



# Eye Movement analysis with HMMs: EMHMM Tutorial

**Antoni B. Chan**

Dept. of Computer Science,  
City University of Hong Kong

**Janet H. Hsiao**

Dept. of Psychology,  
The University of Hong Kong



# EMHMM Toolbox Tutorial

- **Tutorial on how to use the EMHMM toolbox**
  - Obtaining and installing the toolbox
  - Setup in MATLAB
  - Reading in data
  - Estimating individual HMMs
  - Viewing individual HMMs
  - Clustering individuals' HMMs to form group HMMs
  - Viewing group HMMs
  - Analysis of group HMMs
  - Holistic/Analytic Models and H-A Scale

# Obtaining EMHMM Toolbox



- Download from website:
  - <http://visal.cs.cityu.edu.hk/research/emhmm/>

## Downloads

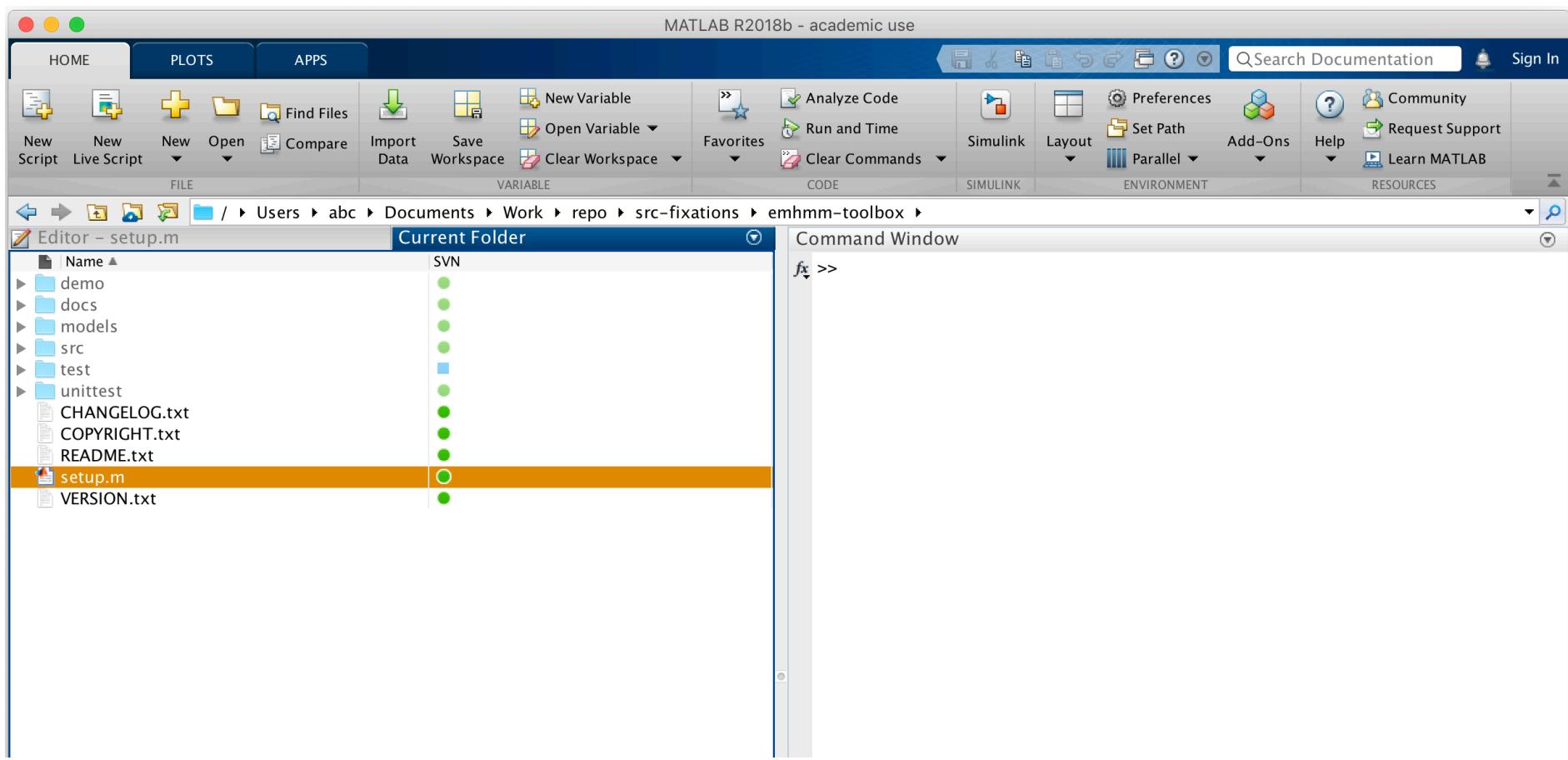
This is the MATLAB toolbox for analyzing eye movement data using hidden Markov models. It includes code for learning HMMs for individuals, as well as clustering individuals' HMMs into groups.

- **Files:** [emhmm-toolbox v0.75](#) (beta version)
- **Version History:** [Change Log](#)
- *If you use this toolbox please cite:*  
[Understanding eye movements in face recognition using hidden Markov models.](#)  
Tim Chuk, Antoni B. Chan, and Janet H. Hsiao,  
*Journal of Vision*, 14(11):8, Sep 2014.

- Installation:
  - Extract the contents of the emhmm-toolbox.zip file into your desired directory.

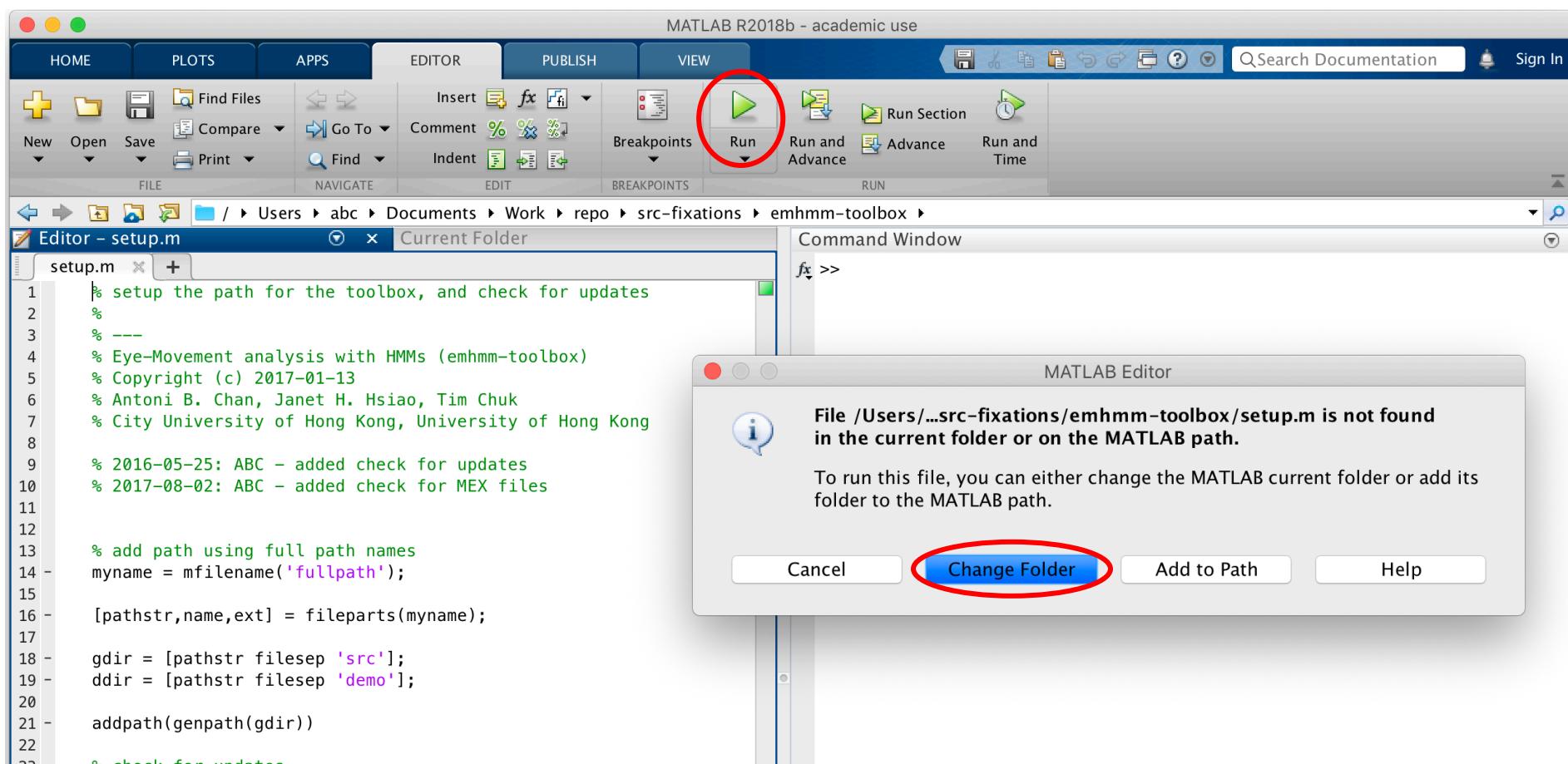
# Setup in MATLAB (1)

- First, need to setup the EMHMM toolbox in MATLAB.
  - In MATLAB, navigate to the emhmm-toolbox directory.
  - Double-click on “setup.m” to open it.



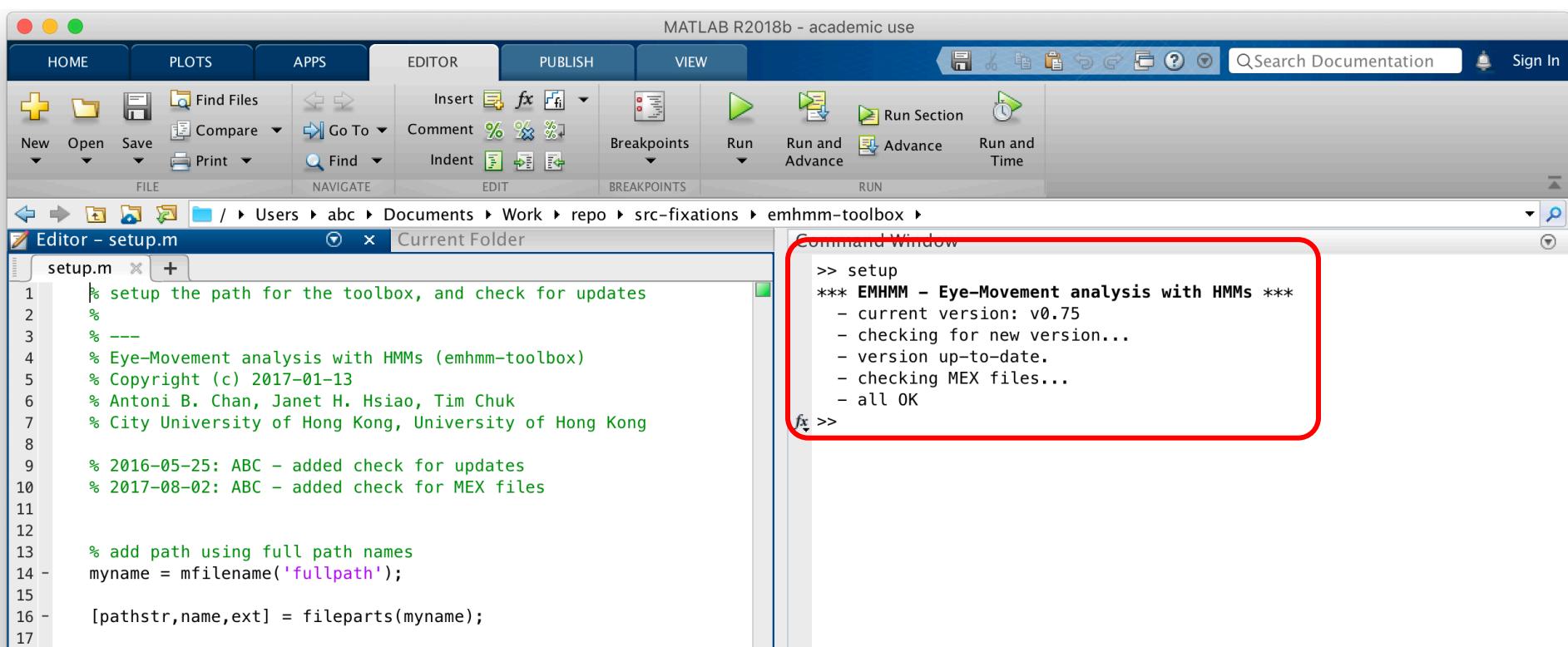
# Setup in MATLAB (2)

- Click the “Run” button to run the script.
  - If prompted, click “Change Folder” to change to the script’s directory.



