BRAIN STATE DYNAMICS (BSD) TOOLBOX USER'S MANUAL

Brain Dynamics Laboratory

BrainDynamicsLab.org

David Araya

Wael El-Deredy

Nelson Trujillo-Barreto

Version 1.0

(November 2018)

Table of Contents

Contents

1 Introduction

1.1 What is BSD?

BSD is an interactive Matlab toolbox for dynamic Brain State Allocation, segmenting a multivariate time series (e.g. EEG/fMRI) into a set of time intervals (Brain States) that have a useful interpretation in some underlying domain. BSD is developed from the perspective that the brain is a Hybrid Dynamical Systems: A continuous system with discrete logic, where the observed sequence of data is generated (or emitted) probabilistically by a hidden sequence of discrete Brain States. Here a Brain State signifies a characteristic quasi-stationary pattern of activity or functional/effective connectivity. Time series data can be any type of signal like EEG, MEG, LFP, fMRI etc.

The BSD toolbox assumes the Hybrid System can be described by a Hidden Semi Markov Model (HsMM) [Yo, 2010], and uses Bayesian inference theory to estimate its parameters from the observed data [Beal, 2003].

1.2 Why BSD?

BSD can be used to solve a variety of Brain State allocation problems in neuroimaging, such as, inferring dynamical brain networks, identifying brain transitions during spontaneous, resting state or ecological conditions, etc. Currently, it is the only implementation that explicitly models Brain State duration.

1.3 Hidden Semi Markov Model structure

As mentioned above, a HsMM models an observed continuous time series as being generated (emitted) from a dynamical system that transitions probabilistically between discrete unobservable (hidden) states. While the system is in a given state, it emits a data sequence for a period of time, called the duration of the state (Figure 1.1).

Thus a HsMM is defined by four components/models:

Emission Model (Emis_model):	This is a probabilistic model that specifies how the emitted sequence is generated from each hidden state.
Transition Matrix (Trans_model):	This parameter specifies the transition probabilities between the hidden states.
Duration Model (Dur_model):	This is a probabilistic model of the duration of each hidden states.
Initial condition (In_model):	This parameter specifies the probability of the being system at each state at the beginning of the time series.

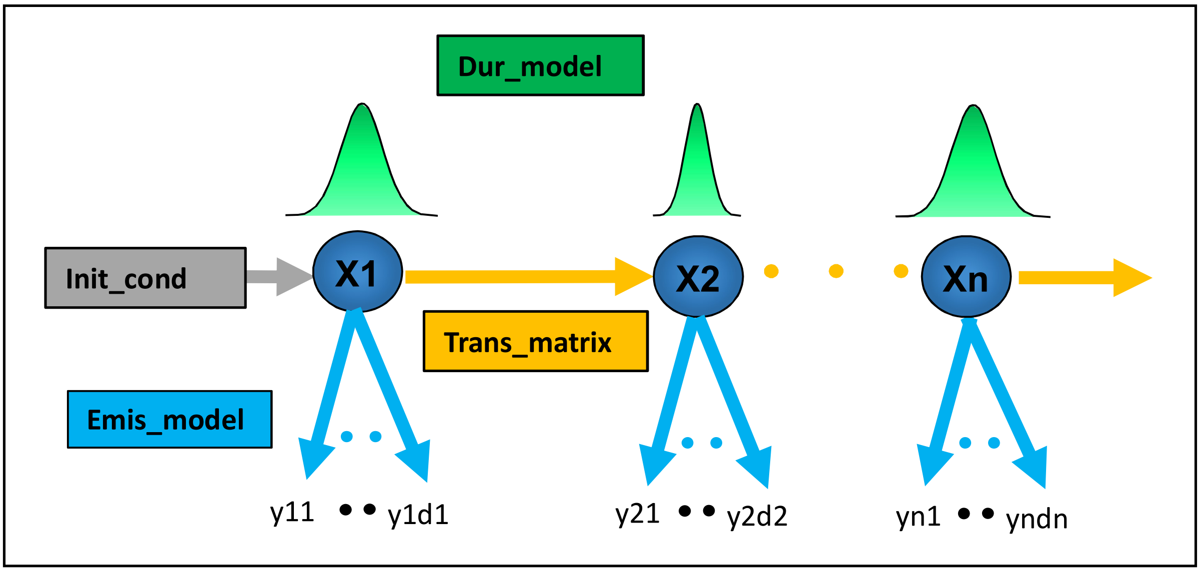


Figure 1.1: Schematic diagram of the components of HsMM. The system evolves from Initial conditions [Init_cond] and transitions between unobservable hidden states [X] according to a transition matrix [Trans_matrix]. When in a given state, the system emits data [y], according to an emission model [Emis_model] for durations specified by [Dur_model]

Model Parameters

HSMM components are defined by probability distributions, prior over the parameters. For Example: to characterize the state duration using a *unidimensional normal distribution*, two parameters need to be estimated, the *mean* and the *precision* (inverse of variance). A prior is defined for each: normal distribution over the mean, and a gamma over the precision, which in turn have their own (hyper)parameters (Figure 1.2)

