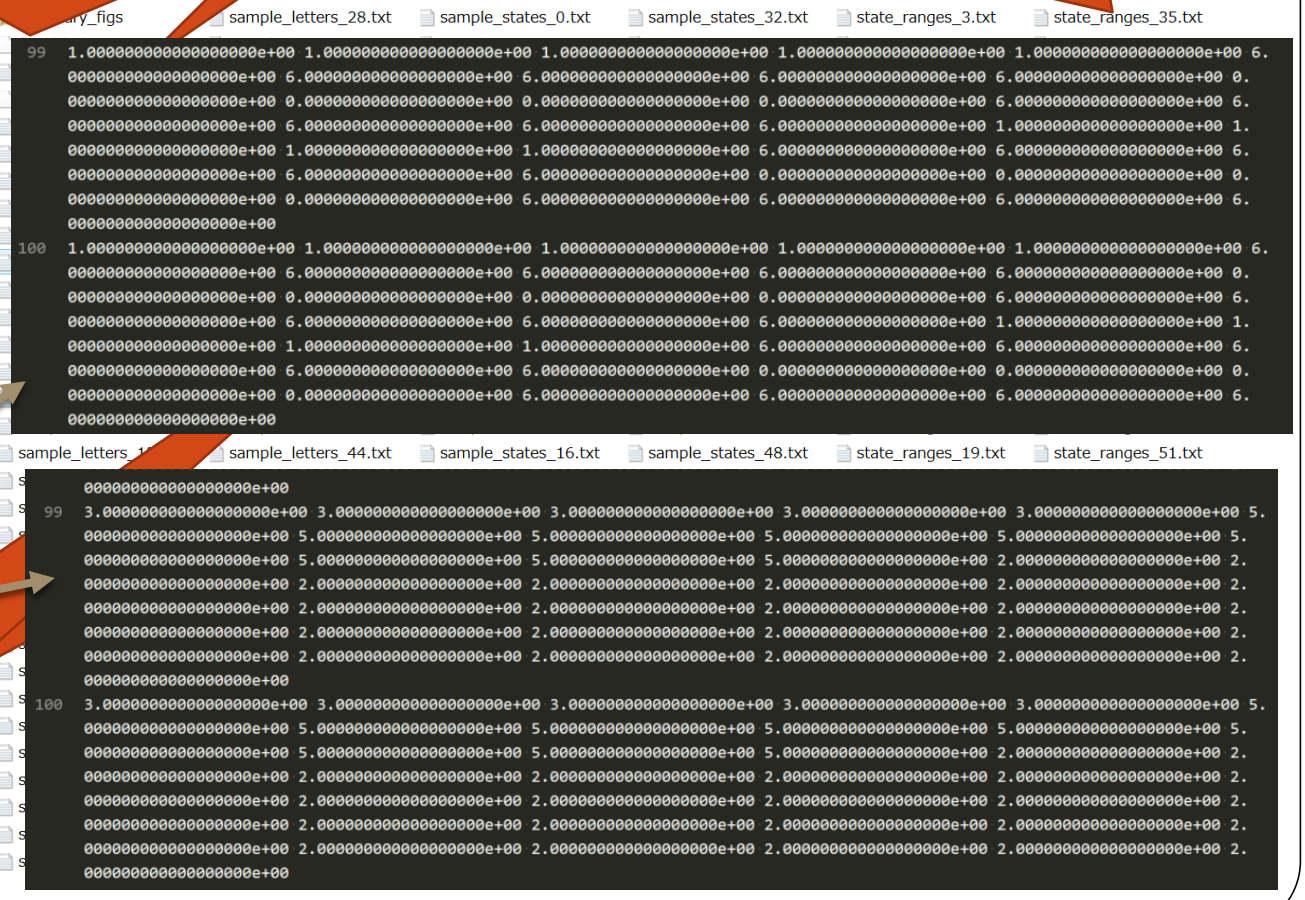
["aioi_aioi", "aioi_aioi2", "aioi_ao", …, "uo_ie", "uo_ie2", "uo_uo", "uo_uo2"]
データ名と順番を保存するファイル

モデルの尤度をイテレーション毎に保存するファイル

単語境界をオブジェクト保存するファイル（機械言語でかかれているので見てもわからない）

{"LETTER_N": 7, "model_hypparams": {"state_dim": 7, "gamma": 10.0, "alpha": 10.0}, "ITER_N": 100, "DATA_N": 60,…, "word_model_params": {"letter_type": 7, "rho": 10}}
設定したパラメータ，実験条件などを保存しておくファイル

sample_letters_*.txt, sample_states_*.txtは*番目のデータにおいてイテレーション毎に推定されたフレームごとの音素,単語ラベルが保存されている