# Setup in MATLAB (3)

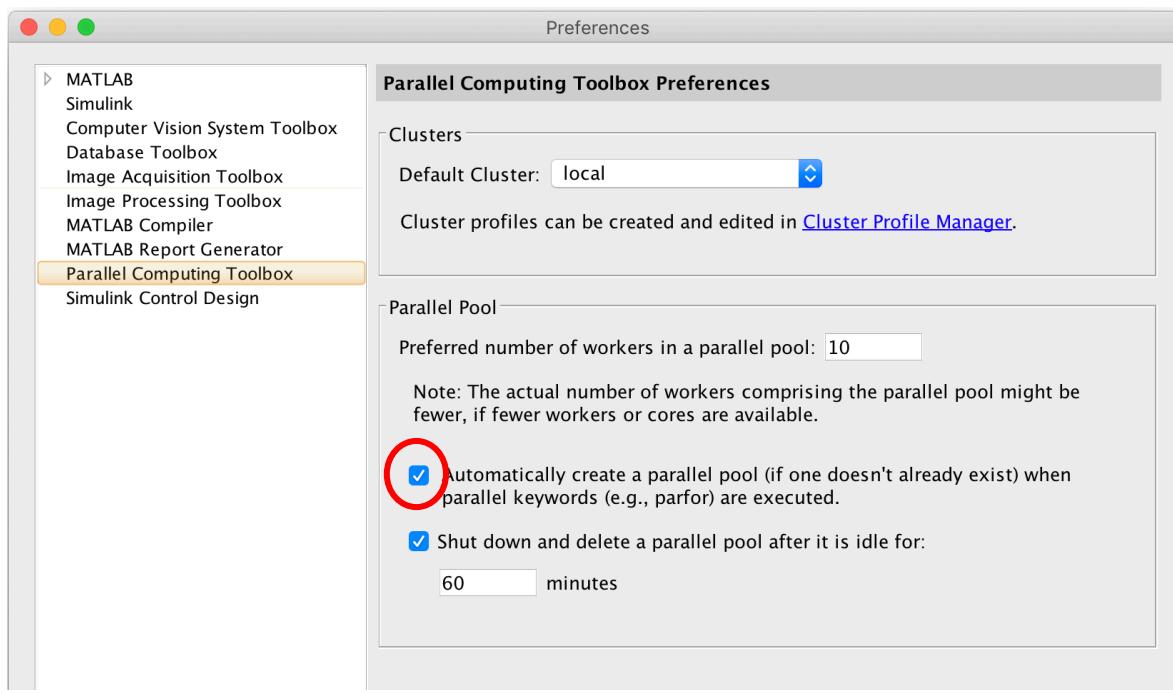
- The setup script will:
  - add the toolbox directories to the MATLAB path.
  - check for MEX files (compiles if necessary).
    - Includes MEX files for: macOS (10.13), Windows (7), Linux (Ubuntu).
    - MEX files make the algorithms 10-20 times faster.



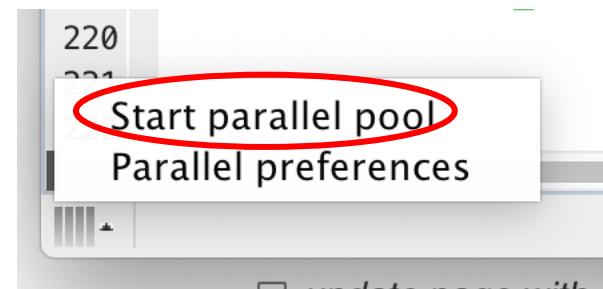
# MATLAB toolbox dependencies

- MATLAB version: tested with **2014b** and **2018b**.
- *Statistics toolbox* (required)
- *Parallel computing toolbox* (optional)
  - Used for speeding up computations on multi-core CPUs.

To enable automatically, use Preferences:

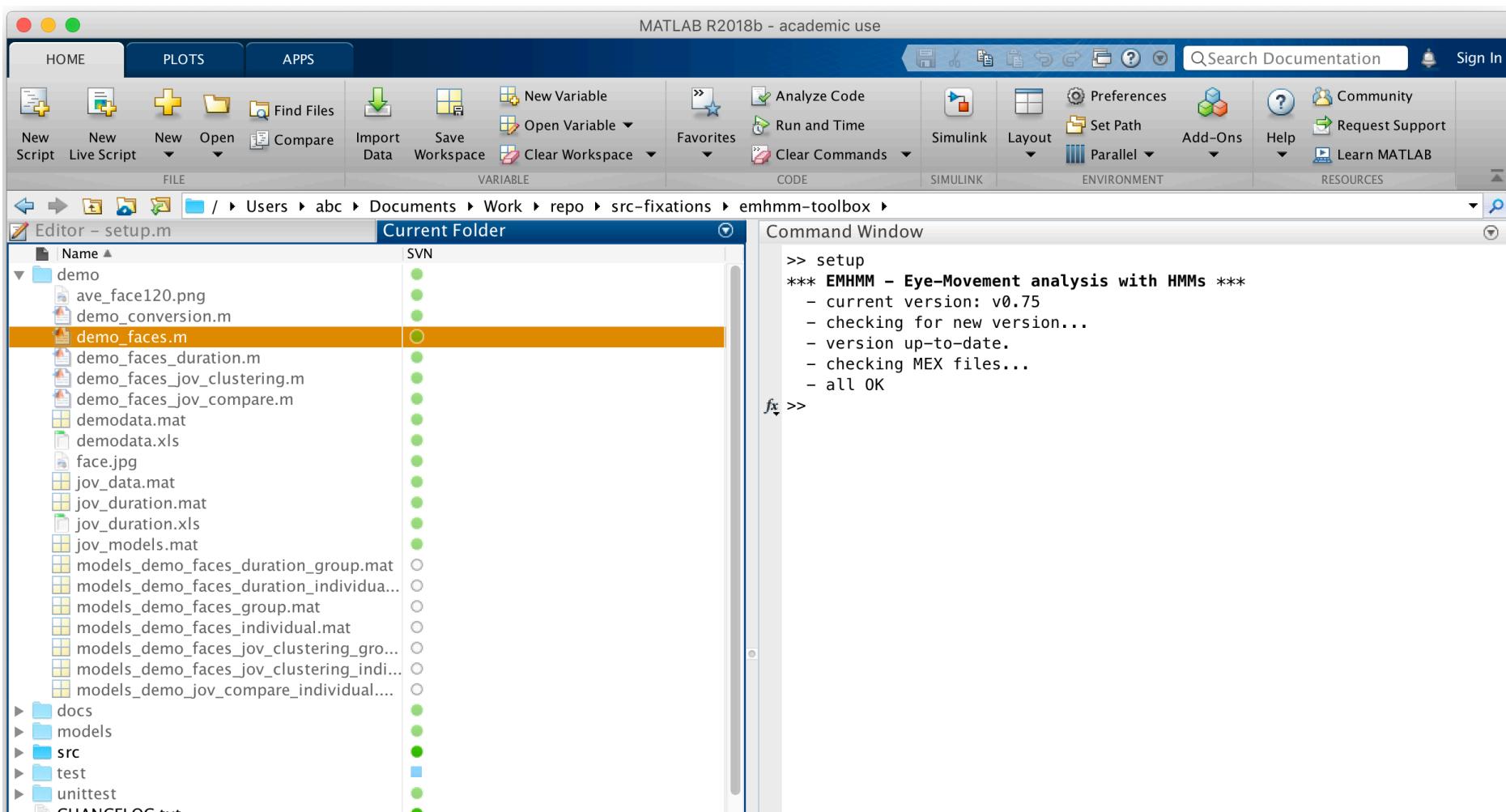


To enable manually, click small icon on bottom-left of the MATLAB window:



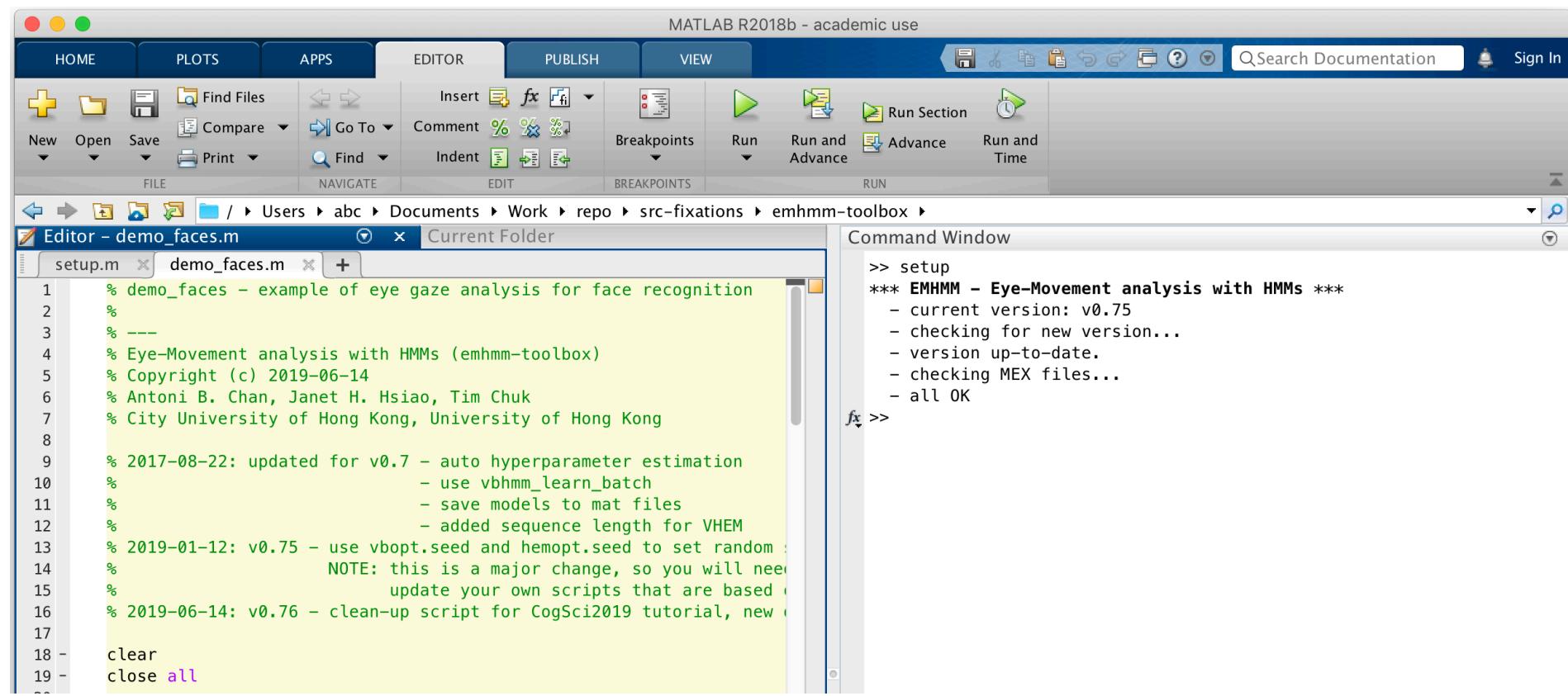
# Demo Script

- In the “demo” folder, the script “`demo_faces.m`” shows most of the functionality of the toolbox.



# Run Demo Script

- To run the `demo_faces` script:
  - Double-click to open the script.
  - Click “Run”, and select “Change Folder” when prompted.
  - Now we will go through the major parts of the script.



The screenshot shows the MATLAB R2018b interface. The top menu bar includes HOME, PLOTS, APPS, EDITOR (selected), PUBLISH, and VIEW. The toolbar below has buttons for New, Open, Save, Compare, Go To, Find, Insert, Comment, Indent, Breakpoints, Run, Run and Advance, and Run and Time. The current folder path is displayed in the toolbar: /Users/abc/Documents/Work/repo/src-fixations/emhmm-toolbox. The Editor window on the left shows the file `demo_faces.m` with its code. The Command Window on the right shows the output of running the `setup` command, which performs several checks and concludes with "all OK".

```
>> setup
*** EMHMM - Eye-Movement analysis with HMMs ***
- current version: v0.75
- checking for new version...
- version up-to-date.
- checking MEX files...
- all OK
fx >>
```

# Setup Data Files

- File with the fixation data: demodata.xls
- File with the image stimuli: ave\_face120.png
  - the average image used for visualization
- Files for saving results for saving results (.mat)

```
17  
18 -     clear  
19 -     close all  
20  
21 %% Data files %%%%%%%%%%%%%%  
22 % names of files used  
23 - xlsname = 'demodata.xls';    % Excel File with fixation data  
24 - faceimg = 'ave_face120.png'; % average face image  
25 |  
26 % names of files for saving results  
27 - matfile_individual = 'models_demo_faces_individual.mat';  
28 - matfile_group      = 'models_demo_faces_group.mat';  
29  
30  
31 % pause after each step in the analysis  
32 - do_pause = 1;  
33 - %do_pause = 0; % uncomment this to skip pausing
```

# Fixation Data Format

- Fixation data is in an Excel spreadsheet (`demodata.xls`).
  - Contains 4 columns:
    - *SubjectID*: subject ID (int or string)
    - *TrialID*: trial ID (int or string)
    - *FixX*: fixation X coordinate
    - *FixY*: fixation Y coordinate
  - Fixation data is automatically separated according to subject ID and trial ID.
  - In each trial, assumes fixations occur in order.
- Assumes that fixation locations have been aligned to the image.