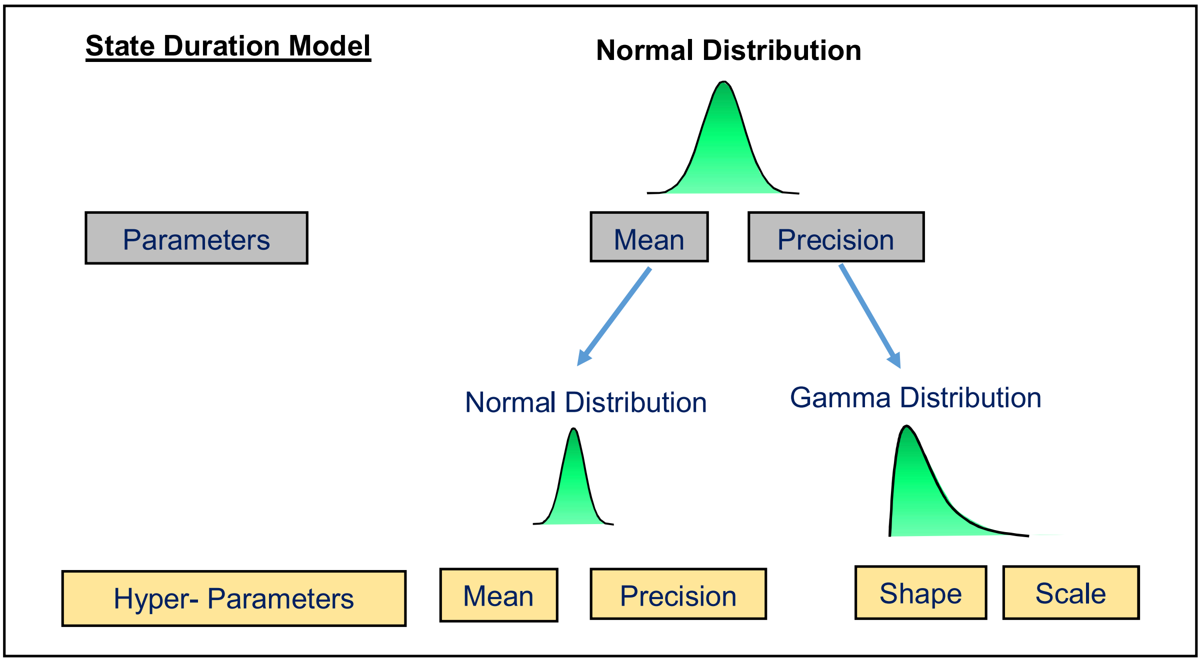


Figure 1.2: Example of the probabilistic specification of model parameters. Here the state duration is specified by a Normal Distribution, parameterized by a mean and a precision. Priors are defined over those paramaters. In this case a Normal distribution for the mean and a Gamma distribution for the precision. The latter distributions areparamterized by their on (hyper)paramaters: a mean and a precision for the Normal and a shape and scale for the Gamma. Parameters are estimated during the model training step.

2 BSD toolbox Overview

2.1 Installation

BSD is a Matlab toolbox, it requires Matlab version 2015 and above. There is no installation, but you need to add it to the matlab path.

>> addpath(folderBSDprogram)

BSD uses some functions from the freely available toolbox EEGlab
https://sccn.ucsd.edu/eeglab/index.php

2.2 BSD is a platform

BSD is a software developed using oriented object programming (OOP), which allows to implement a layer of abstraction that facilitates modularity of BSD. For this reason BSD is a platform that offers a variety of model components. Furthermore, BSD allows HSMM to grow over time, avoiding incompatibility problems and making it possible for external users to create their own components (See Figure 1.1)

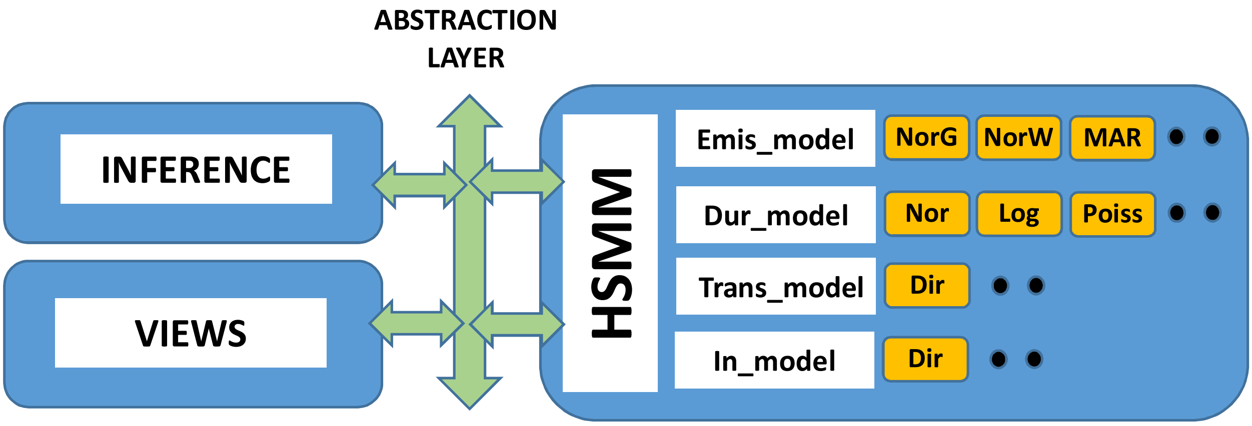


Figure 2.1: Scheme of abstraction layer between object

2.3 Starting BSD Program

To start BSD type

>> BSD

2.4 Graphics User Interface (GUI)

The BSD GUI is designed to allow non-experienced Matlab users to analyze data. It is also possible for more experienced users to alternate between the GUI and Matlab Command line.

BSD has Main window where it keeps the relevant information about the current analysis: loaded data; Models specified; progress of processing; among others (see figure 3).

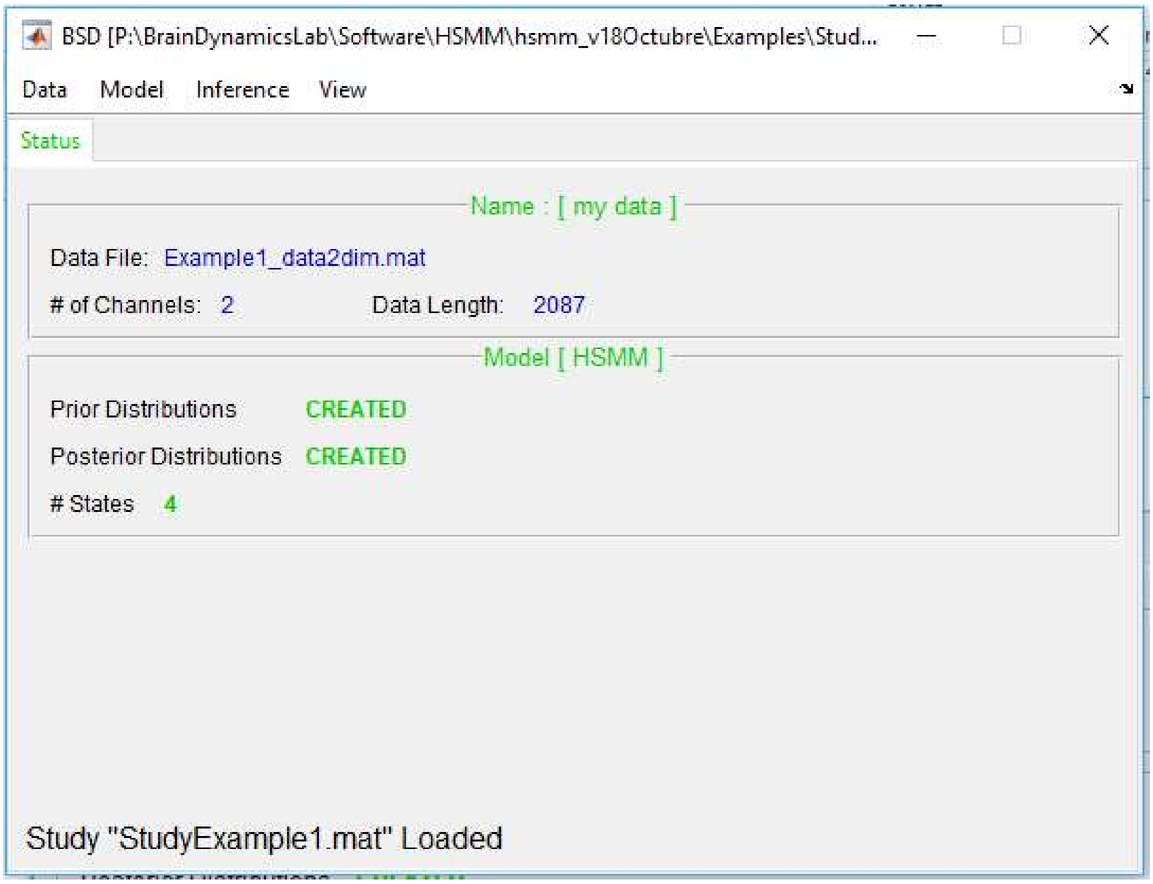


Figure 2.2: BSD Main window

The BSD Main window information is the following:

<u>Name Frame:</u>

Data File:	Name of data file loaded
# of Channels:	Number of channels/sensor of data
Data Length:	Number of data points

Model Frame

Prior Distribution:	Indicates if and which priors over the model parameters have been specified. The priors are selected when the model is created.
Posterior Distribution:	Indicates if the model parameters have been estimated. Model parameters are estimated using Bayesian inference during Model training.
# States:	Indicates the number of brain states. During the model specification one can choose a number or allow the model to find the optimal number of state. After training, this will show the number of states estimated.

The text in the bottom the windows indicates the status of study: Loaded data (in the example of Figure 3 “Study “StdExample1.mat”); Training Model, Model Trained; etc.

The menu bar of the Main window allows the following options:

Data:

This dropdown menu allows importing data of different formats and management the Study (Detail in Data Management section 3).

Model:

This dropdown menu includes all options related with specifying a model, as well as viewing and editing the model parameters.

Inference:

This dropdown menu includes the possible inference procedures that could be carried out on the model: e.g estimate the model parameters; estimate the state probabilities, estimate most probable sequence, among others (Detail in Inference section 6).

Views

This menu includes different types of plots and figures of the results of the inference (Detail in

Views section 0).

3 Data Management

3.1 Single block data

This is the simple case of one data matrix, where the number rows is the number of time points (the length of the time series), and the number of columns is the number of emission channels/sensors.

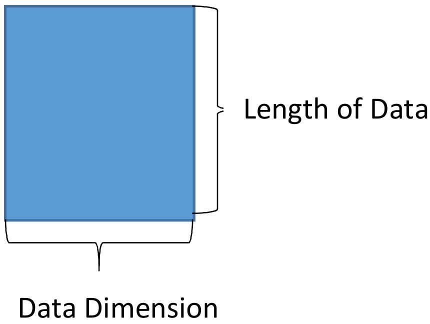


Figure 3.1: Single Block data

It is possible to load data using the options:

Data -> Import Data -> From ASCII File

Loads an ASCII format file from disk

Data -> Import Data -> From Matlab File

Loads a Matlab file from disk. Since Matlab files can contain several variables, the option opens another menu to select the desired data variable to import.

Data -> Import Data -> From Workspace Matlab

Imports an existing variable from Matlab workspace.

3.2 Multiblock data

This option deals with studies with multi -subject, -condition or -blocks. Here the data organized as a Matlab structure in the following order:

VariableName.cond(j).subj(k).block(i).data

The data in each block can have different length, in addition each subject can have different number of blocks. All blocks must have the same number of emission channels/sensors.

Data in each block is organized, as in 3.1, where the number rows is the number of time points (the length of the time series), and the number of columns is the number of emission channels/sensors.

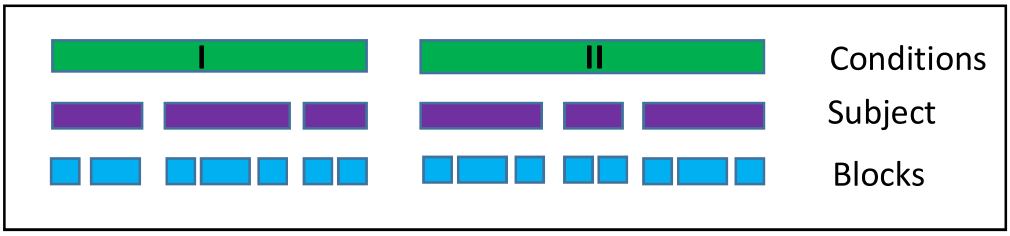


Figure 3.2: Multiblock data

Multiblock data can be imported using

Data -> Import Data -> From Matlab File

or

Data -> Import Data -> From Workspace Matlab

4 Study

BSD organizes the data, models and results in an instantiation called **Study**.

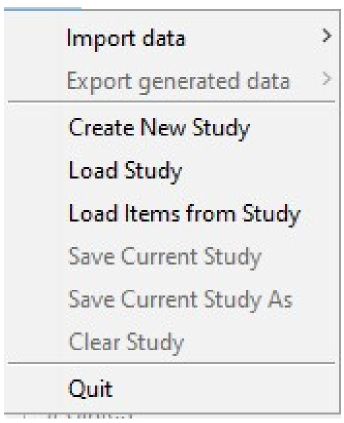


Figure 4.1: Study options

4.1 Load existing Study

In addition to creating a new Study, you'll often need to load a Study that had been previously saved.

4.2 Load Items from an existing Study

BSD allows loading one or more components of a Study, for example: the previously trained model; or some of the inference results. A pop-up windows will open to select the required component(s) (Figure 4.2).

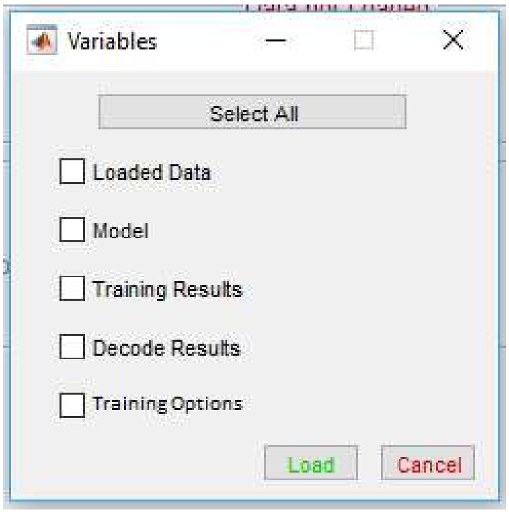


Figure 4.2: Load items

4.3 Save and Save As current Study

The two options save the current Study. Save as requires a name to be specified.

4.4 Clear Study

This option clears all components from the current Study.