Step5 結果を図示する

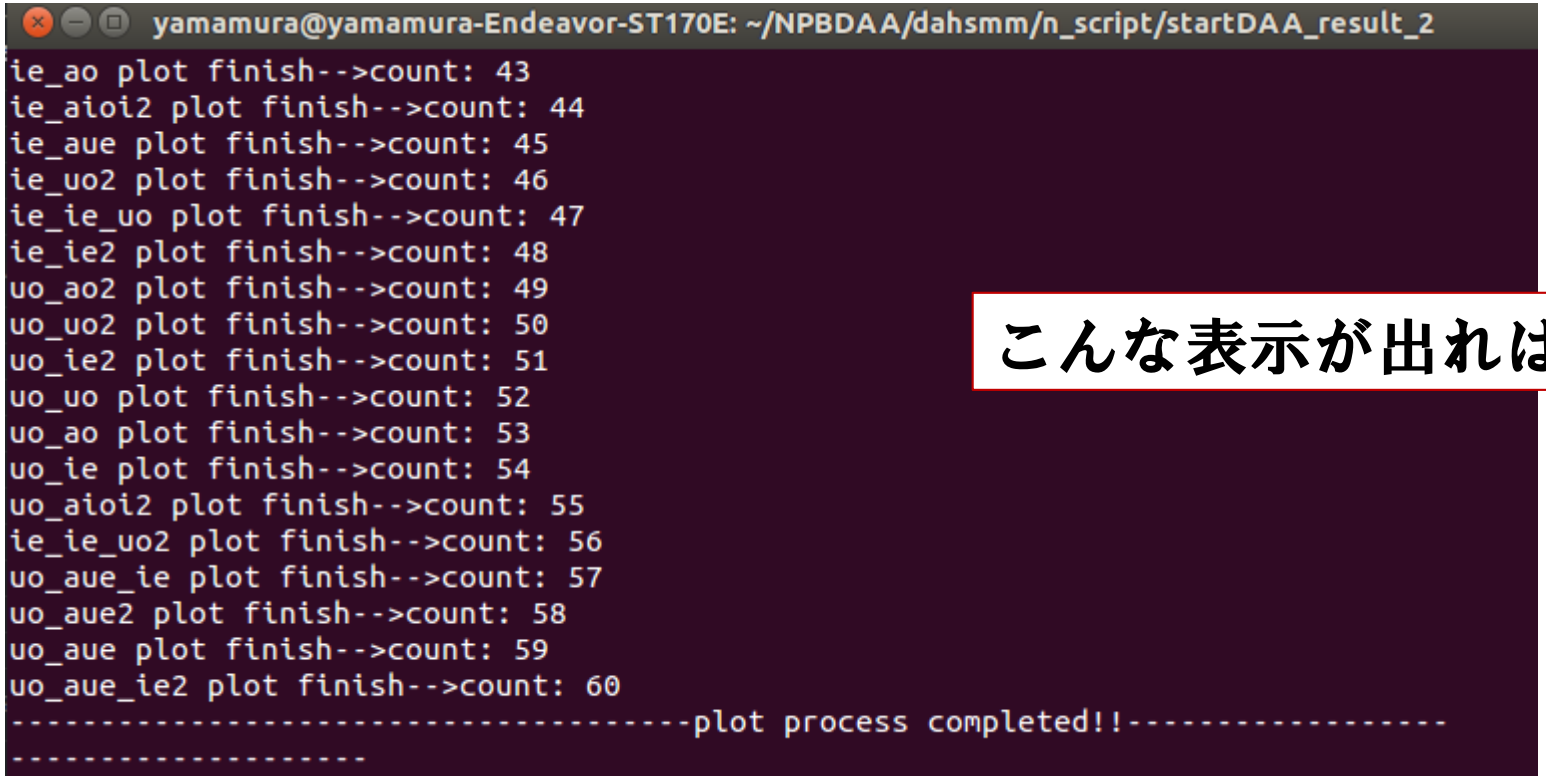
startDAA_result_のフォルダに移動し
summary.pyを起動(summary.pyはn_scriptフォルダにあるので注意)

```
yamamura@yamamura-Endeavor-ST170E:~/NPBDAA/dahsmm/n_script$ cd startDAA_result_
startDAA_result_1/ startDAA_result_2/
yamamura@yamamura-Endeavor-ST170E:~/NPBDAA/dahsmm/n_script$ cd startDAA_result_
startDAA_result_1/ startDAA_result_2/
yamamura@yamamura-Endeavor-ST170E:~/NPBDAA/dahsmm/n_script$ cd startDAA_result_2
yamamura@yamamura-Endeavor-ST170E:~/NPBDAA/dahsmm/n_script/startDAA_result_2$ ipython ../sum
```