	A	B	C	D
1	SubjectID	TrialID	FixX	FixY
2	1	1	182.16	209.52
3	1	1	201.68	192.13
4	1	1	134.48	186.12
5	1	2	166.08	192.21
6	1	2	188.96	200.38
7	1	2	122.88	208
8	1	3	147.84	275
9	1	3	172.72	212.24
10	1	3	109.36	217.45
11	1	4	168.96	202.39
12	1	4	180.56	223.39
13	1	4	109.68	226.91
14	1	5	170.88	231.32
15	1	5	120.16	234.53
16	1	5	180.88	239.58
17	1	6	172.32	204.71
18	1	6	98.16	206.87
19	1	6	180.64	203.03
20	1	7	161.76	242.7
21	1	7	173.92	222.98
22	1	7	137.12	216.97
23	1	8	169.36	222.02
24	1	8	178.88	222.9
25	1	8	128.48	230.68
26	1	9	182.56	205.51
27	1	9	190.08	213.29
28	1	9	142.64	217.53
29	1	10	171.44	210.96
30	1	10	123.28	217.53
31	1	10	183.44	202.31
32	1	11	112.8	200.3
33	1	11	145.12	235.97
34	1	11	157.84	110.00

# Reading in Data

MATLAB R2018b - academic use

HOME PLOTS APPS EDITOR PUBLISH VIEW

Editor - demo\_faces.m Current Folder

setup.m demo\_faces.m +

```
20 % Data files %%%%%%
21 % names of files used
22 xlsname = 'demodata.xls'; % Excel File with fixation data
23 faceimg = 'ave_face120.png'; % average face image
24
25 % names of files for saving results
26 matfile_individual = 'models_demo_faces_individual.mat';
27 matfile_group = 'models_demo_faces_group.mat';
28
29
30
31 % pause after each step in the analysis
32 do_pause = 1;
33 %do_pause = 0; % uncomment this to skip pausing
34
35
36 %% Load data from xls %%%%%%
37 % see the xls file for the format
38 [data, SubjNames, TrialNames] = read_xls_fixations(xlsname);
39
40 % the data is read and separated by subject and trial, and stored in a cell
41 % data{i} = i-th subject
42 % data{i}{j} = ... j-th trial
43 % data{i}{j}(t,:) = ... [x y] location of t-th fixation
44
45 % the same data is stored in a mat file.
46 %load demodata.mat
47
48 % the number of subjects
49 N = length(data);
50
51 % load image
52 img0 = imread(faceimg);
```

Command Window

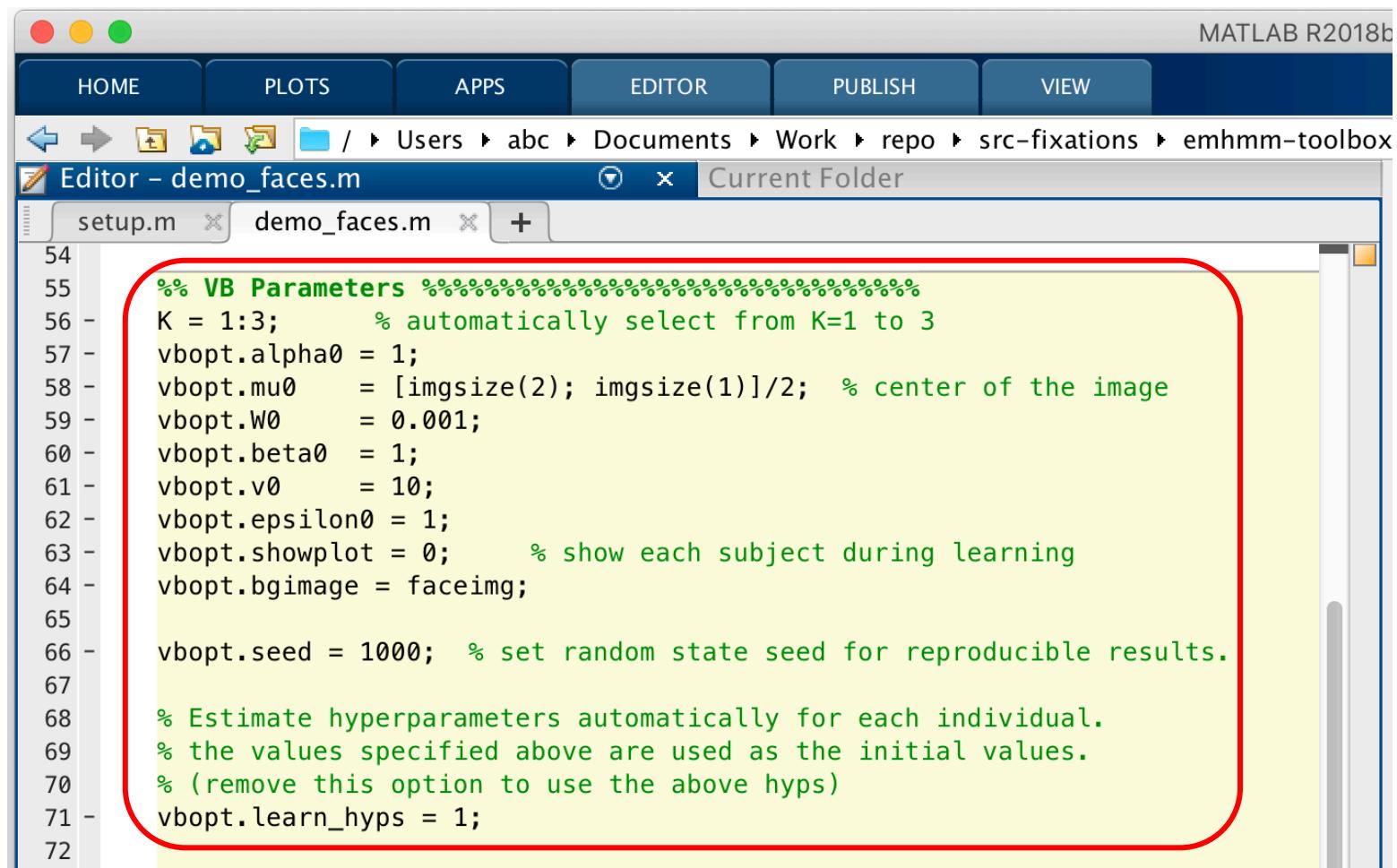
```
Reading demodata.xls
- found SubjectID in column 1
- found TrialID in column 2
- found FixX in column 3
- found FixY in column 4
- found 10 subjects:
  1 2 3 4 5 6 7 8 9 10
    * subject 1 had 40 trials
    * subject 2 had 40 trials
    * subject 3 had 40 trials
    * subject 4 had 40 trials
    * subject 5 had 40 trials
    * subject 6 had 40 trials
    * subject 7 had 40 trials
    * subject 8 had 40 trials
    * subject 9 had 40 trials
    * subject 10 had 39 trials
    === running Subject 1 ===
    === running Subject 2 ===
    === running Subject 3 ===
    === running Subject 4 ===
    === running Subject 5 ===
    === running Subject 6 ===
    [Subject 1] -- vbhmm K=1: (seed=1000).
    [Subject 1] (K=1) optimizing trial 1: -1
    [Subject 1] (K=1) best run=1; LL=-1104.6
    [Subject 1] (K=1) {alpha0=1; epsilon0=1
    [Subject 2] -- vbhmm K=1: (seed=1000).
    [Subject 2] (K=1) optimizing trial 1: -1
    [Subject 2] (K=1) best run=1; LL=-993.39
    [Subject 2] (K=1) {alpha0=1; epsilon0=1
    [Subject 3] -- vbhmm K=1: (seed=1000).
    [Subject 3] (K=1) optimizing trial 1: -9
```

# Estimating individual HMMs

- An individual HMM is estimated from each subject's fixation data using the variational Bayesian algorithm.
- Automatically selects the model hyperparameters by maximizing the log-likelihood of the data.
  - Best of 50 trials with random initializations.
- Automatically selects the number of ROIs,  $K$ .
  - The model with highest log-likelihood over all  $K$  is selected.

# Setting up Options

- K contains an array of number of ROIs to try.
- vbopt is a structure with various options.



MATLAB R2018b

HOME PLOTS APPS EDITOR PUBLISH VIEW

/ Users abc Documents Work repo src-fixations emhmm-toolbox

Editor - demo\_faces.m Current Folder

setup.m demo\_faces.m +

```
54
55 %% VB Parameters %%%%%%
56 - K = 1:3;      % automatically select from K=1 to 3
57 - vbopt.alpha0 = 1;
58 - vbopt.mu0    = [imgsize(2); imgsize(1)]/2; % center of the image
59 - vbopt.W0     = 0.001;
60 - vbopt.beta0 = 1;
61 - vbopt.v0    = 10;
62 - vbopt.epsilon0 = 1;
63 - vbopt.showplot = 0;    % show each subject during learning
64 - vbopt.bgimage = faceimg;
65
66 - vbopt.seed = 1000; % set random state seed for reproducible results.
67
68 % Estimate hyperparameters automatically for each individual.
69 % the values specified above are used as the initial values.
70 % (remove this option to use the above hypers)
71 - vbopt.learn_hyps = 1;
```

# Estimating Individual HMMs

- Using data from 10 subjects, learn 10 HMMs

The screenshot shows a MATLAB R2018b interface. The Editor window displays a script named `demo_faces.m` containing MATLAB code for estimating HMMs. The Command Window shows the execution of this code, outputting logs for 10 subjects. A red box highlights the command `[hmms, Ls] = vhmm_learn_batch(data, K, vbopt);` in the Editor, which is annotated with a callout "Optimizing hyperparameters". Another blue box highlights the parameter `K` in the same command, with a callout "Trying different values of K".

```
75 %vbopt.learn_hyps_batch = 1;
76
77 %% Learn Subjects' HMMs %%%%%%%%%%%%%%
78 %
79 % estimate HMMs for each individual
80 [hmms, Ls] = vhmm_learn_batch(data, K, vbopt);
81
82 % show subject 4
83 mys = 4;
84 vhmm_plot(hmms{mys}, data{mys}, faceimg);
85
86 % compact plot
87 figure, vhmm_plot_compact(hmms{mys}, faceimg);
88
89 % show fixations and compact plot
90 figure
91 subplot(2,1,1)
92 plot_fixations(data{mys}, faceimg, [], 's');
93 subplot(2,1,2)
94 vhmm_plot_compact(hmms{mys}, faceimg, 'r', data{mys});
95
96 pause_msg(do_pause);
97
98 % plot each subject
99 figure
100 for i=1:N
101 subplot(2,5,i)
102 vhmm_plot_compact(hmms{i}, faceimg);
103 title(sprintf('SubjectID=%s (index=%d)', SubjNames{i}, i));
104 end
105
106 % save to individual models to disk
107 fprintf('saving individual models to %s\n', matfile_individual);
108 save(matfile_individual, 'hmms', 'faceimg', 'SubjNames', 'vbopt');
109 % later on you can use the following command to read back the models:
110 % load models_demo_faces_individual.mat
111
112 pause_msg(do_pause);
113
114 %% Run HEM clustering (1 group) %%%%%%%%%%%%%%
115 % summarizing all subjects with one HMM
```

MATLAB R2018b - academic use

Editor - demo\_faces.m

Command Window

```
* Subject 10 had 33 trials
*** running Subject 1 ***
*** running Subject 2 ***
*** running Subject 3 ***
*** running Subject 4 ***
*** running Subject 5 ***
*** running Subject 6 ***
[Subject 1] -- vhmm K=1: (seed=1000).
[Subject 1] (K=1) optimizing trial 1: -1122.78.....
[Subject 1] (K=1) best run=1; LL=-1104.64
[Subject 1] (K=1) {alpha0=1; epsilon0=1; v0=830.3; beta0=Inf; W0=1.658e-06; mu0=[1
[Subject 2] -- vhmm K=1: (seed=1000).
[Subject 2] (K=1) optimizing trial 1: -1005.12.....
[Subject 2] (K=1) best run=1; LL=-993.392
[Subject 2] (K=1) {alpha0=1; epsilon0=1; v0=987.8; beta0=Inf; W0=3.339e-06; mu0=[1
[Subject 3] -- vhmm K=1: (seed=1000).
[Subject 3] (K=1) optimizing trial 1: -916.899.....LL=-903.895
[Subject 3] (K=1) best run=1; LL=-903.895
[Subject 3] (K=1) {alpha0=1; epsilon0=1; v0=17.37; beta0=1.058e+15; W0=0.00011; mu
[Subject 4] -- vhmm K=1: (seed=1000).
[Subject 4] (K=1) optimizing trial 1: -1010.87.....
[Subject 4] (K=1) best run=1; LL=-996.495
[Subject 4] (K=1) {alpha0=1; epsilon0=1; v0=1184; beta0=5.788e+221; W0=2e-06; mu0=
[Subject 5] -- vhmm K=1: (seed=1000).
[Subject 5] (K=1) optimizing trial 1: -953.233.....LL=
[Subject 5] (K=1) best run=1; LL=-941.716
[Subject 5] (K=1) {alpha0=1; epsilon0=1; v0=147.7; beta0=6.402e+99; W0=2.1e-05; mu
[Subject 6] -- vhmm K=1: (seed=1000).
[Subject 6] (K=1) optimizing trial 1: -820.548.....LL=-812.185
[Subject 6] (K=1) best run=1; LL=-812.185
[Subject 6] (K=1) {alpha0=1; epsilon0=1; v0=5.537; beta0=8.425e+14; W0=0.0007937;
[Subject 1] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 1] (K=2) optimizing trial 1: -1080.29.....LL=-1073.53
[Subject 2] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 3] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 4] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 4] (K=2) optimizi.....-975.826.....LL=-969.663
[Subject 5] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 6] -- vhmm K=2: (seed=1000).....10.....20.....30.....
[Subject 1] (K=2) optimizing trial 5: -1087.12.....LL=-1078.19
[Subject 2] (K=2) optimizing trial 1: -968.344.....LL=-957.532
[Subject 3] (K=2) optimizing trial 2: -972.822.....LL=-972.826
```