5 Model

5.1 Create Model

The create model window specifies the probability distributions that describe each component of model (Figure 5.1)

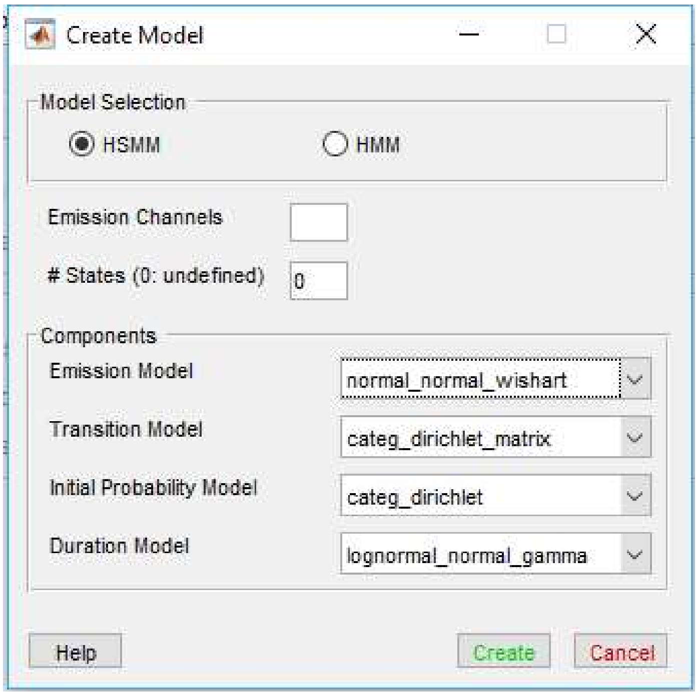


Figure 5.1: Load Create Model

Table 5.1: Window Description of Create Model

N	Definition	Description	Options
1	Model Selection	Choose the type of model to build	HSMM / HMM
2	Emission Channels	The number of rows of data matrix	Integer number
3	# States	Specify number of brain states in the model	Integer number. If 0 is entered, BSD will search for and return the optimal number of states (can be slow)

N	Definition	Description	Options
4	Emission Model	Choose the type of model that describes the states emissions.	normal_normal_wishart (This indicates that the emissions come from a multivariate normal distribution with a Mean and a Precision matrix. The prior of the mean is also a multivariate normal, while the prior of the precision is a Wishart distribution). normal_normal_gamma (This indicates the emissions come from a multivariate normal distribution with a Mean and a ***Diagonal*** Precision matrix. The prior of the mean is also a multivariate normal, while the prior of the precisions are Gamma distribution). mar This indicates that the emissions are described by a Multivariate Autoregressive process with zero mean Gaussian additive noise.
5	Transition Matrix	The rows of the transition matrix are assumed to have Dirichelet distrbutions	categ_dirichlet_matrix categ_dirichlet_matrixdiag0: Matrix diagonal values are zeros
6	Initial Probability	The prior on the initial probability are conjugate Dirichelet distrbutions (symmetric and scaled)	There are currently no options here
7	Duration Model	Selection of Duration Model	Normal and Log normal

5.2 Viewing and editing the prior distributions

When the model is created the priors are set to be “Non Informative” by default. To view and edit the priors select:

Model -> Prior Distribution (see **Error! Reference source not found.**)

For a worked example see section **Error! Bookmark not defined. Error! Reference source not found.**

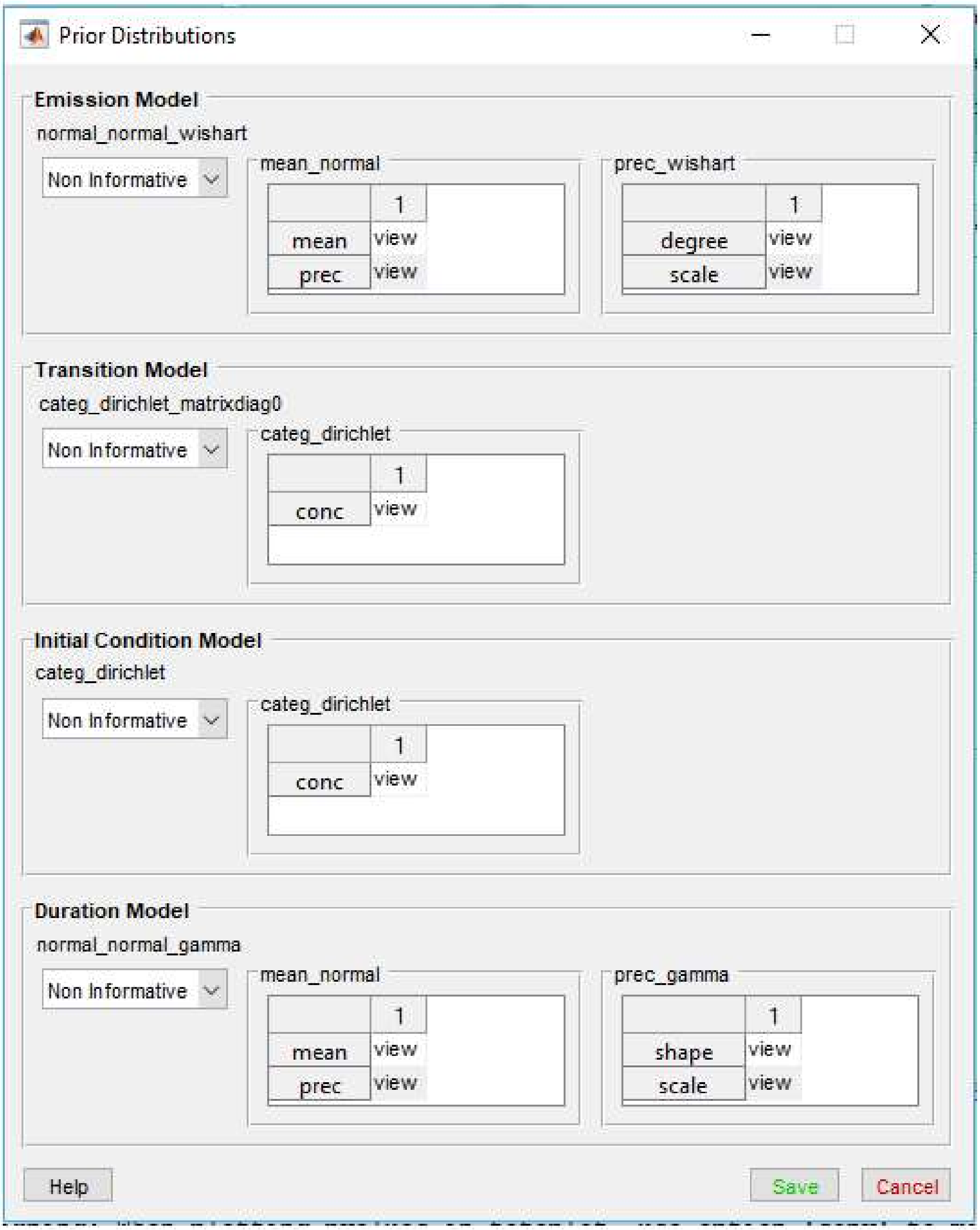


Figure 5.2**:** The Prior distributions edit window. In the above example, the emission model is "normal_normal_wishart", that is the mean emission of each state follows a normal distribution, the priors over its mean and precision matrix are mean and Wishart distributions, respectively. In turn, the mean and precision are parameterized by editable (hyper)priors.