一番数字が
新しいフォルダを選択

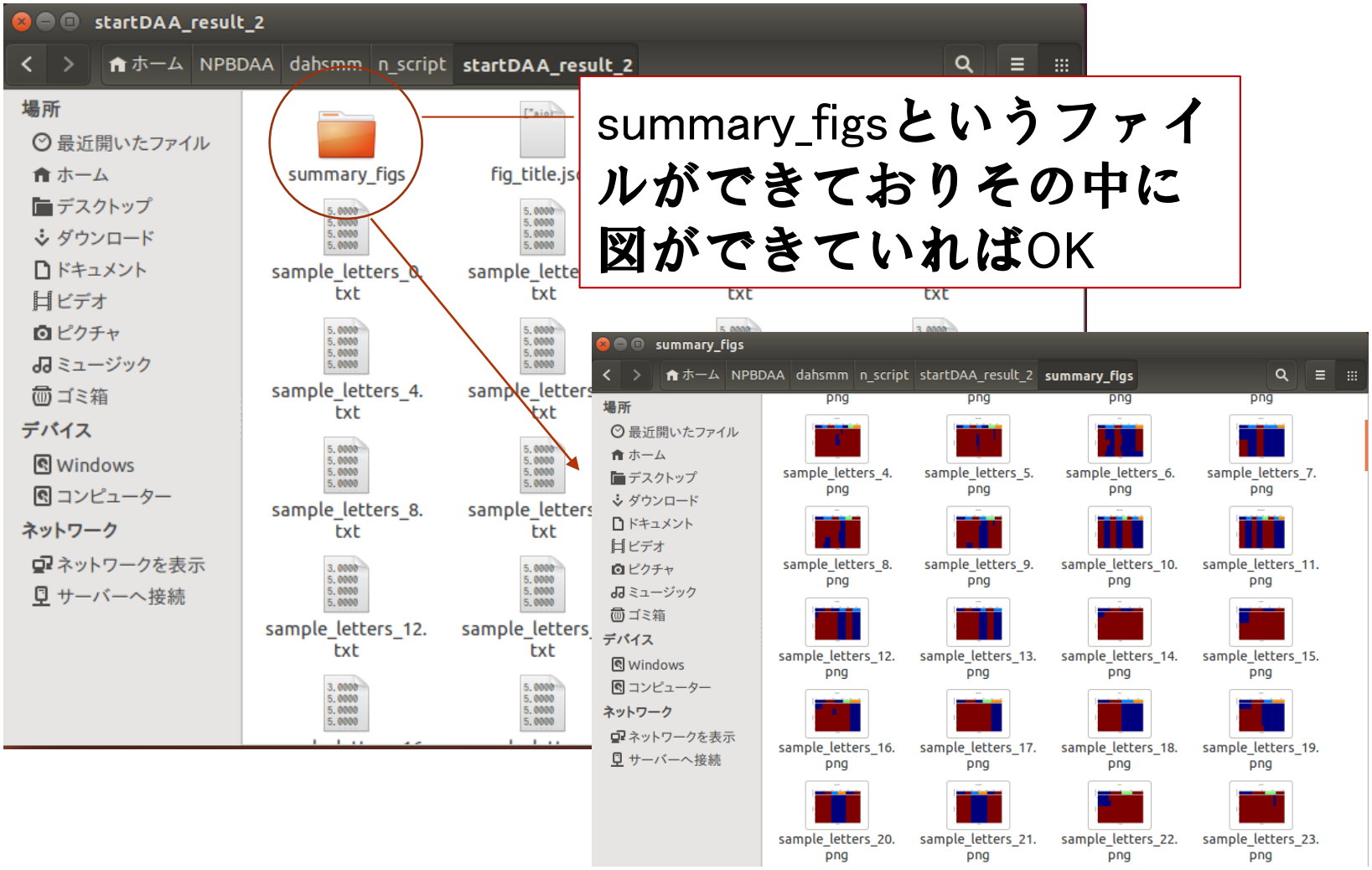
summary,.pyを起動

Step5 結果を図示する

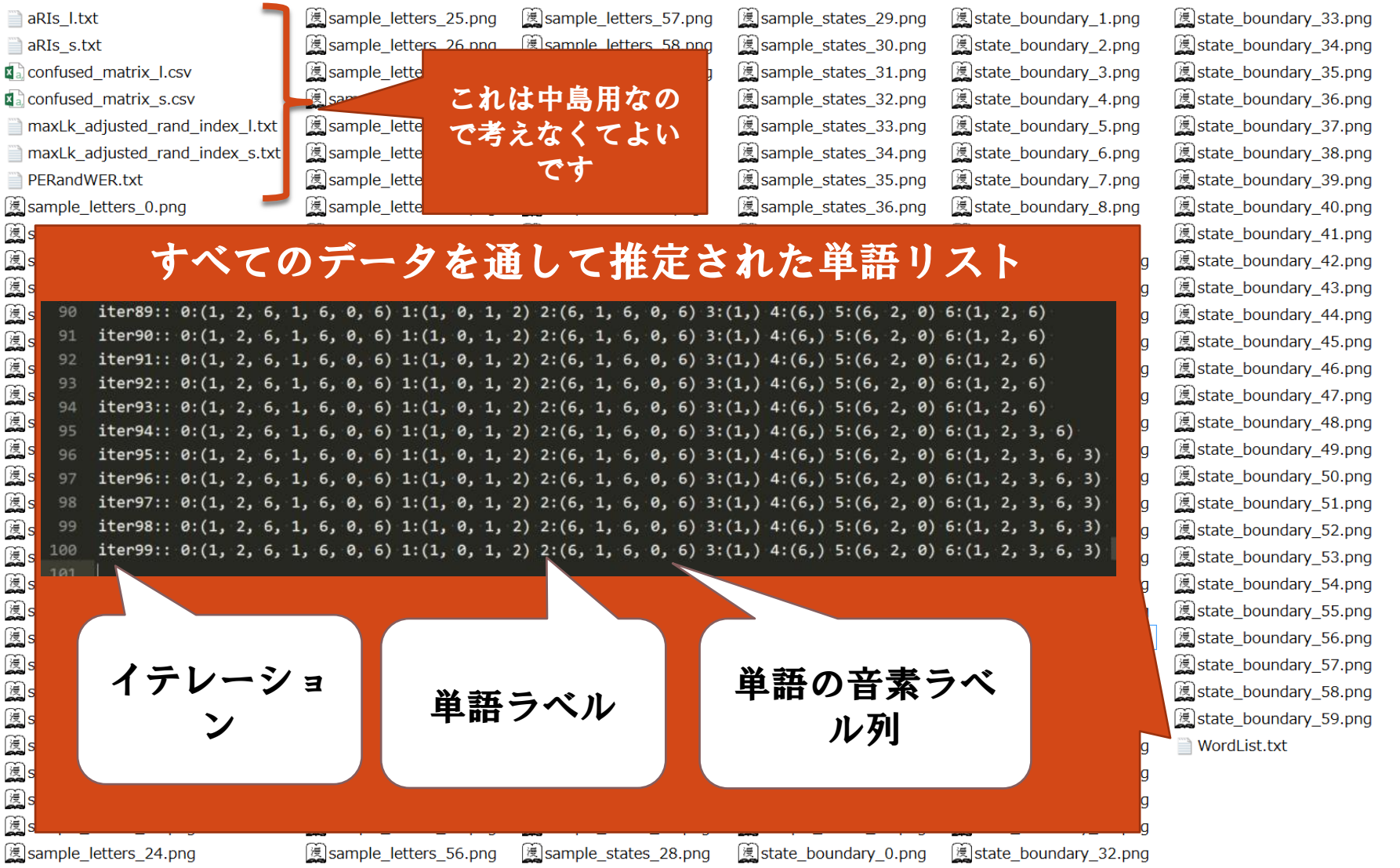


こんな表示が出ればOK

Step5 結果を図示する



Step5. 5 出力ファイル説明



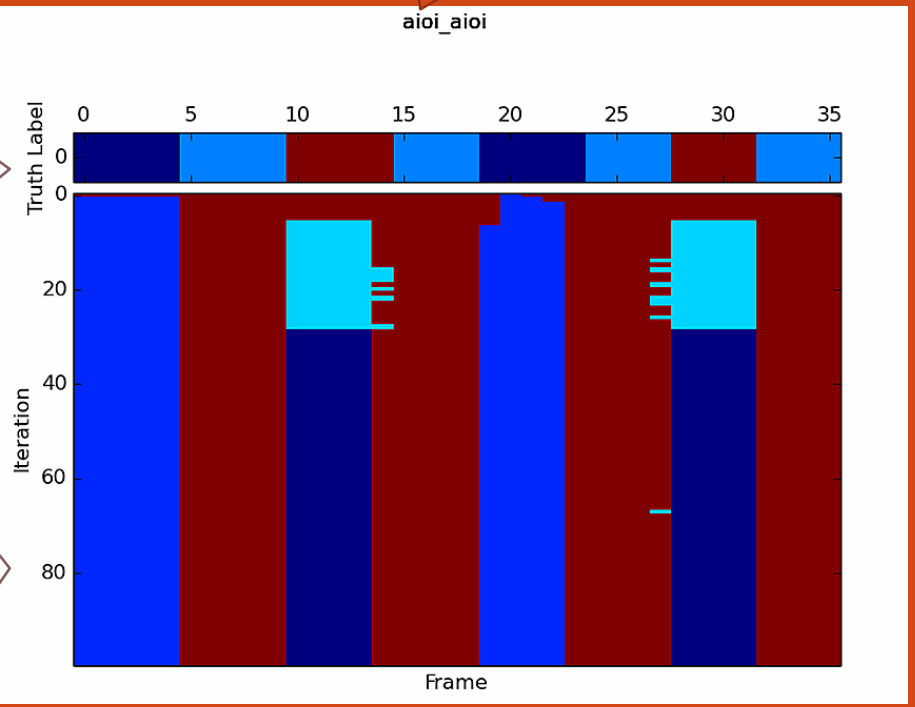
Step5.　5 出力ファイル説明

aRIs_l.txt
aRIs_s.txt
confused_matrix_l.csv
confused_matrix_s.csv
maxLk_adjusted_rand_index_l.txt
maxLk_adjusted_rand_index_s.txt
PERandWER.txt
sample_letters_0.png
sample_letters_1.png
sample_letters_2.png
sample_letters_3.png
sample_letters_4.png
sample_letters_5.png
sample_letters_6.png
sample_letters_7.png
sample_letters_8.png
sample_letters_9.png
sample_letters_10.png
sample_letters_11.png
sample_letters_12.png
sample_letters_13.png
sample_letters_14.png
sample_letters_15.png
sample_letters_16.png
sample_letters_17.png
sample_letters_18.png
sample_letters_19.png
sample_letters_20.png
sample_letters_21.png
sample_letters_22.png
sample_letters_23.png
sample_letters_24.png
sample_letters_25.png
sample_letters_26.png
sample_letters_27.png
sample_letters_28.png
sample_letters_29.png
sample_letters_30.png
sample_letters_31.png
sample_letters_32.png
sample_letters_57.png
sample_letters_58.png
sample_states_5.png
sample_states_29.png
sample_states_30.png
sample_states_37.png
state_boundary_1.png
state_boundary_2.png
state_boundary_9.png
state_boundary_33.png
state_boundary_34.png
state_boundary_35.png
state_boundary_36.png
state_boundary_37.png
state_boundary_38.png
state_boundary_39.png
state_boundary_40.png
state_boundary_41.png