Optimizing hyperparameters

Trying different values of K

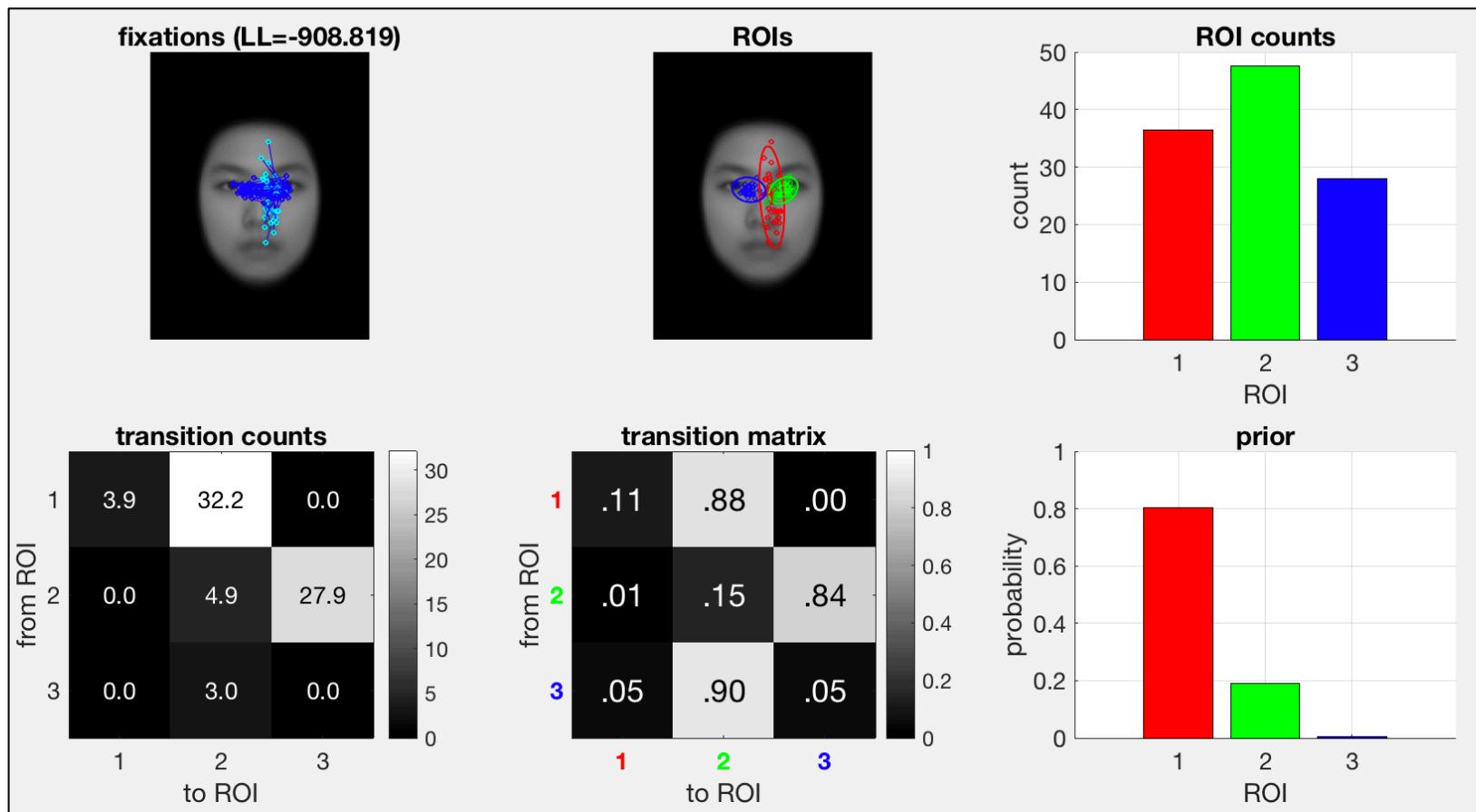
# Visualizing an HMM

Figure 1

- Look at the 4<sup>th</sup> subject:

- Output Figure

```
76  
77  
78  
79  
80 -  
81  
82 % estimate HMMs for each individual  
83 [hmms, Ls] = vhmm_learn_batch(data, K, vbopt);  
84 -  
85  
86  
87 % show subject 4  
88 mys = 4;  
89 vhmm_plot(hmms{mys}, data{mys}, faceimg);  
90  
91
```

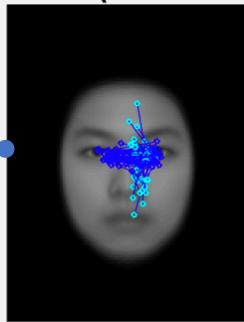


# Visualizing an HMM

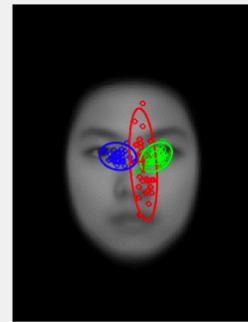
Figure 1

Fixation sequences: cyan “o” is the first fixation.

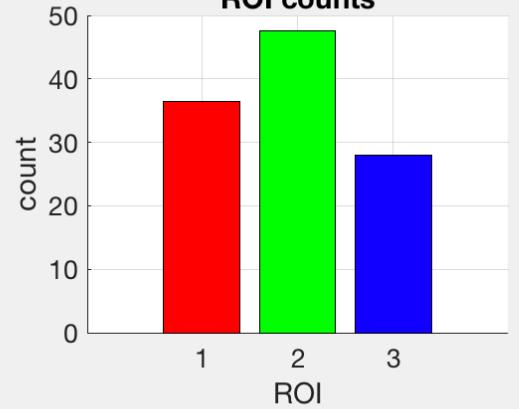
fixations (LL=-908.819)



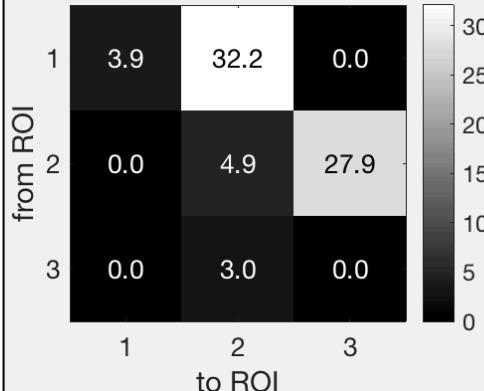
ROIs



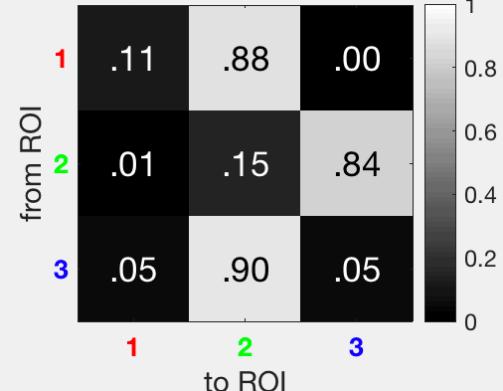
ROI counts



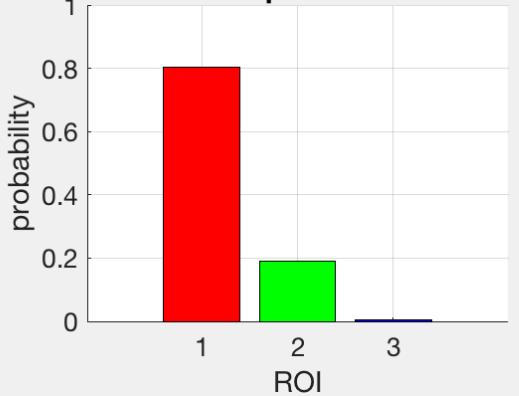
transition counts



transition matrix



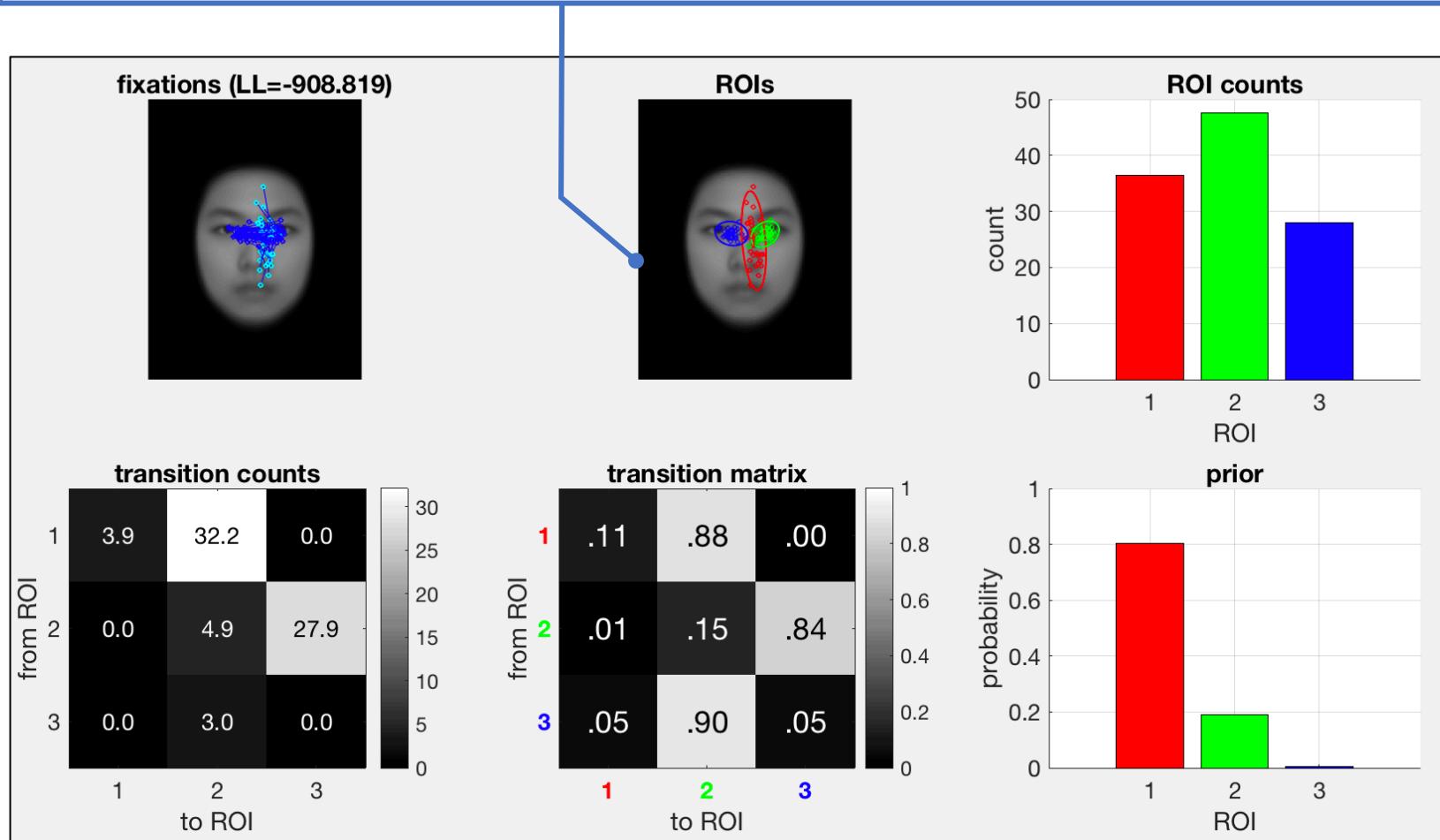
prior



# Visualizing an HMM

Figure 1

**ROIs:** ellipses represent 2 standard deviation contours from the mean of the 2D Gaussian (95% of the probability density). Color indicates the fixations assigned to an ROI.

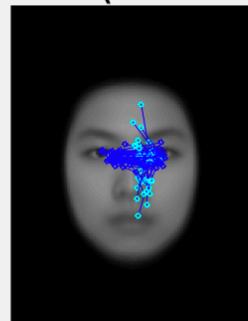


# Visualizing an HMM

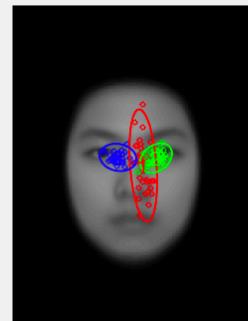
Figure 1

**ROI counts:** total number of fixations in each ROI.

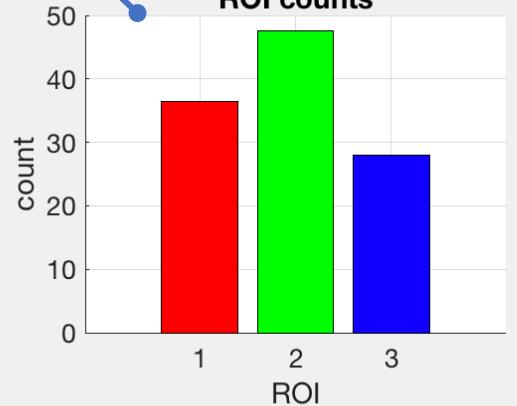
fixations (LL=-908.819)