While the default of the priors is "Non Informative", to allow the observed data to provide the optimal estimates, the values of the prior parameters could be set manually, by selecting "User defined" and "Edit" manually each parameter's box(figure), or alternatively load the values a Matlab variable from the workspace (**Error! Reference source not found.**)

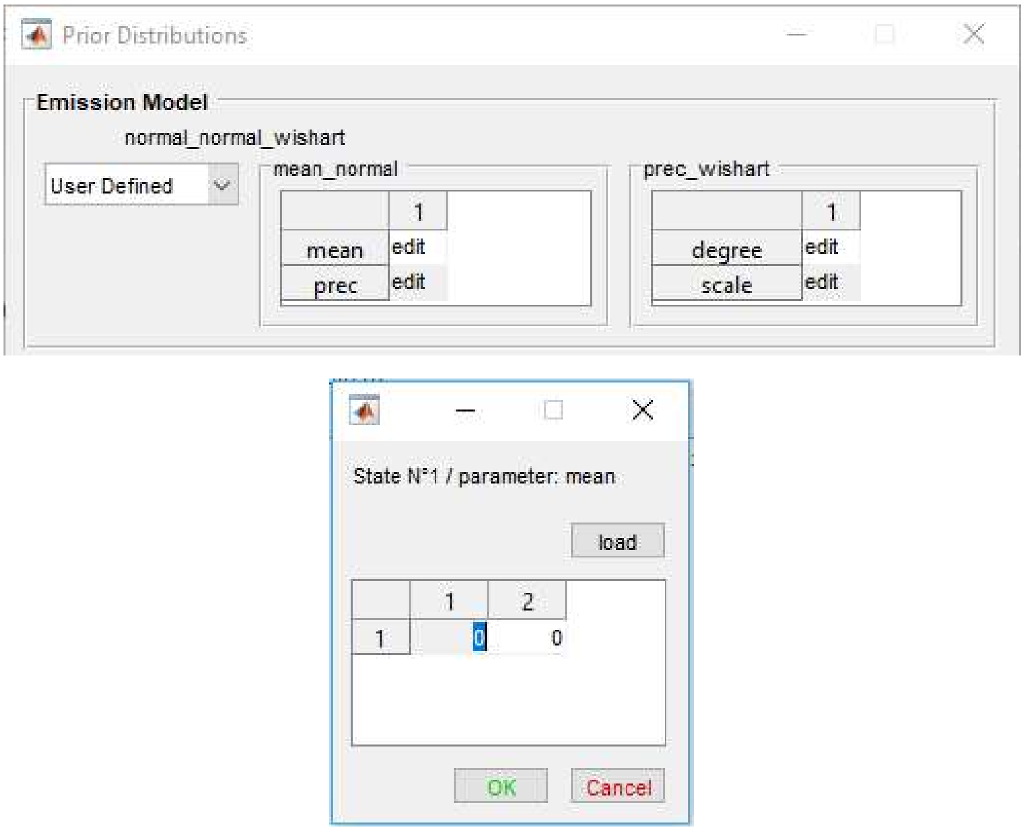


Figure 5.3: Example of manual emission model parameter specification.

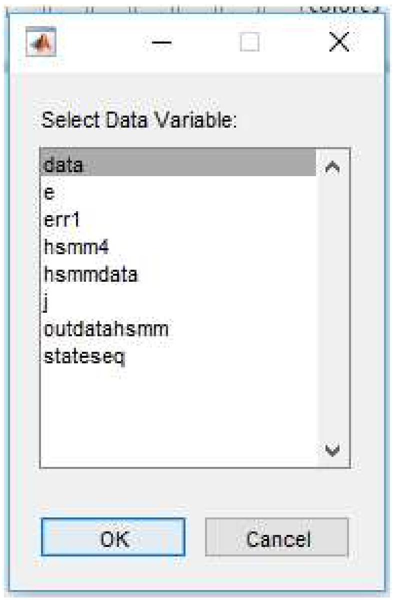


Figure 5.4: Loading prior parameters values as variables from Matlab workspace

5.3 Viewing the Posterior Distributions

After train the model is estimated using the data, the model parameters can be displayed by selecting the "Posterior Distribution" option.

Model -> Posterior Distribution (See **Error! Reference source not found.**)

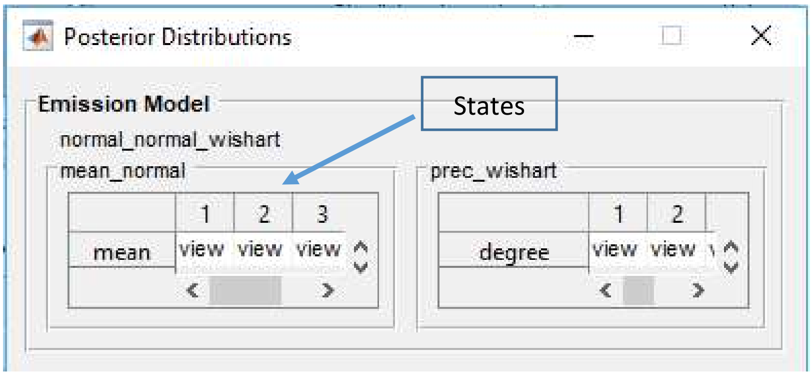


Figure 5.5**:** Displaying the estimated model parameters. In the above case the mean and precision matrix of each state can be displayed by pressing the respective view button

6 Inference

The Inference menu allows the following actions: Train Model, Decode, Estimate Parameters and Generate Data. The requirements and outputs from each action are shown in **Error! Reference source not found.**