Based sample_letters_0.txt

フレームごとの真音素ラベル

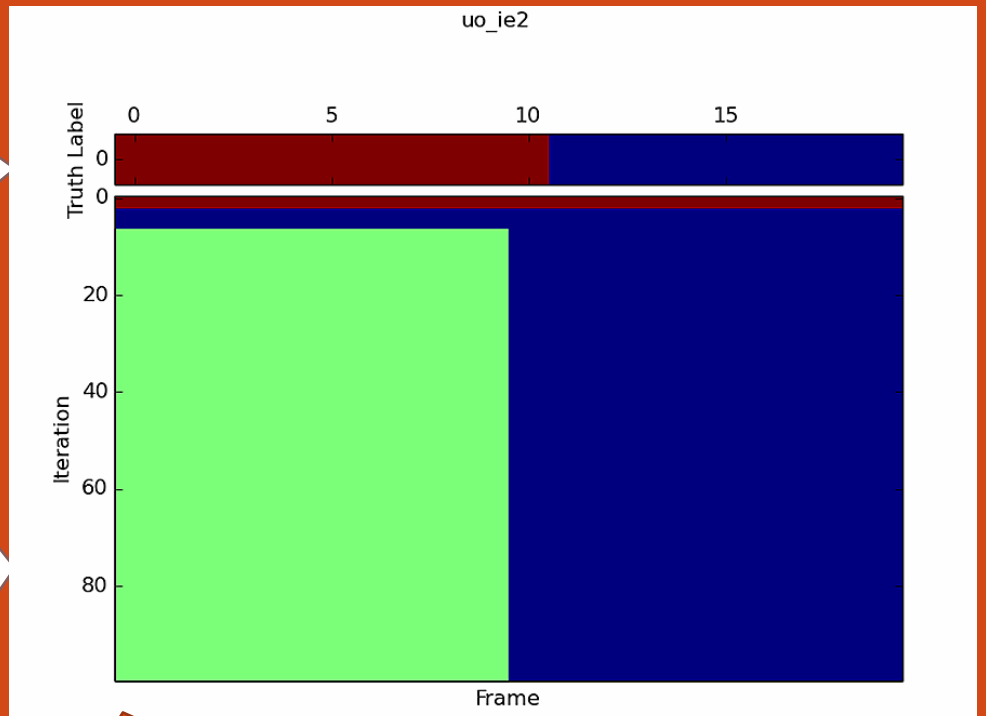
イテレーション毎におけるフレームごとの推定音素ラベル



Step5.　5 出力ファイル説明

フレームごとの真単語ラベル

イテレーション毎におけるフレームごとの推定単語ラベル



Based sample_states_57.txt

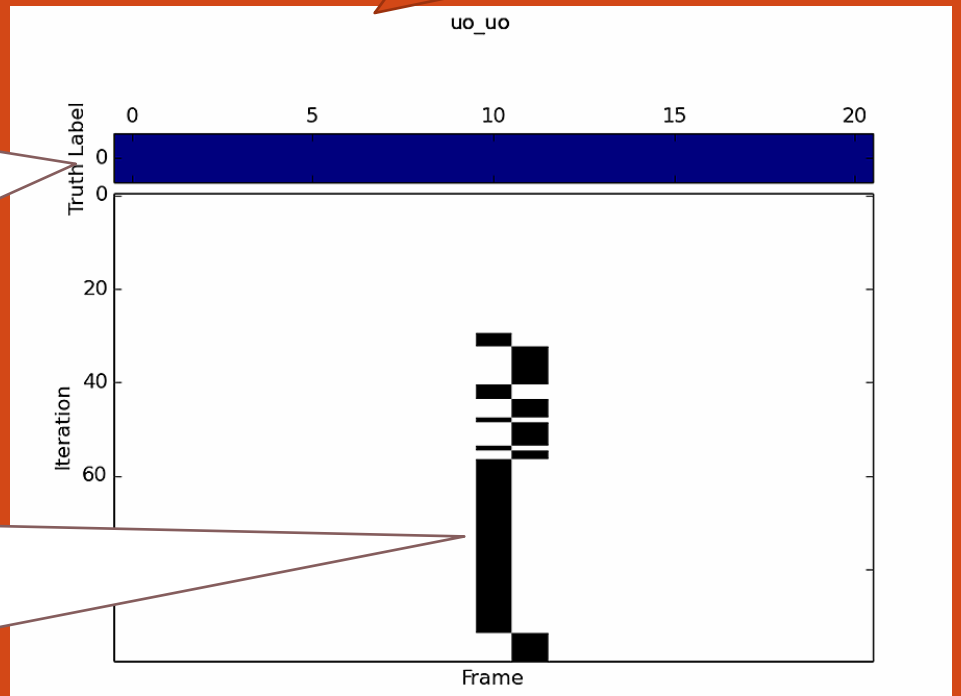
Step5.　5 出力ファイル説明

aRIs_l.txt
aRIs_s.txt
confused_matrix_l.csv
confused_matrix_s.csv
maxLk_adjusted_rand_index_l.txt
maxLk_adjusted_rand_index_s.txt
PERandWER.txt
sample_letters_0.png
sample_letters_1.png
sample_letters_24.png
sample_letters_25.png
sample_letters_26.png
sample_letters_27.png
sample_letters_56.png
sample_letters_57.png
sample_letters_58.png
sample_letters_59.png
sample_states_28.png
sample_states_29.png
sample_states_30.png
sample_states_31.png
state_boundary_0.png
state_boundary_1.png
state_boundary_2.png
state_boundary_3.png
state_boundary_4.png
state_boundary_5.png
state_boundary_6.png
state_boundary_7.png
state_boundary_8.png
state_boundary_9.png
state_boundary_32.png
state_boundary_33.png
state_boundary_34.png
state_boundary_35.png
state_boundary_36.png
state_boundary_37.png
state_boundary_38.png
state_boundary_39.png
state_boundary_40.png
state_boundary_41.png
state_boundary_42.png
state_boundary_43.png
state_boundary_44.png
state_boundary_45.png
state_boundary_46.png
state_boundary_47.png
state_boundary_48.png
state_boundary_49.png
state_boundary_50.png
state_boundary_51.png
state_boundary_52.png
state_boundary_53.png
state_boundary_54.png
state_boundary_55.png
state_boundary_56.png
state_boundary_57.png
state_boundary_58.png
state_boundary_59.png
WordList.txt

Based state_ranges_58.txt

フレームごとの真単語ラベル

イテレーション毎における単語境界が何フレーム目に存在するか



Step6 startDAA.py 説明

```
import os
if "XDG_CACHE_HOME" in os.environ: del os.environ["XDG_CACHE_HOME"]
import pyhsmm
import pyhsmm.parallel
from pyhsmm import parallel
from pyhsmm.basic.pybasicbayes.distributions import Gaussian as Gaussian
from pyhsmm.util.text import progprint_xrange
from dahsmm.hsmm import DAHSMM
from dahsmm.word import LLHSMM
from dahsmm.util import get_result_dir, save_parameters, save_fig_title, Result
from dahsmm.states import HSMMState
import numpy as np
import glob
import multiprocessing
import pickle
import time
#-----------------------------------save paras---------------------------------
SAVE_PARAMS = '''ITER_N
LETTER_N
WORD_N
DATA_N
model_hypparams
word_model_params
obs_hypparams
dur_hypparams
filename'''.split('\n')
```