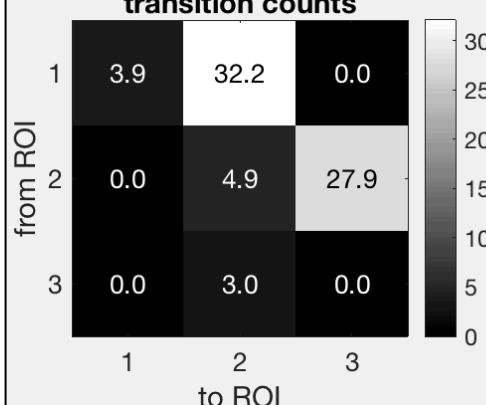
ROIs



ROI counts



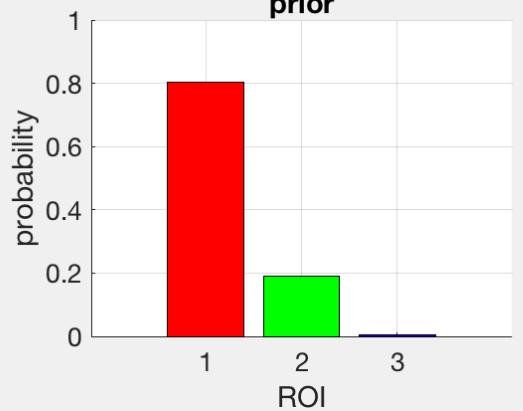
transition counts



transition matrix



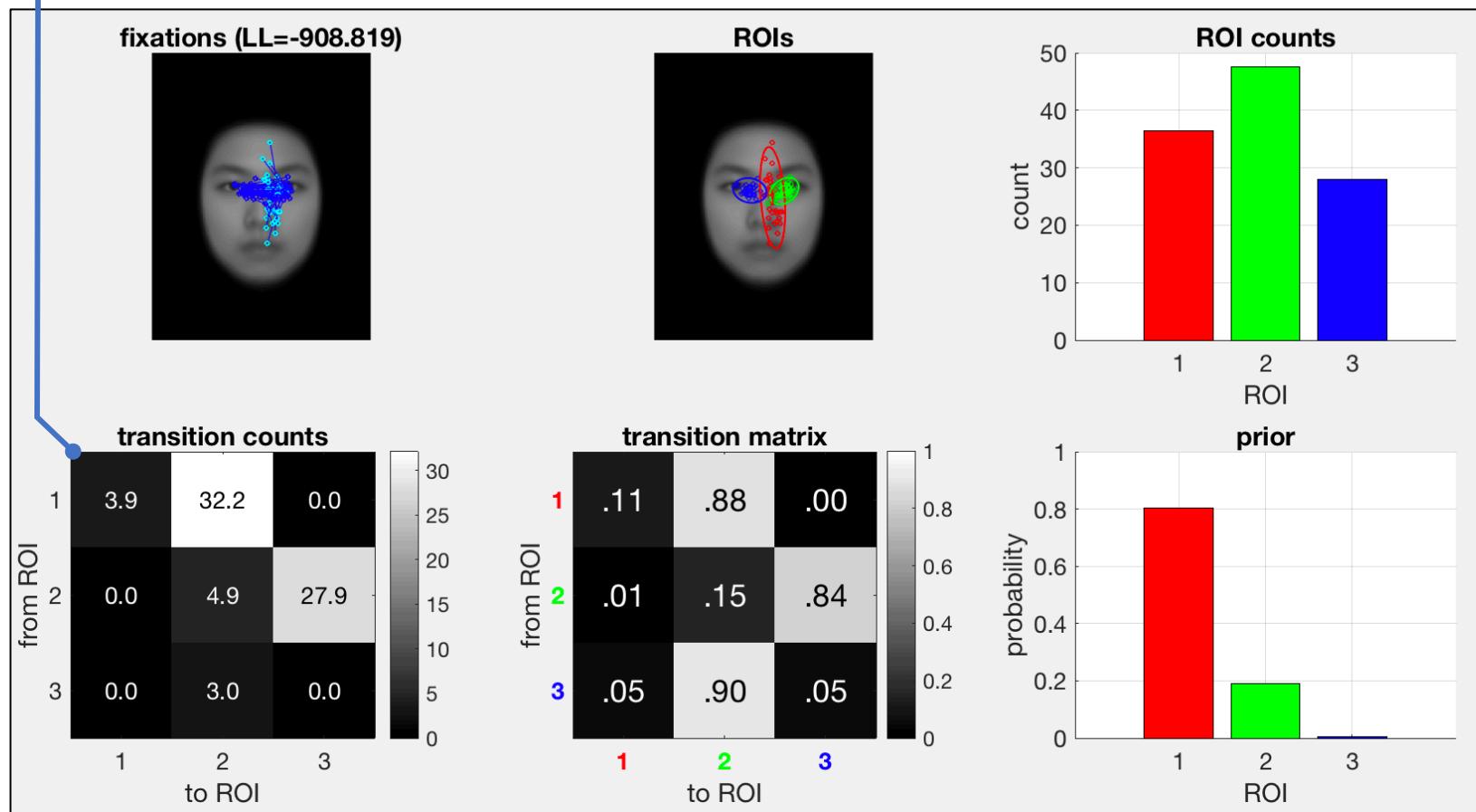
prior



# Visualizing an HMM

Figure 1

**Transition counts:** total number of transitions from one ROI to another ROI.

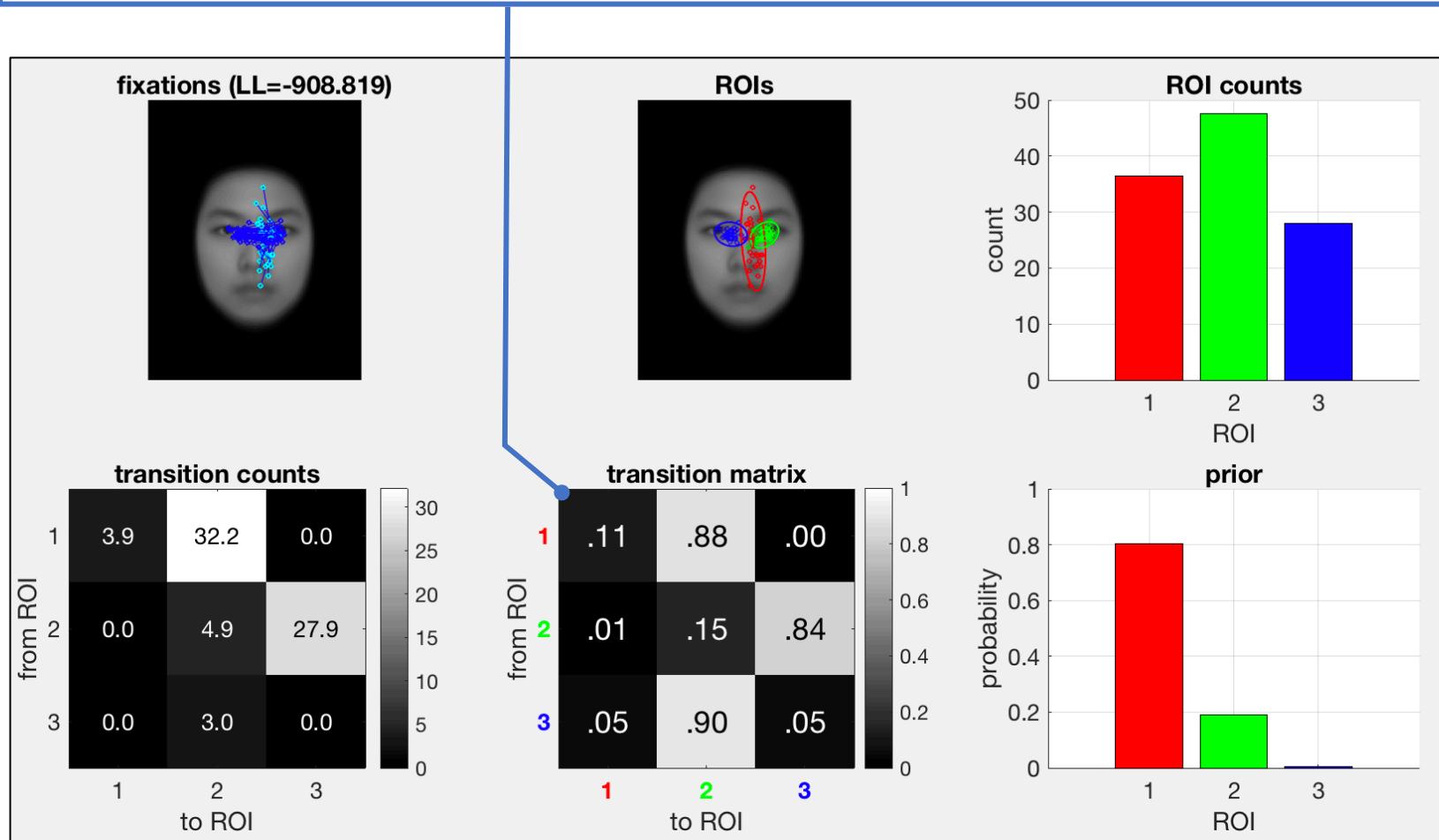


# Visualizing an HMM

Figure 1

**Transition matrix:** given a particular ROI, probability of moving to another ROI.

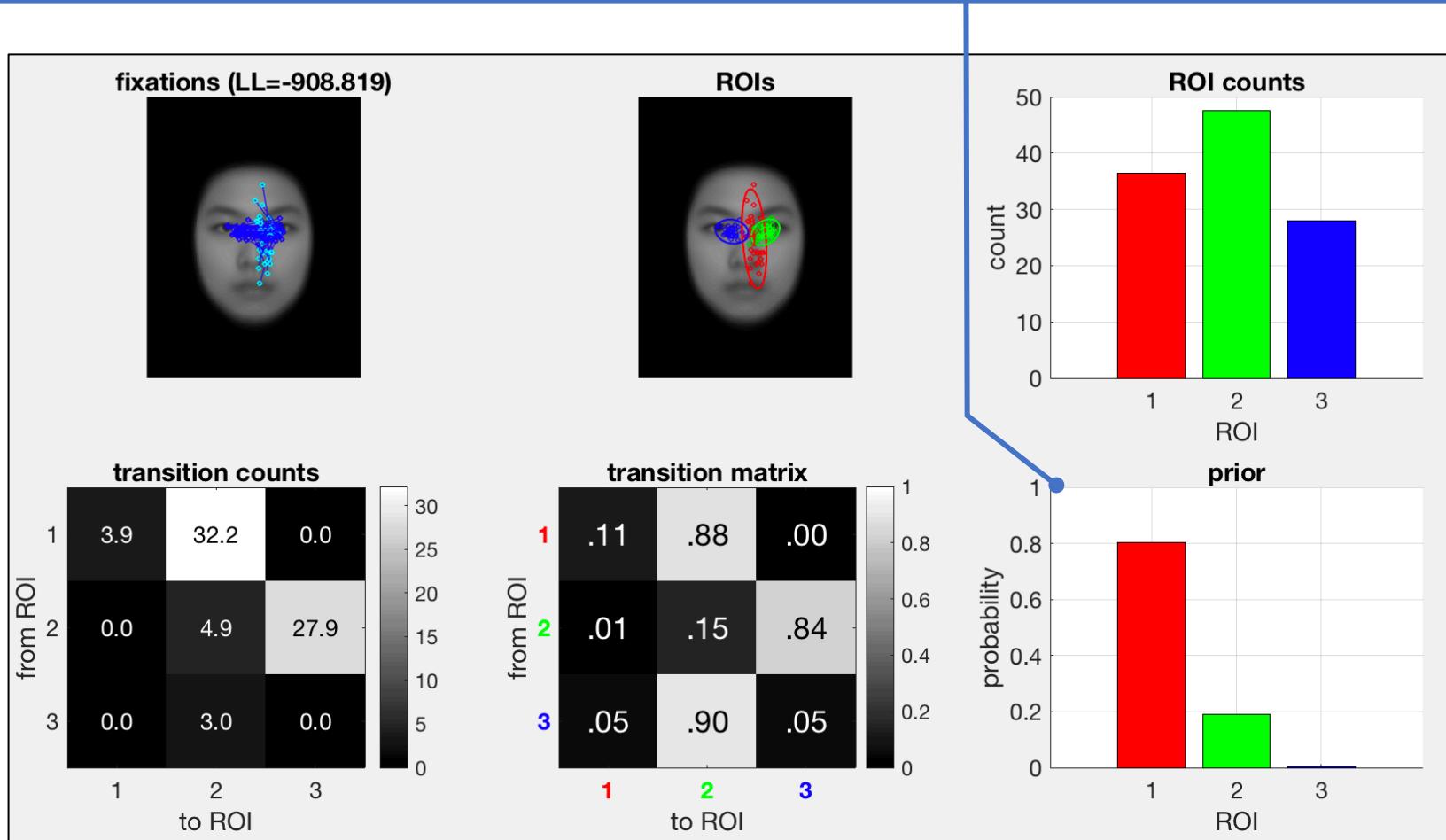
Each row is a probability distribution (sums to one).



# Visualizing an HMM

Figure 1

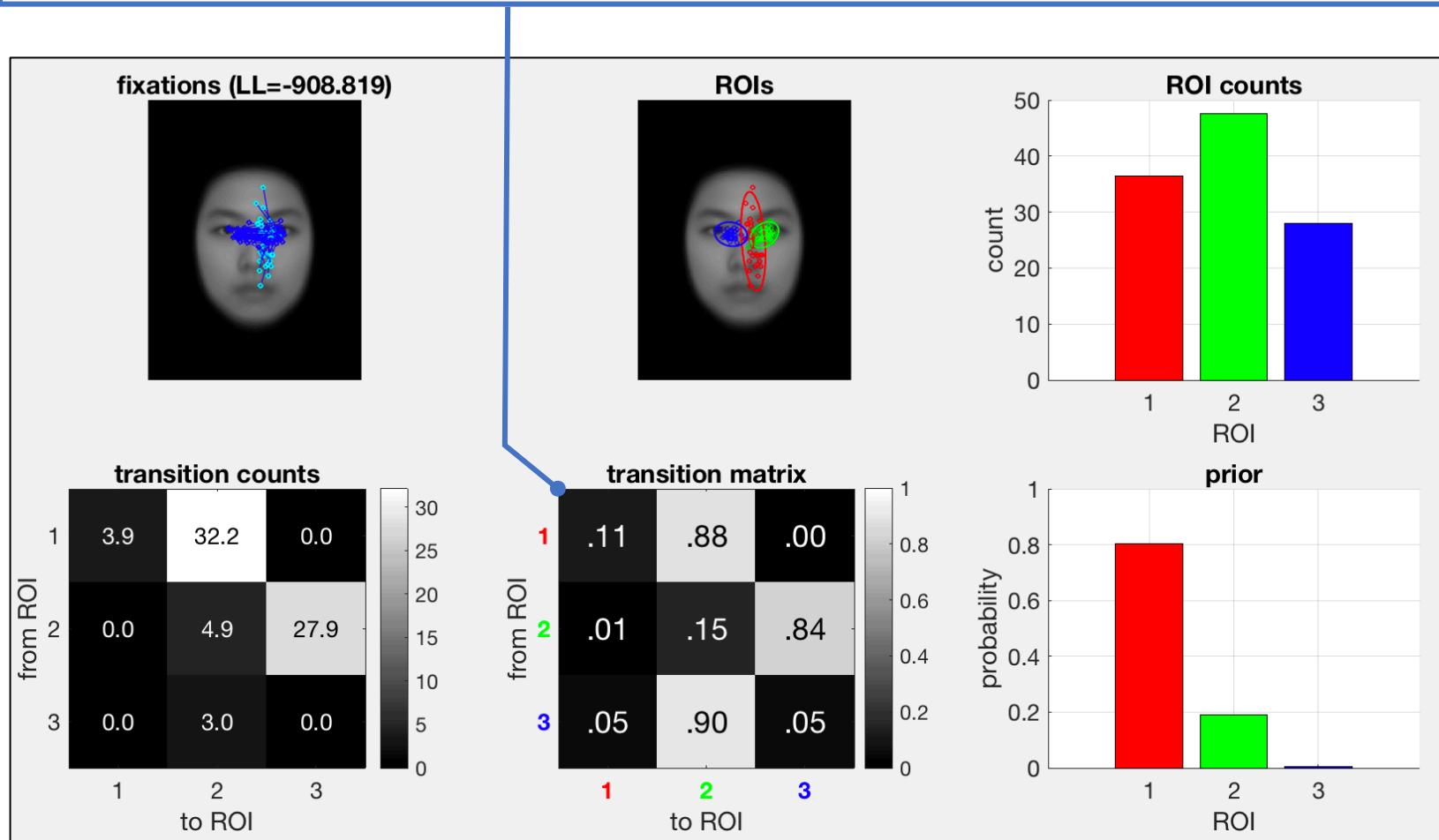
**Prior Probabilities:** probability of the ROI of the first fixation.