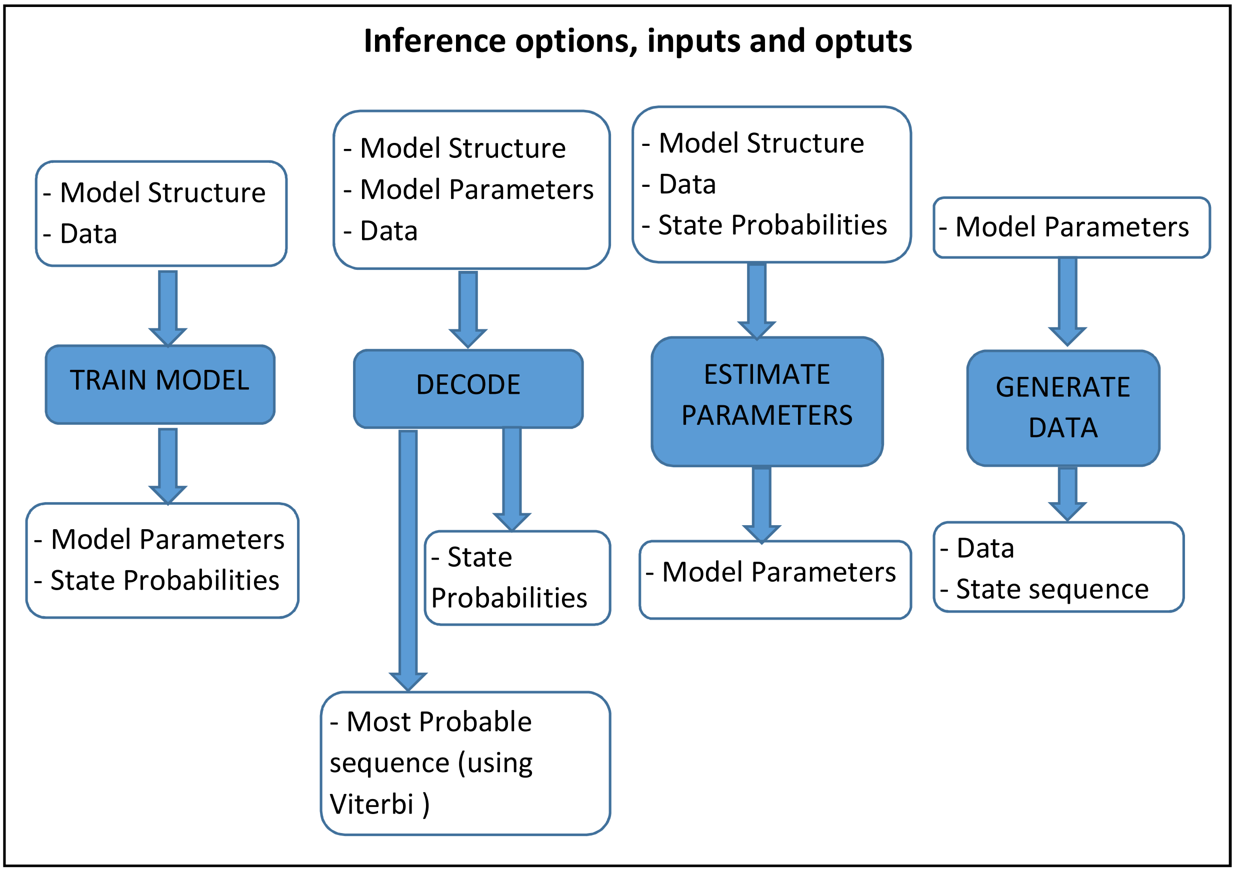


Figure 6.1: Scheme about Inference Options

<u>Typical Examples of Infrence</u>

1.- Train Model: Model parameters are estimated by fitting the measurement (observations) to a proposed model structure. Most studies would typically use this as the first inference step.

2.- Decode (State Probability): This option is used when the model parameter are given or have already been estimated to identify the hidden state of a new sequence data, different from the training data. The most probable hidden state sequence can also be estimated using the Viterbi option.

3.- Estimate Parameters: This option is used if the state probabilities sequence is given. It results in model parameters that characterize this sequence.

4.- Generate Data: This option is used to generate or simulate from a known model parameters are known, either because the model had been trained or because the parameters have been manually defined by the user.

6.1 Train Model [state probabilities unknown]

Train model uses a specified model, its prior parameters and the measurement data to estimate the posterior distributions of the parameters. (See **Error! Reference source not found.**)

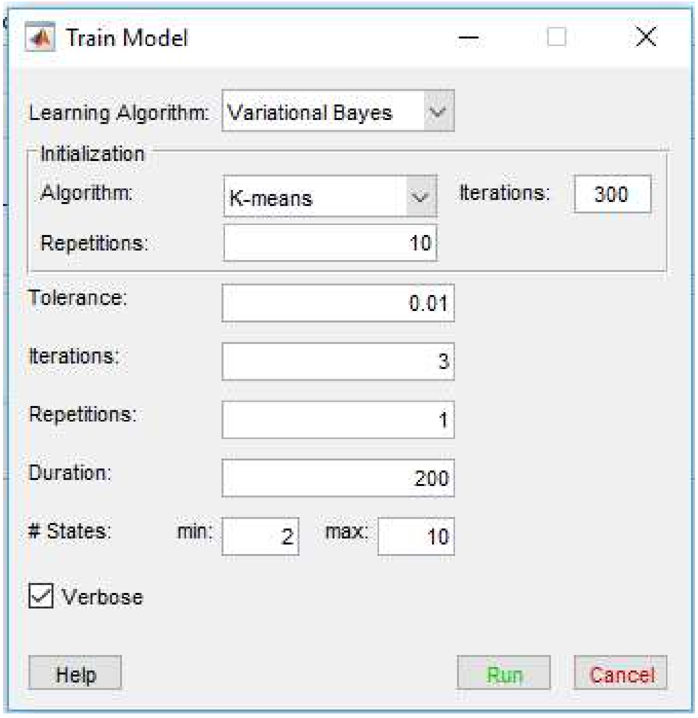


Figure 6.2: Train model options

Table 6.1*:* Train model options

~~N~~	Definition	Description	Selectable Options
~~1~~	Learning Algorithm	Select Learning Algorithm	- Variational Bayes
~~2~~	Initialization: Algorithm	Method of initializing State Probabilities	- K-means: Clustering method - Random initialization
~~3~~	Initialization: Iterations	Maximum number of iteration of Initialization Algorithm	Integer number
~~4~~	Initialization: Repetitions	Repetitions number of Initialization Algorithm (Applies only to K-means)	Integer number
~~5~~	Tolerance	Convergence criterion. Minimum percentage difference between iteration that define convergence	0% to 100%
~~6~~	Iterations	Maximum number of iteration of Learning Algorithm	Integer number
~~7~~	Repetitions	Repetitions number of Learning Algorithm	Integer number
~~8~~	Durations	Maximum State Duration	Integer number
~~9~~	# States min / max	Search Range of Number of	Integer numbers
~~10~~	Verbose	Display of free energy values	True or False

6.2 Decode Model

The Decode Model option allows for two actions:

Viterbi Decoding: Estimate the Most Probable Sequence

State Probabilities: Estimate the probabilities: Estimate the probability that an emission has been generated by the emission model of a state

6.3 Estimate Parameters [state probabilities known]

This option does not require any user input

6.4 Generate Data

The generate data option simulates new data from a given model (Figure 6.3):