- 各種インポート
- 保存するパラメータ
  - parameter.json に保存される

Step6 startDAA.py 説明

```
36  def main(result_dir = None):
37      if result_dir == None:
38          result_dir = get_result_dir(__file__)
39      os.mkdir(result_dir)
40      param_path = os.path.join(result_dir, 'parameter.json')
41      fig_title_path = os.path.join(result_dir, 'fig_title.json')
42  #initialize model params#
43      #O(T*L_max*W_max^2*d_max^3)
44      ITER_N = 100
45      LETTER_N = 7
46      WORD_N = 7
47      DATA_N = 60
48      obs_dim = 3
49      model_hyperparams = {'state_dim': WORD_N, 'alpha': 10.0, 'gamma': 10.0}
50      word_model_params = {'letter_type': LETTER_N, 'rho': 10}
51      obs_hypparams = {'mu_0': np.zeros(obs_dim), 'sigma_0': np.eye(obs_dim) ,'kappa_0': 0.01, 'nu_0': obs_dim + 5}
52      dur_hyppara s = {'alpha_0': 200.0, 'beta_0': 10.0}
53      length_dist = pyhsmm.distributions.PoissonDuration(alpha_0 = 30, beta_0 = 10, lmbda = 3)
```

結果を保存するファイルのパス&フォルダ生成

- パラメータ設定部分（一回目のpyhsmmと基本は同じ
  - dahsmm部分
    - 最大単語数
    - 潜在単語の遷移確率ハイパーパラメータ
    - 単語の文字列長分布仮定（ポアソン分布を仮定

Step6 startDAA.py 説明

```
#setting#
    obs_dists = [Gaussian(**obs_hypparams) for state in range(LETTER_N)]
    dur_dists = [pyhsmm.distributions.PoissonDuration(**dur_hypparams) for state in range(LETTER_N)]
    letter_hsmm = LLHSMM(
        init_state_concentration = 10,
        alpha = 10.0,
        gamma = 10.0,
        dur_distns = dur_dists,
        obs_distns = obs_dists
    )
    mode  = DAHSMM(
        odel_hypparams,
        etter_hsmm,
        length_dist,
        bs_dists,
        dur_dists
    )
```

pyhsmmと同じ

- hsmmオブジェクト生成
- dahsmmオブジェクト生成

Step6 startDAA.py 説明

```
#dump_object in multi engines#
    #"""
    print "-------------------------------------dump process start----------------------------
    count = multiprocessing.Value('i', 0)
    datatxt_names = glob.glob('./DATA/*.txt')
    datatxt_names.sort()
    for f in datatxt_names:
        input_mat = np.loadtxt(f)
        f = f.replace("./DATA/", "")
        f = f.replace(".txt", "")
        pr = multiprocessing.Process(target=multi_dump_object, args=(input_mat,model,f,count))
        pr.start()
    while (1):
        if count.value > 59:
            time.sleep(1)
            print "-------------------------------------dump process completed!!---------------
            break
    #"""
#add_data in one engine#
    """
    filename = []
    for f in glob.glob('./DATA/*.txt'):
        model.add_data(np.loadtxt(f))
        f = f.replace("./DATA/","")
        filename.append(f.replace(".txt",""))
    """
```

```
#-------------------------------------multi process function---
def multi_dump_object(data,model,f,count):
    print f, " dumping..."
    fp = open("TMP/"+f+".dump", 'w')
    pickle.dump(HSMMState(data,model), fp)
    fp.close()
    count.value = count.value + 1
    print f, 'dump finish-->count:',count.value
```

- DATAフォルダ上の入力データ読み込みを並列処理 multiprocessing.Process()
- 読み込みの逐次処理バージョン

データをHSMMオブジェクトに読み込みそれをTMPフォルダに保存する関数（これを並列処理）

Step6 startDAA.py 説明

```
97  #add_data in multi engines#
98      print "-----------------------------------add_data process start--------
99      filename = []
100     datadmp_names = glob.glob('./TMP/*.dump')
101     datadmp_names.sort()
102     for f in datadmp_names:
103         print f, " loading..."
104         fp = open(f)
105         f = f.replace("./TMP/", "")
106         f = f.replace(".dump", "")
107         filename.append(f)
108         obj = pickle.load(fp)
109         model.states_list.append(obj)
110         if model.parallel:
111             parallel.add_data(model.states_list[-1].data)
112         fp.close()
113     print "-----------------------------------add_data process completed!!--
```

- TMPフォルダ上の保存したオブジェクトファイル（*.dump）を読み込み，DAHSMMオブジェクトに追加

一度TMP上に保存したら，使いまわすことができる．
（前ページの部分をコメントアウトする）

```
114 #save params&charm#
115     save_fig_title(fig_title_path, SAVE_PARAMS, locals())
116     save_parameters(param_path, SAVE_PARAMS, locals())
117     obs_hypparams['sigma_0']=np.eye(obs_dim)
118     obs_hypparams['mu_0']=np.zeros(obs_dim)
```

fig_title.json（データ名，順番），parameter.json（各種パラメータ）を保存する

Step6 startDAA.py 説明

```
119  #estimation&result_write#
120      print "-------------------------------------estimation process start--------
121      result = Result(result_dir, DATA_N)
122      loglikelihood = []
123      for idx in progprint_xrange(ITER_N, perline = 10):
124          model.resample_model()
125          loglikelihood.append(result.save_loglikelihood(model))
126          result.save(model)
127      result.write_loglikelihood(loglikelihood)
128      print "-------------------------------------estimation process completed!!---
```

pyhsmmと同じ

- **推定部分**
  - startDAA_result_*上にイテレーション毎でリサルトファイルを保存，更新（sample_letters,sample_states,state_boundary）
  - **サンプリング**
  - startDAA_result_*上にイテレーション毎の尤度を保存