# Visualizing an HMM

Figure 1

**Roi Order:** ROIs are automatically sorted according to the most likely fixation path: ROI-1 is the most probable initial fixation; ROI-2 is the most likely next fixation given ROI-1, etc.



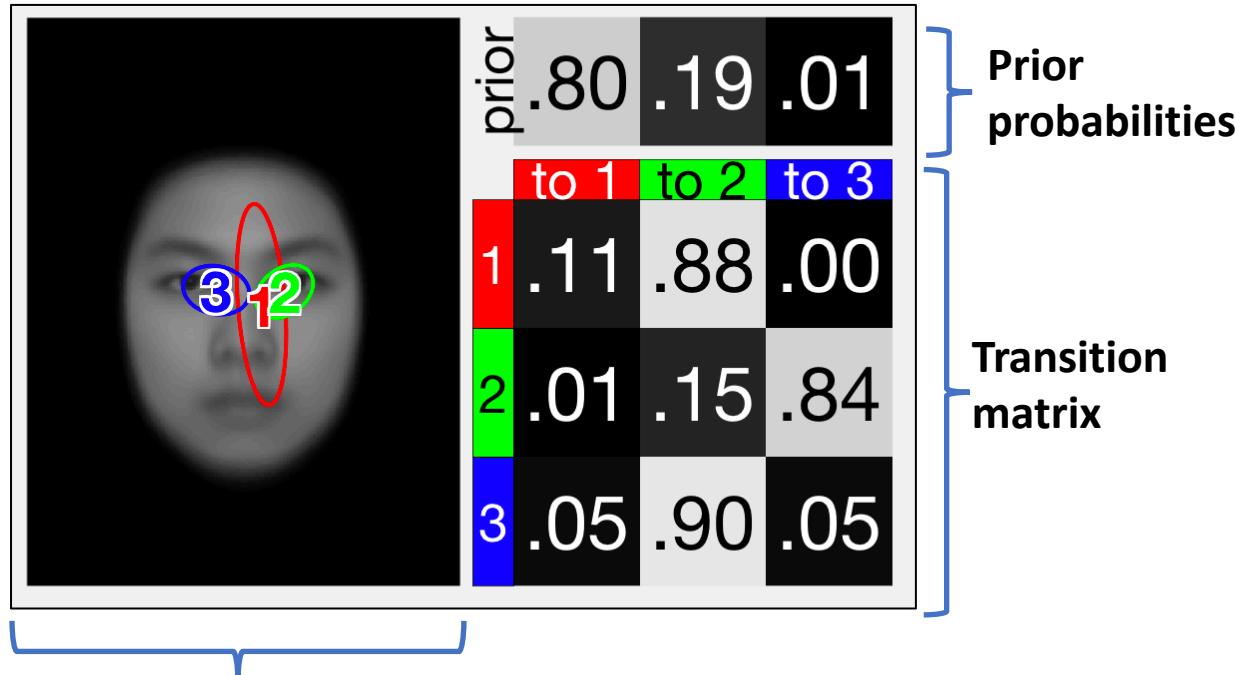
# Compact Visualization

Figure 2

- Visualization without fixations:

```
83 -
84 -
85 -
86 -
87 -
88 -
89 -
90 -  
mys = 4;
vhmm_plot(hmms{mys}, data{mys}, faceimg);
% compact plot
figure, vhmm_plot_compact(hmms{mys}, faceimg);
% show fixations and compact plot
figure
```

- Figure:



**ROIs:** numbers are the ROI centers,  
ellipse represents 95% probability region

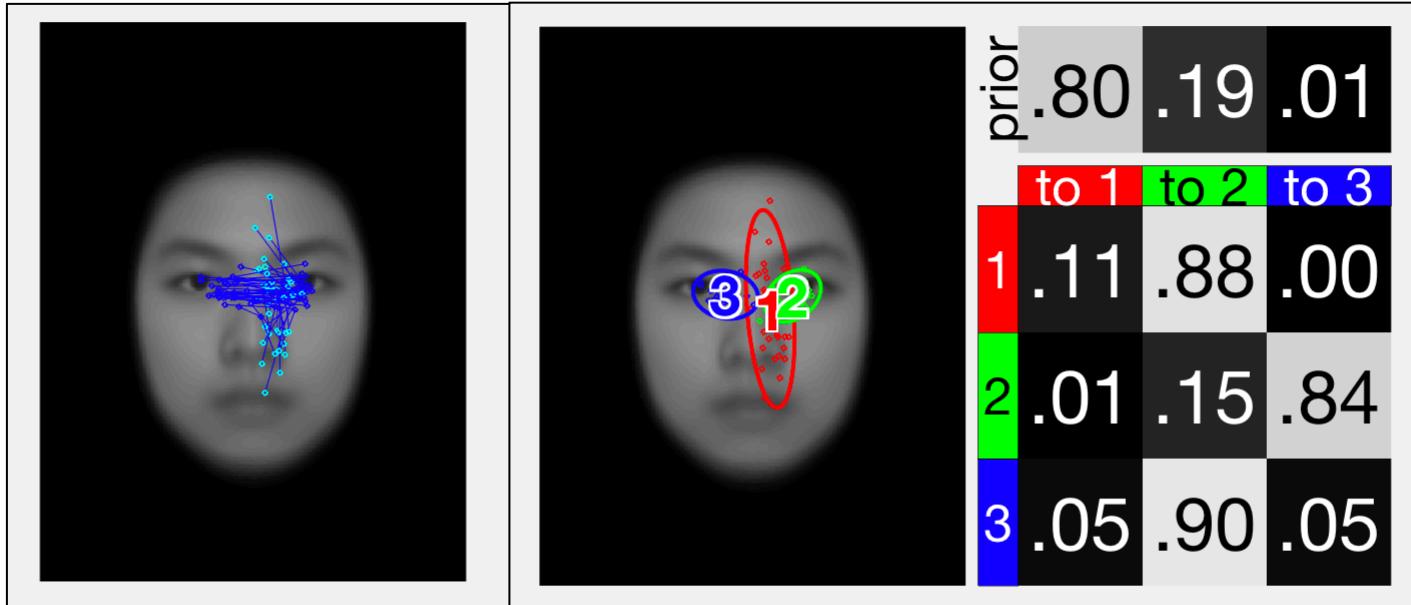
# Visualization with Fixations

Figure 3

- Visualize fixations:

```
88
89 % show fixations and compact plot
90 -
91 -
92 -
93 -
94 -
95 figure
    subplot(2,1,1)
    plot_fixations(data{mys}, faceimg, [], 's');
    subplot(2,1,2)
    vbhmm_plot_compact(hmms{mys}, faceimg, 'r', data{mys});
```

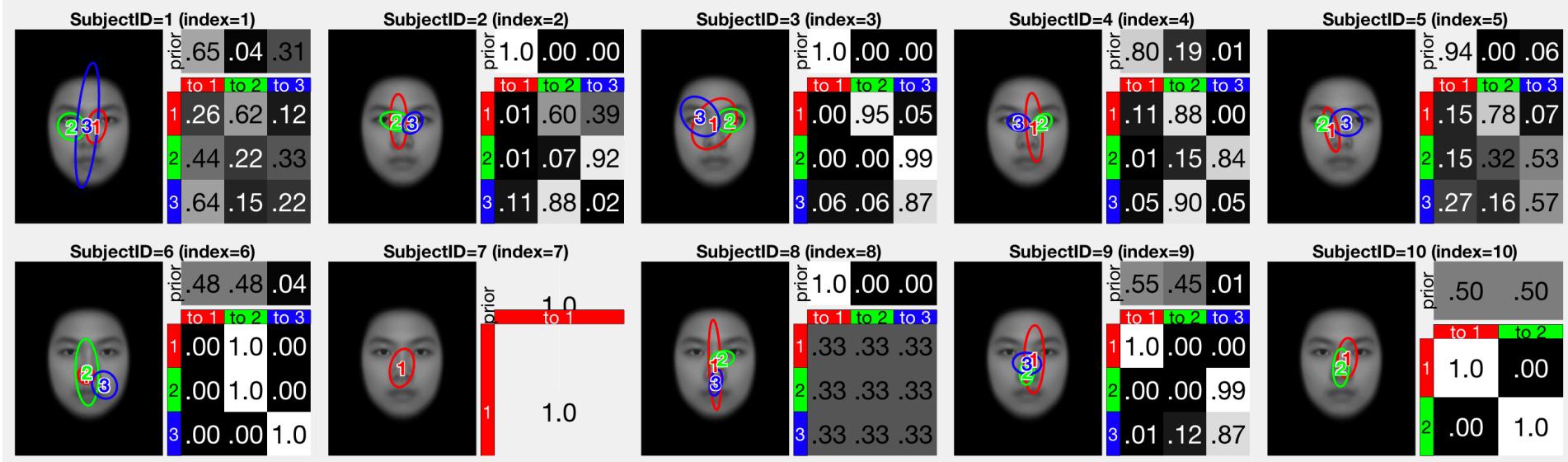
- Output Figure:



# Individual HMMs

Figure 4

- HMMs for the 10 subjects



# Clustering HMMs – One Group

- We use the variational HEM (VHEM) algorithm to cluster HMMs to discover group patterns.
  - 3 different initialization methods are used.
  - 100 trials each.
  - The run with the highest log-likelihood is kept.
- The overall eye gaze pattern is obtained by clustering into 1 group.
- Common eye gaze strategies are obtained by clustering into 2 groups.

# Clustering – 1 Group

- Use **1** cluster to get overall strategy.
    - Set the number of ROIs to **3**

# Viewing the Overall HMM

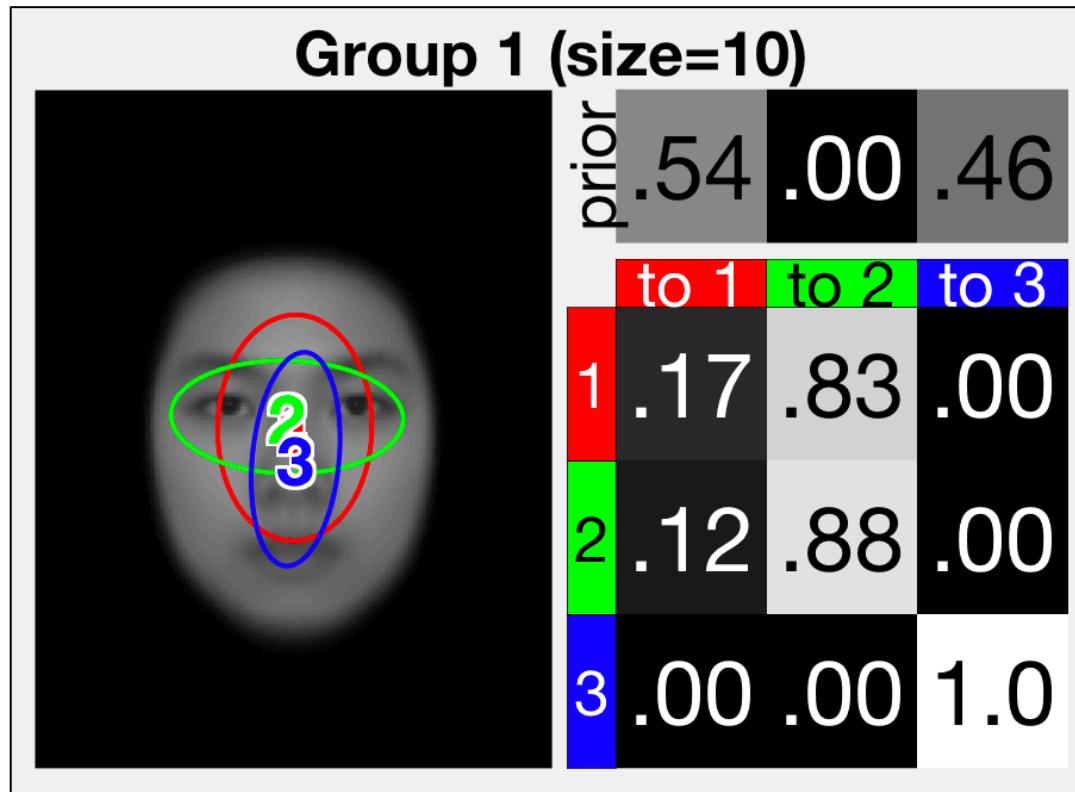
Figure 5

- Plot group HMM:

```
120  
121 -  
122  
123  
124 -  
125
```

```
[all_hmms1] = vhem_cluster(hmms, 1, 3, hemopt) % 1 g  
% plot the overall HMM  
vhem_plot(all_hmms1, faceimg);
```

- Output figure:



# Clustering – 2 Groups

- Set number of clusters to 2

MATLAB R2018b - academic use

Editor - demo\_faces.m Current Folder

```
127
128
129 %% Run HEM Clustering (2 groups) %%%%%%
130 % cluster subjects into 2 groups
131 - fprintf('==> Clustering (2 groups) ==\n');
132 - [group_hmms2] = vhem_cluster(hmms, 2, 3, hemopt) % 2 groups, 3 hidden stat
133
134 % plot the groups
135 - vhem_plot(group_hmms2, faceimg);
136
137 % plot the groups and cluster members|
138 - vhem_plot_clusters(group_hmms2, hmms, faceimg);
139
140 % plot fixations for each group
141 - vhem_plot_fixations(data, group_hmms2, faceimg);
142
143 % plot fixations for each group w/ transition matrix
144 - vhem_plot_fixations(data, group_hmms2, faceimg, 'c');
145
146
147 % show group membership
148 - fprintf('Group membership (indices): \n');
149 - for j=1:length(group_hmms2.groups)
150 -   fprintf('  group %d = %s\n', j, mat2str(group_hmms2.groups{j}));
151 - end
152
153 - fprintf('Group membership (SubjectID): \n');
154 - for j=1:length(group_hmms2.groups)
155 -   fprintf('  group %d = ', j)
156 -   fprintf('%s, ', SubjNames{group_hmms2
157 -   fprintf('\n');
158 - end
```