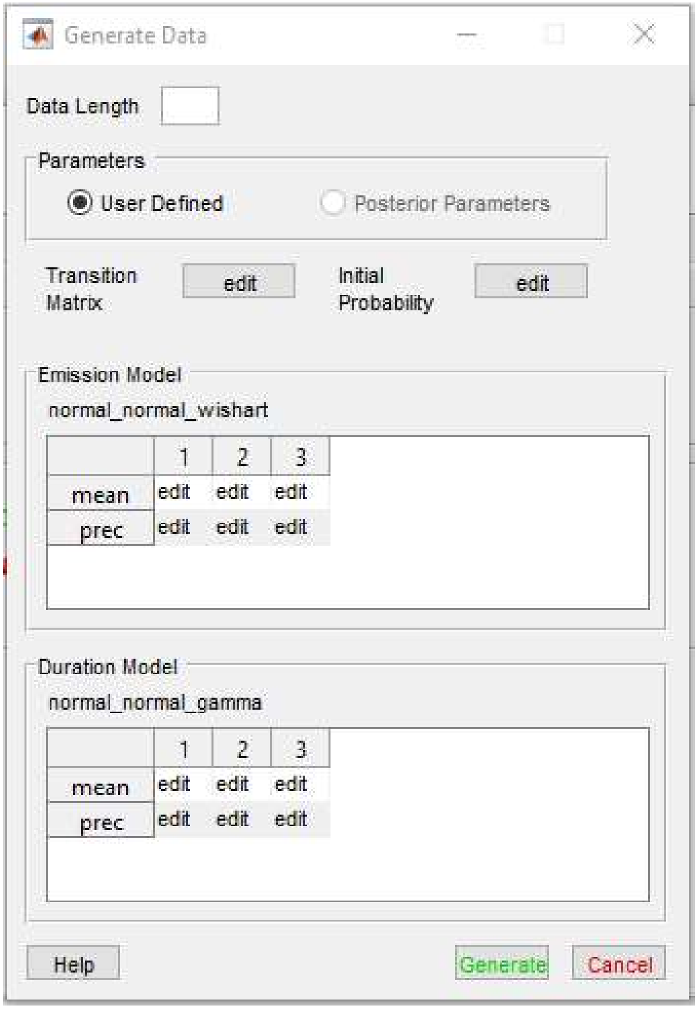


Figure 6.3: Editing model parameter before generating data

Table 6.2: Option to generate data from a given model

N	Options	Description	Selectable Options
1	Data Length	Length of data generated using model	Integer number from 1 to 1e9
2	Parameters	Indicate what are the model parameters used for generate	-User Defined: Parameters has been defined by used - Posterior Parameters had been previously estimated estimated
3	Transition Matrix	Define Transition Matrix (User Defined option)	Values between 0 and 1. The rows have to add to one.
4	Initial Probabilities	Probability of each state at the first time point of the sequence	Values between 0 and 1 and adding up to 1.

N	Options	Description	Selectable Options
5	Emission Model	Define the Emission Model Parameters	Real numbers
6	Duration Model	Define the Duration Model Parameters	Real numbers

7 Views

The Views menu give to user different display options of the results of the inference actions. Depending on the Inference action, a different set of display options are available:

- Train Model
  - Free Energy Plot
  - Topography
  - Emission Covariance Matrix
  - State Probabilities
  - Duration Posterior distribution
- Decode (Viterbi)
  - State Sequence
  - Proportion of State Occupancy
  - Duration Histogram
- Generate
  - State Sequence
  - Proportion of State Occupancy
  - Duration Histogram
  - Topography
  - Emission Covariance Matrix