Step6 startDAA.py 説明

```
55  class Result(object):
56      def __init__(self, dirname, data_n):
57          self._dir = dirname
58          self.states = [[] for _ in range(data_n)]
59          self.state_ranges = [[] for _ in range(data_n)]
60          self.letters = [[] for _ in range(data_n)]
61          self.word_list = []
62
63      def dir_path(self, name):
64          return os.path.join(self._dir, name)
65
66      def save(self, model):
67          self.save_sampled_states(model)
68          self.save_word_list(model)
69          self.save_state_ranges(model)
70          self.write()
71
72      #loglikelihood_nakashima----------
73      def save_loglikelihood(self,model):
74          ll=[]
75          for k in model.states_list:
76              b=k.betal[0]
77              c=np.logaddexp.reduce(b)
78              ll.append(c)
79          reloglid = np.logaddexp.reduce(ll)
80      print reloglid
81      return reloglid
82
83      def write_loglikelihood(self,loglikelihood):
84          savetxt(self.dir_path('loglikelihood.txt'), loglikelihood, '%
                .2f')
85      #----------------------------------
```

```
87      def save_sampled_states(self, model):
88          for i, sampled in enumerate(model.states_list):
89              self.states[i].append(sampled.stateseq)
90              self.letters[i].append(sampled.letterseq)
91
92      def save_state_ranges(self, model):
93          for i, sample in enumerate(model.states_list):
94              print sample.state_ranges
95              self.state_ranges[i].append(copy.copy(sample.state_ranges
                    ))
96
97      def save_word_list(self, model):
98          self.word_list.append(copy.copy(model.word_list))
99
100     def write(self):
101         for idx, (states, letters) in enumerate(zip(self.states, self
                .letters)):
102             savetxt(self.dir_path('sample_states_%d.txt' % idx), self
                    .states[idx])
103             savetxt(self.dir_path('sample_letters_%d.txt' % idx),
                    self.letters[idx])
104
105         with open(self.dir_path('sample_word_list.txt'), 'w') as f:
106             pickle.dump(self.word_list, f)
107
108         for idx, ranges in enumerate(self.state_ranges):
109             with open(self.dir_path('state_ranges_%d.txt' % idx), 'w'
                    ) as f:
110                 pickle.dump(ranges, f)
111
```

Result class 部分 （dahsmm/util.py）

Step7 summary.py 説明

```
26  #-------------------------------------main function-------------------------------------#
27  def main():
28  #result_file make#
29      parser = argparse.ArgumentParser()
30      parser.add_argument('directory')
31      #opts = parser.parse_args()
32      figs_dir = 'summary_figs'
33      os.path.exists(figs_dir) or os.mkdir(figs_dir)
34      summary = Summary()
35  #evaluation_result save#
36      with pushd(figs_dir):
37          #gen confused matrix
38          summary.letter_confused_matrix()
39          summary.state_confused_matrix()
40          #gen PER and WER
41          summary.culPER()
42          summary.culWER()
43          #gen adjusted rand index
44          summary.a_rand_index(summary.sample_letters,summary.input_data,'l')
45          summary.a_rand_index(summary.sample_states,summary.input_data2,'s')
46          #gen word list
47          with open('WordList.txt',"w") as f:
48              for num, key in enumerate(summary.word_list):
49                  f.write("iter%d:: " % num)
50                  for num2, key2 in enumerate(key):
51                      f.write("%d:" % num2 + str(key2) + " ")
52                  f.write("\n")
53  #multi plot sample states and letters#
54          print "-------------------------------------plot process start-----------------------
55          count = multiprocessing.Value('i', 0)
56          for idx in range(summary.data_size):
57              pr = multiprocessing.Process(target=multi_plot_object, args=(summary,idx,count))
58              pr.start()
59              time.sleep(0.1) #charm...!!!(koreganaito roop karanukenai)
60          while (1):
61              if count.value > 59:
62                  time.sleep(1)
63                  print "-------------------------------------plot process completed!!----------
64                  break
65  #====================summary(main process?) class====================#
```

summary_figs フォルダ作成，Summaryクラス生成

summary_figs フォルダ移動

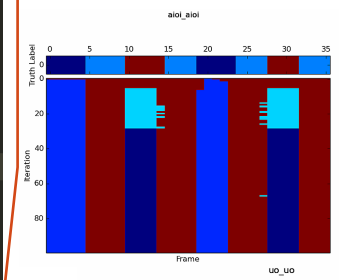
中島用

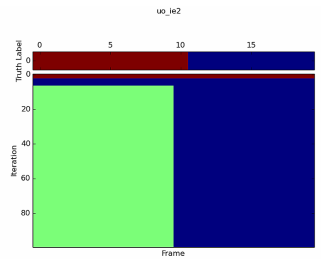
推定単語リスト作成 WordList.txt

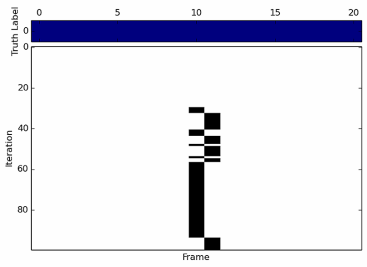
グラフ作成処理を並列処理

multiprocessing.Process()

```
14  #-------------------------------------multi process function---------
15  def multi_plot_object(summary,idx,count):
16      print summary.fig_title[idx], " plotting..."
17      summary.plot_states(idx)
18      plt.savefig('sample_states_%d.png' % idx)
19      summary.plot_state_boundaries(idx)
20      plt.savefig('state_boundary_%d.png' % idx)
21      summary.plot_letters(idx)
22      plt.savefig('sample_letters_%d.png' % idx)
23      plt.clf()
24      count.value = count.value + 1
25      print summary.fig_title[idx], 'plot finish-->count:',count.value
```