Command Window

```
.....
.....
.....
.....
Best run is 44: LL=-879.151
auto initialization: trying gmmNew: VHEM Trial: .....
.....
.....
Best run is 44: LL=-878.839
best init was gmmNew; LL=-878.838
group_hmms2 =
struct with fields:
    Z: [10x2 double]
    LogLs: [1x37 double]
    LogL: -878.8376
    label: [2 2 2 2 1 1 1 1 1]
    groups: {[6 7 8 9 10] [1 2 3 4 5]}
    group_size: [5 5]
    hmms: {[1x1 struct] [1x1 struct]}
    hemopt: [1x1 struct]
    emhmm_version: 'v0.75'

Group membership (indices):
group 1 = [6 7 8 9 10]
group 2 = [1 2 3 4 5]
Group membership (SubjectID):
group 1 = 6, 7, 8, 9, 10,
group 2 = 1, 2, 3, 4, 5,
```

Group memberships

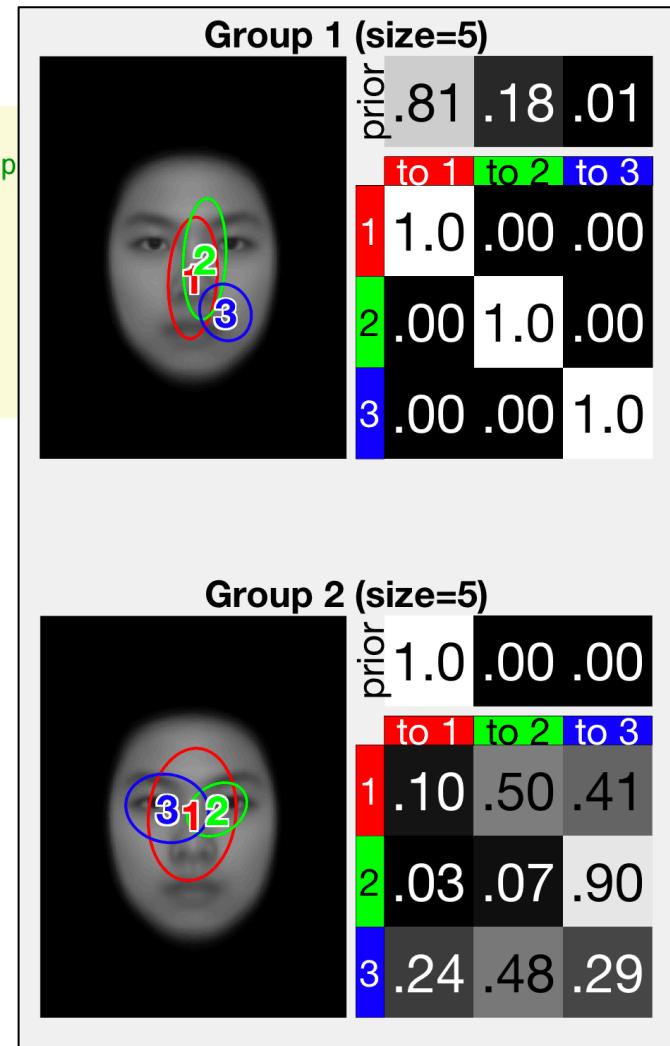
J> <press any key to continue>

# Viewing Group HMMs

Figure 6

- View 2 common strategies

```
131 - fprintf('==== Clustering (2 groups) ====\n');
132 - [group_hmms2] = vhem_cluster(hmms, 2, 3, hemopt) % 2 group
133
134 % plot the groups
135 vhem_plot(group_hmms2, faceimg);
136
137 % plot the groups and cluster members|
138 vhem_plot_clusters(group_hmms2, hmms, faceimg);
```



# Visualize Groups with fixations

Figure  
10

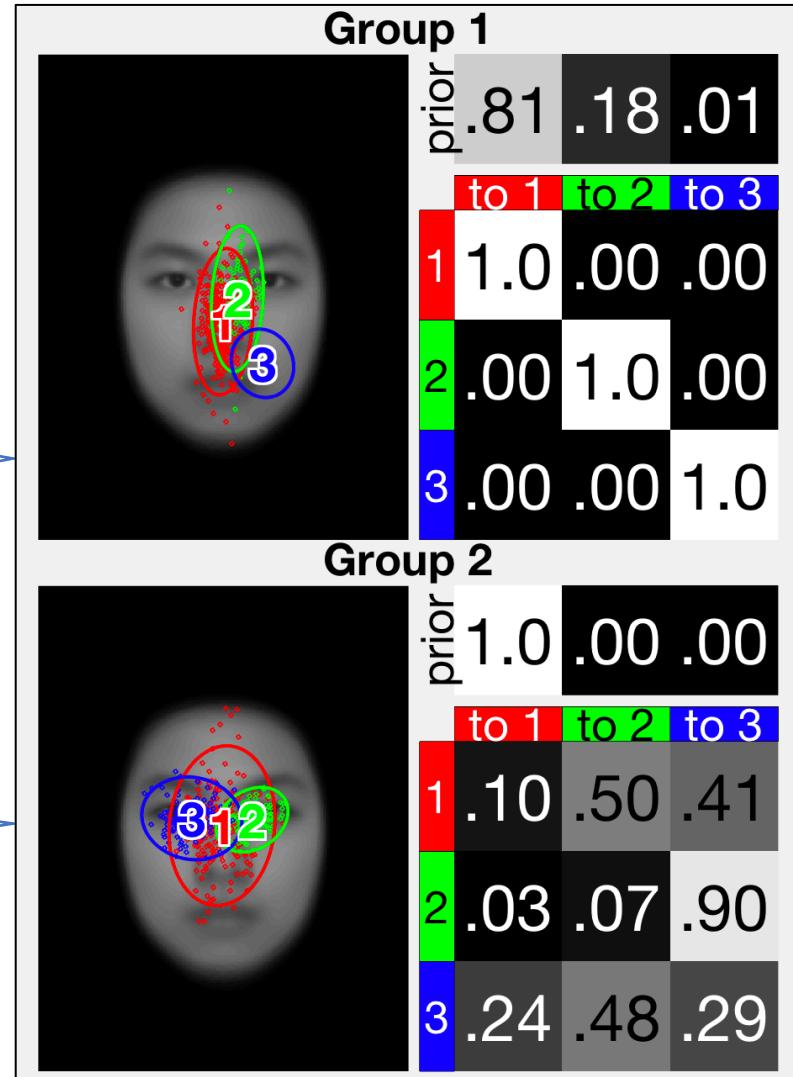
- Plot fixations with group HMMs:

```
142  
143  
144 -  
145  
146
```

```
% plot fixations for each group w/ transition matrix  
vhem_plot_fixations(data, group_hmms2, faceimg, 'c');
```

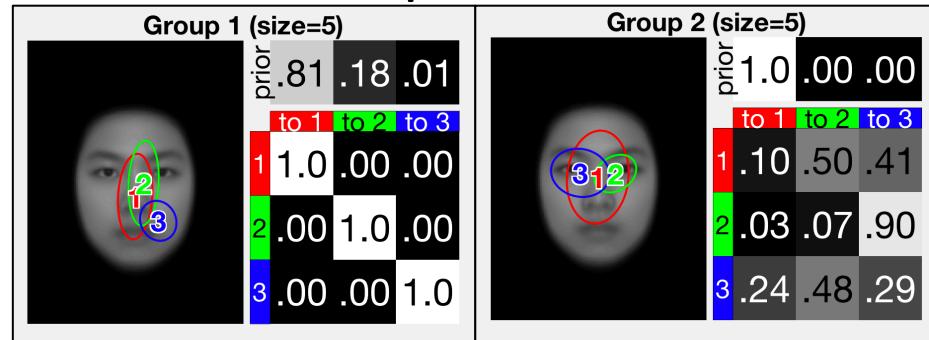
Shows fixations from all subjects assigned to that group.

Fixation color indicates the assigned ROI



# Analysis: Most Probable ROI Sequence

- Compute most probable sequences (length 3).



MATLAB R2018b - academic use

HOME PLOTS APPS EDITOR PUBLISH VIEW

/ Users abc Documents Work repo src-fixations emhmm-toolbox demo

Editor - demo\_faces.m Current Folder

```
161
162 % get the most likely ROI sequences ****%
163 - fprintf('n*** top-5 most probable ROI sequences ***n');
164 - [seqs{1}, seqs_pr{1}] = stats_stateseq(group_hmms2.hmms{1},| 3);
165 - [seqs{2}, seqs_pr{2}] = stats_stateseq(group_hmms2.hmms{2}, 3);
166
167 % show the sequences
168 - fprintf('prob : hmm1 ROI seq n');
169 - fprintf('-----n');
170 - for i=1:5
171 -   fprintf('%0.4f : ', seqs_pr{1}(i));
172 -   fprintf('%d ', seqs{1}{i});
173 -   fprintf('n');
174 - end
175
176 % show the sequences
177 - fprintf('nprob : hmm2 ROI seq n');
178 - fprintf('-----n');
179 - for i=1:5
180 -   fprintf('%0.4f : ', seqs_pr{2}(i));
181 -   fprintf('%d ', seqs{2}{i});
182 -   fprintf('n');
183 - end
```

Most-likely ROI sequence  
is 1-1-1, with 80.7%  
probability

Most-likely ROI sequence  
is 1-2-3, with 44.7%  
probability

group 1 = [6 7 8 9 10]
group 2 = [1 2 3 4 5]
Group membership (SubjectID):
group 1 = 6, 7, 8, 9, 10,
group 2 = 1, 2, 3, 4, 5,
<press any key to continue>

\*\*\* top-5 most probable ROI sequences \*\*\*
prob : hmm1 ROI seq
-----
0.8067 : 1 1 1
0.1849 : 2 2 2
0.0084 : 3 3 3
0.0000 : 1 2 2
0.0000 : 1 1 2

prob : hmm2 ROI seq
-----
0.4465 : 1 2 3
0.1934 : 1 3 2
0.1163 : 1 3 3
0.0954 : 1 3 1
0.0488 : 1 1 2
<press any key to continue>

# Analysis: Group Differences

- Test if two group HMMs are different:
  - Using data from Group 1, calculate average log-likelihood difference under Group 1 and Group 2 HMMs.
    - This is an approximation of the KL divergence between Group 1 and Group 2.
  - Use t-test to check if the average log-likelihood difference is significantly different from 0 → the two HMMs are different.
  - Same process using data from Group 2.

# Analysis: Group Differences

- Select Group data and perform t-test.

```
187
188 %% Statistical test %%%%%%
189 % collect data for group 1 and group 2
190 - data1 = data(group_hmms2.groups{1});
191 - data2 = data(group_hmms2.groups{2});
192 -
193 - fprintf('\n*** t-test ***\n');
194 -
195 % run t-test for hmm1
196 [p, info, lld] = stats_ttest(group_hmms2.hmms{1}, group_hmms2.hmms{2}, data1);
197 fprintf('- test group hmm1 different from group hmm2: t(%d)=%g; p=%g\n', info);
198 -
199 % run t-test for hmm2
200 [p, info, lld] = stats_ttest(group_hmms2.hmms{2}, group_hmms2.hmms{1}, data2);
201 fprintf('- test group hmm2 different from group hmm1: t(%d)=%g; p=%g\n', info);
202 -
203 -
204 %% get mean log-likelihoods (e.g., to use with a correlation test) %%%%%%
205 - fprintf('\n*** get mean LL for each subject ***\n');
206 -
207 [mLL1] = stats_meanll(group_hmms2.hmms{1}, data1);
208 [mLL2] = stats_meanll(group_hmms2.hmms{2}, data2);
209 -
210 - fprintf('- mean LL for each subject under group hmm1:\n');
211 mLL1
212 - fprintf('- mean LL for each subject under group hmm2:\n');
213 mLL2
214 -
215 %% save group models to file
216 - fprintf('saving group models to %s\n', matfile_group);
217 - save(matfile_group, 'hemopt', 'all_hmms1', 'group_hmms2', 'faceimg');
218 - % later on you can use the following command to read back the models:
219 - % load models_demo_faces_group.mat
220 -
221 -
222
```

```
prob : hmms1 ROI seq
-----
0.8067 : 1 1 1
0.1849 : 2 2 2
0.0084 : 3 3 3
0.0000 : 1 2 2
0.0000 : 1 1 2

prob : hmms2 ROI seq
-----
0.4465 : 1 2 3
0.1934 : 1 3 2
0.1163 : 1 3 3
0.0954 : 1 3 1
0.0488 : 1 1 2
<press any key to continue>
```

Both *p*-values show  
HMMs are significantly  
different.

```
*** t-test ***
- test group hmm1 different from group hmm2: t(4)=8.15202; p=0.000616171
- test group hmm2 different from group hmm1: t(4)=6.41628; p=0.00151614

*** get mean LL for each subject ***
- mean LL for each subject under group hmm1:

mLL1 =
-11.5196 -9.9474 -12.3179 -10.6476 -12.3742 -8.5738 -8.1270

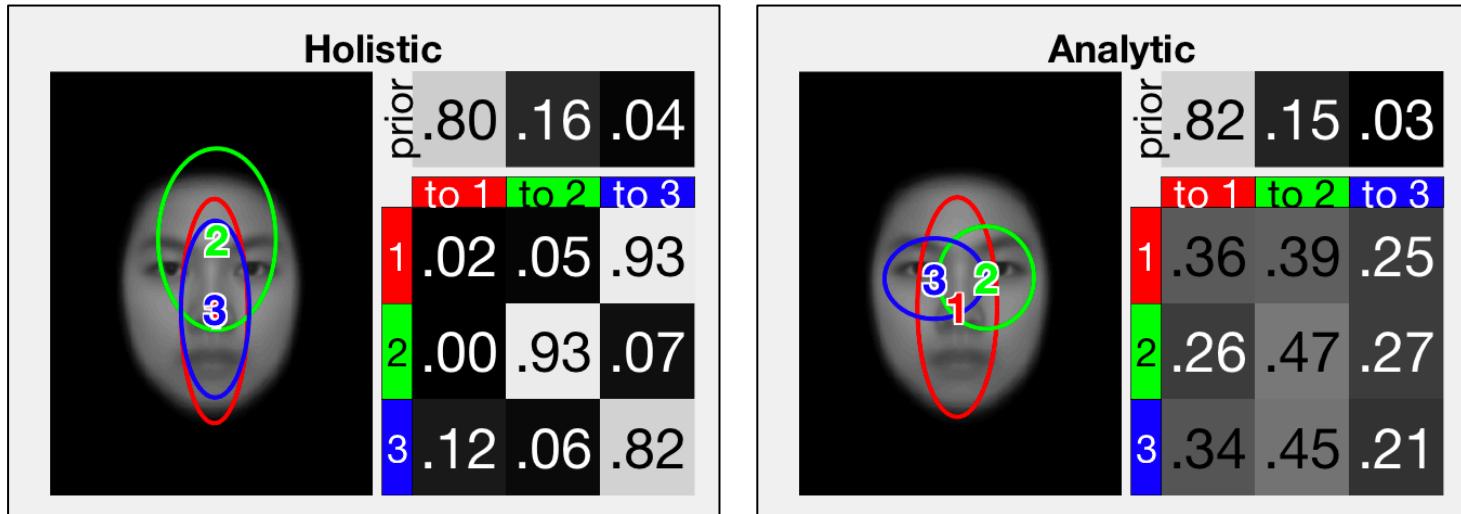
- mean LL for each subject under group hmm2:

mLL2 =
-9.2684 -8.4376 -8.8677 -8.5927 -8.8478 -10.1217 -9.1723

saving group models to models_demo_faces_group.mat
fx >>
```

# Representative Holistic/Analytic Models

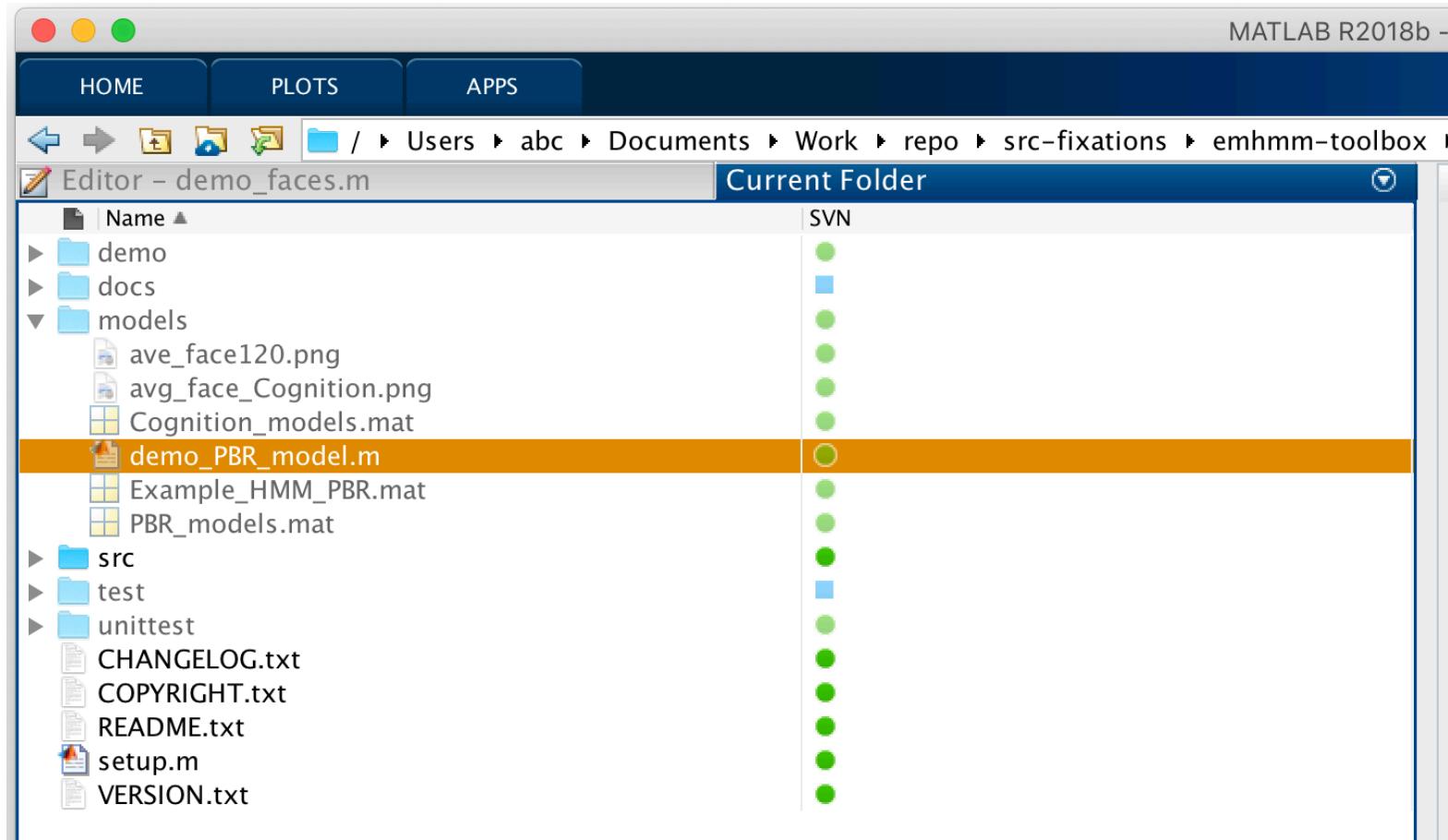
- Representative models for holistic/analytic strategies are available for computing H-A scale.



- Models are from: Cynthia Y.H. Chan, Antoni B. Chan, Tatia M.C. Lee, and Janet H. Hsiao, “Eye Movement Patterns in Face Recognition are Associated with Cognitive Decline in Older Adults.” *Psychonomic Bulletin & Review*, 25(6): 2200-2207, Dec 2018.

# Demo Script for HA Models

- In the “models” directory of the toolbox:



# Computing H-A Scale

- Using eye fixation data from new subjects, compute H-A scale of each subject.
  - *Note: requires similar stimuli sizes and layout.*
- Compute log-likelihood under each group with `stats_meanll`.

The screenshot shows the MATLAB R2018b interface. The top menu bar includes HOME, PLOTS, APPS, EDITOR, PUBLISH, and VIEW. The toolbar has various icons for file operations. The current folder path is displayed as /Users/abc/Documents/Work/repo/src-fixations/emhmm-toolbox/models. The Editor window shows a script named demo\_PBR\_model.m with several lines of MATLAB code. The Command Window displays the output of the script, specifically the computation of mean log-likelihood (LL) for each subject and the resulting matrix LL\_all.

```
101 % calculate Log-likelihood of converted data with the model %%
102 % The output will be stored in LL_all under workspace
103
104 fprintf('\n*** get mean LL for each subject ***\n');
105
106 LL_all = [];
107 for i=1:length(model.group_hmms2.hmms)
108     [mLL] = stats_meanll(model.group_hmms2.hmms{i}, data_s);
109     LL_all = [LL_all;mLL];
110     fprintf('Column %d refers to the LL of the %s pattern\n',i, model.model
111 end
112
113 LL_all=LL_all'
114
115
116 %% compute H-A scale
117 fprintf('\n*** H-A scale ***\n');
```

Command Window Output:

```
1
*** get mean LL for each subject ***
Column 1 refers to the LL of the Holistic pattern
Column 2 refers to the LL of the Analytic pattern
LL_all =
-10.3345 -9.3578
-9.8063 -9.1211
-9.5015 -9.2958
-9.3297 -9.2799
-8.5411 -9.0032
-9.6745 -10.6560
-9.2844 -9.0184
-9.6761 -9.2645
```

# Computing H-A Scale

- H-A scale:  $HA = \frac{LLH - LLA}{|LLH| + |LLA|}$

```
106 -  
107 -     LL_all = [];  
108 -     for i=1:length(model.group_hmms2.hmms)  
109 -         [mLL] = stats_meanll(model.group_hmms2.hmms{i}, data_s);  
110 -         LL_all = [LL_all;mLL];  
111 -         fprintf('Column %d refers to the LL of the %s pattern\n', i, model.model  
112 -     end  
113 -     LL_all=LL_all'  
114 -  
115 -  
116 -     %% compute H-A scale  
117 -     fprintf('\n*** H-A scale ***\n');  
118 -     HAscale = (LL_all(:,1) - LL_all(:,2)) ./ sum(abs(LL_all), 2)  
119 -  
120 -     % sort HA values in ascending order  
121 -     [HAscale_sort, inds] = sort(HAscale);  
122 -  
123 -     % select 8 values  
124 -     inds2 = inds(round(linspace(1,length(inds),8)));  
125 -  
126 -     %% make a nice plot
```

-9.5015	-9.2958
-9.3297	-9.2799
-8.5411	-9.0032
-9.6745	-10.6560
-9.2844	-9.0184
-9.6761	-9.2645

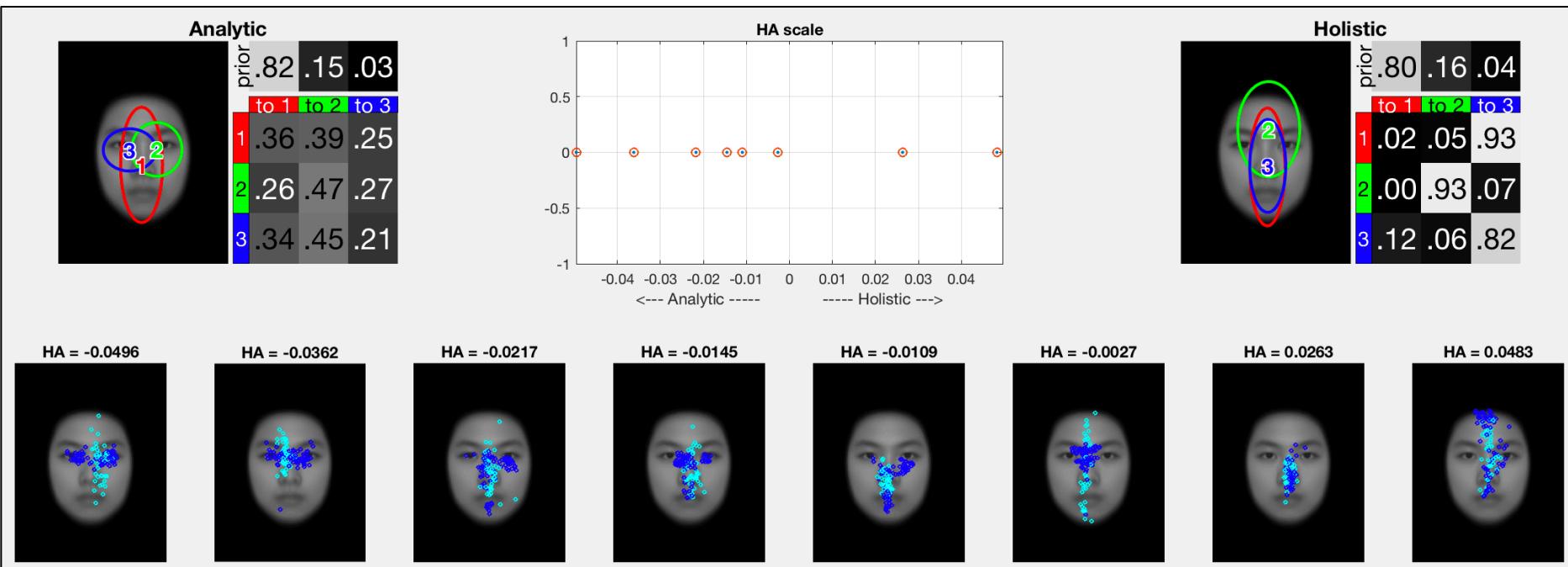
\*\*\* H-A scale \*\*\*

HAscale =
-0.0496
-0.0362
-0.0109
-0.0027
0.0263
0.0483
-0.0145
-0.0217

Positive HA indicate more holistic;  
negative HA is more analytic

# Visualization

- Showing fixations of each subject and their HA value.

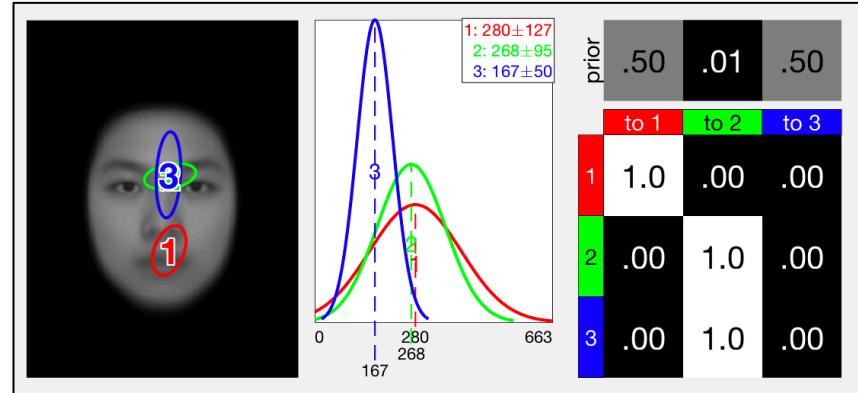


Most analytic

Most holistic

# More Resources

- Analyzing eye fixation location & duration
  - `demo_faces_duration.m`
- JOV paper demo:
  - `demo_faces_jov_clustering.m`
  - `demo_faces_jov_compare.m`
- Converting coordinates between face images
  - `demo_conversion.m`
- More details about options for each function
  - See `emhmm-documentation.pdf`
  - MATLAB help: e.g., “`help vbhmm_learn`”



# Questions?

- EMHMM toolbox:
  - <http://visal.cs.cityu.edu.hk/research/emhmm/>



- Acknowledgements:

- This research was supported by the Research Grant Council of Hong Kong SAR: #17402814 and HKU 745210H for J.H. Hsiao; CityU 110513 and G-CityU109/14 for A.B. Chan. We also thank the HKU Seed Funding Programme for Basic Research (Project number 201311159131 to J.H. Hsiao).