Step7 summary.py 説明（Summary class）

```
65  #====================summary(main process?) class====================#
66  class Summary(object):
67  #-------------------------------------init paras-------------------------------------#
68      def __init__(self, dirpath = '.'):
69          with open('parameter.json') as f:
70              params = self.params = json.load(f)
71          with open('fig_title.json') as f2:
72              fig_title = self.fig_title = json.load(f2)
73          with open('sample_word_list.txt') as f3:
74              self.word_list = pickle.load(f3)
75          self.data_size = params['DATA_N']
76          self.input_data = [np.loadtxt("../LABEL/"+ i + ".lab") for i in fig_title]
77          self.input_data2 = [np.loadtxt("../LABEL/"+ i + ".lab2") for i in fig_title]
78          self.sample_states = [np.loadtxt('sample_states_%d.txt' % i)for i in range(params['DATA_N'])]
79          self.sample_letters = [np.loadtxt('sample_letters_%d.txt' % i)for i in range(params['DATA_N'])]
80          self.state_ranges = []
81          for i in range(params['DATA_N']):
82              with open('state_ranges_%d.txt' % i) as f:
83                  self.state_ranges.append(pickle.load(f))
84          llist = np.loadtxt("loglikelihood.txt").tolist()
85          self.maxlikelihood = (max(llist), llist.index(max(llist)))
86          #manipulation part
87          self.l_label_dic={}
88          self.s_label_dic={}
89          #manipulation part end
```

中島用

各種ファイル読み込み

parameter.json：**パラメータ）**

fig_title.json：**データ名，順番**

LABEL/*.lab

（2）：真ラベル

sample_letters_*.txt

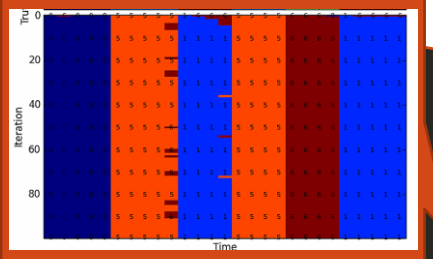
sample_states_*.txt

state_ranges_*.txt

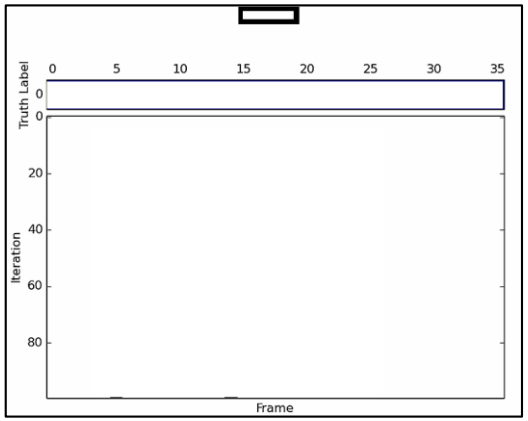
loglikelihood.txt：**尤度（イテレーションの中で最大だった尤度判定用）**

```python
#-------------------------------------write result graph-------------------------------------#
#base_graph function#
    def _plot_discreate_sequence(self, true_data, title, sample_data, label = u'', plotopts = {}):
        ax = plt.subplot2grid((10, 1), (1, 0))
        plt.sca(ax)
        ax.matshow([true_data], aspect = 'auto')
        plt.ylabel('Truth Label')
        #label matrix
        ax = plt.subplot2grid((10, 1), (2, 0), rowspan = 8)
        plt.suptitle(title)
        plt.sca(ax)
        ax.matshow(sample_data, aspect = 'auto', **plotopts)
        #write per 10 iterations(max_likelihood) label
        """
        for i in range(label.shape[0]):
            for j in range(label.shape[1]):
                if i%10==0 or i==99 or i==self.maxlikelihood[1]:
                    if i==self.maxlikelihood[1]:
                        ax.text(j, i+1.5, int(label[i][j]), ha='center', va='bottom', color = 'red', fontsize=8)
                    else:
                        ax.text(j, i+1.5, int(label[i][j]), ha='center', va='bottom', color = 'black', fontsize=8)
                ax.text(j, i+1.5, int(label[i][j]), ha='center', va='bottom', color = 'black', fontsize=8)
        """
        #write x&y label
        plt.xlabel('Frame')
        plt.ylabel('Iteration')
        plt.xticks(())
```

グラフにラベル印字



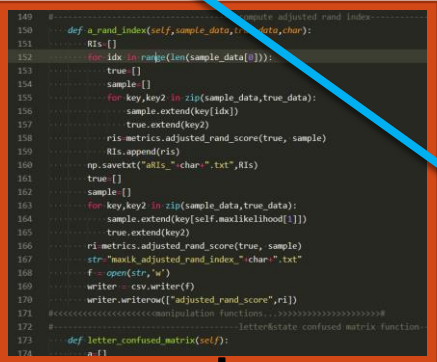
グラフの共通デザイン設定
（軸ラベルや図位置など）



各種グラフ作成関数

- sample_letters_*.png
- sample_states_*.png
- state_boundary_*.png

Summaryクラス内の以降の関数は中島用なので必要ないです（消しても良い）



```python
#plot letter_result graph#
    def plot_letters(self, idx):
        self._plot_discreate_sequence(
                self.input_data[idx],
                self.fig_title[idx],
                self.sample_letters[idx],
                label=self.sample_letters[idx]
        )
#plot state_result graph#
    def plot_states(self, idx):
        self._plot_discreate_sequence(
                self.input_data2[idx],
                self.fig_title[idx],
                self.sample_states[idx],
                label=self.sample_states[idx]
        )
#plot boundary graph#
    def _plot_label_boundary(self, true_data, title, sample_data, label = u''):
        boundaries = [[stop for state, (start, stop) in r] for r in sample_data]
        size = boundaries[0][-1]
        data = np.zeros((len(sample_data), size))
        for i, b in enumerate(boundaries):
            for x in b[:-1]:
                data[i, x] = 1.0
        self._plot_discreate_sequence(true_data, title, data, label, plotopts = {'cmap': 'Greys'})
    def plot_state_boundaries(self, idx):
        self._plot_label_boundary(
                self.input_data2[idx],
                self.fig_title[idx],
                self.state_ranges[idx],
                label=self.sample_states[idx]
        )
```