The following figures are example of the plots that can be obtained

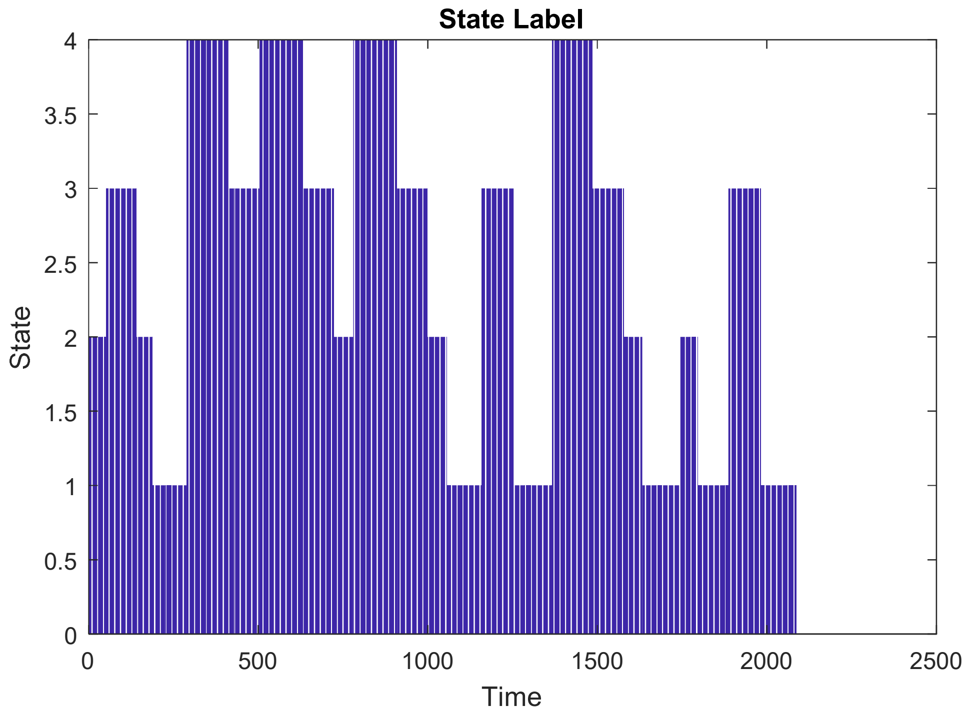


Figure 7.1: State squence

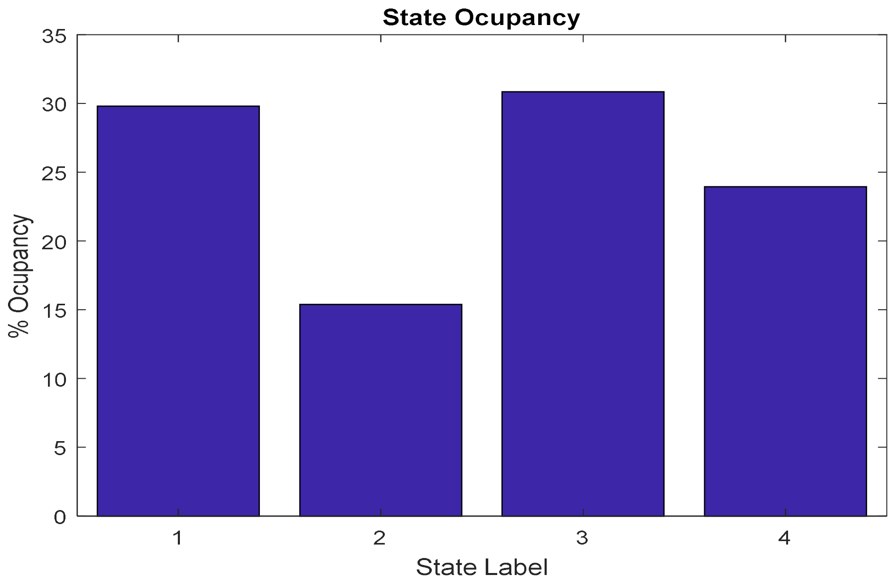


Figure 7.2: Proportional state occupancy

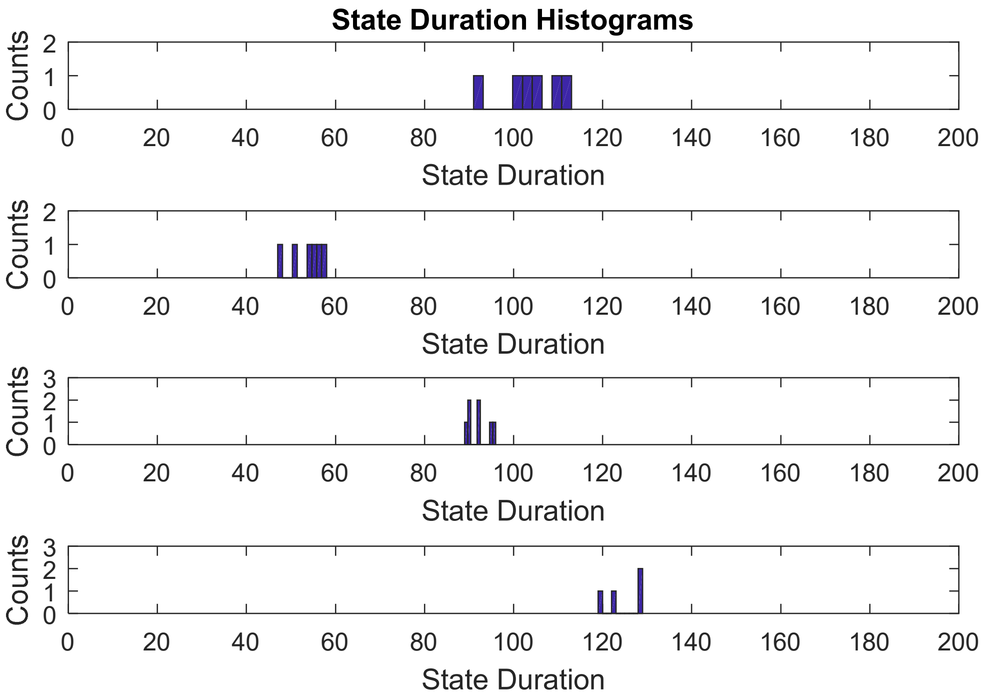


Figure 7.3: State durational histograms

Figure 7.4: State topography

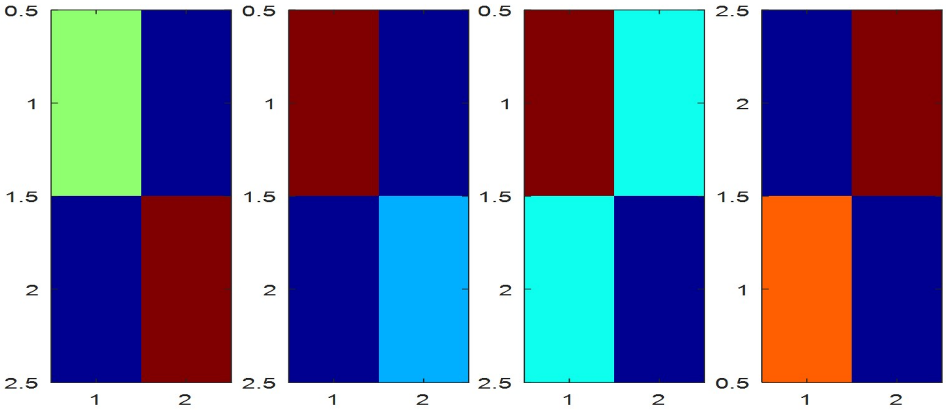


Figure 7.5**:** Emission Covariance Matrix.

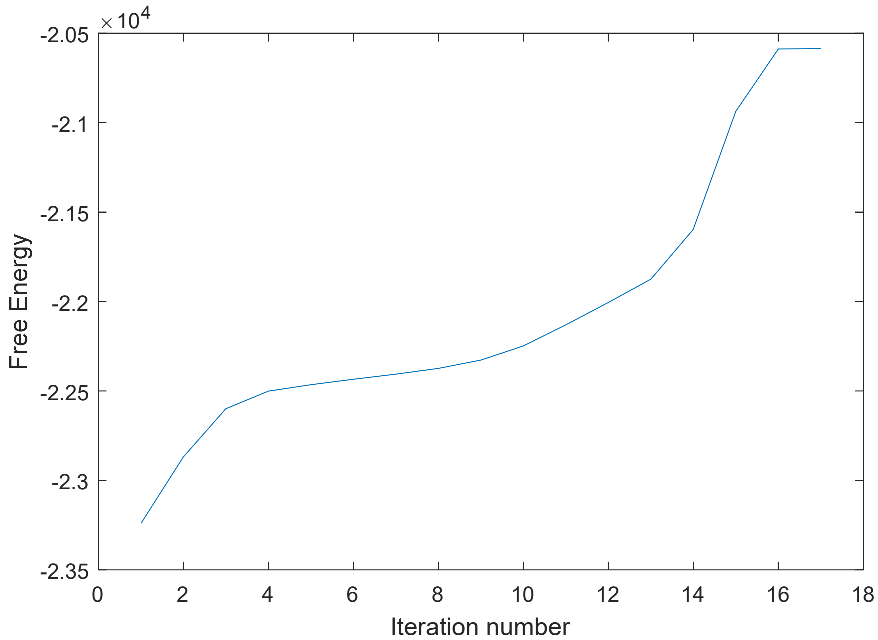


Figure 7.6**:** Free Energy convergence curve

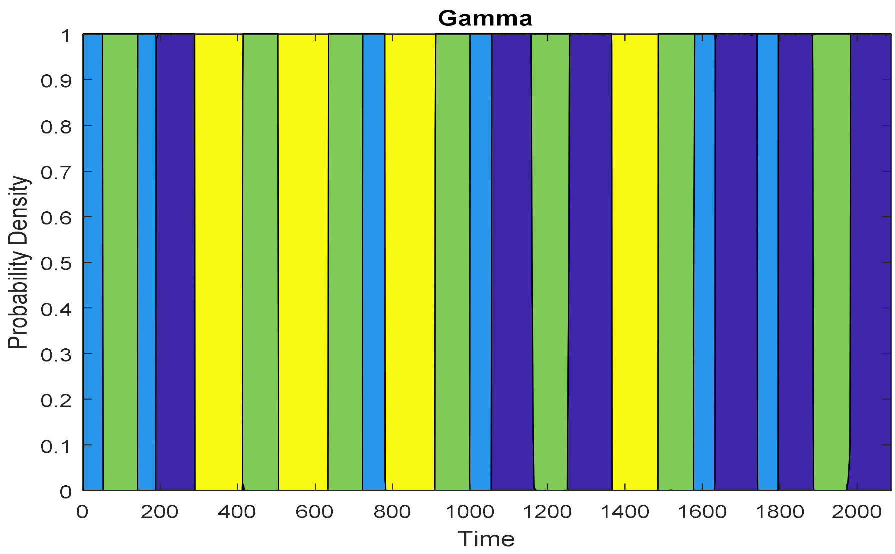


Figure 7.7: State Probabilities

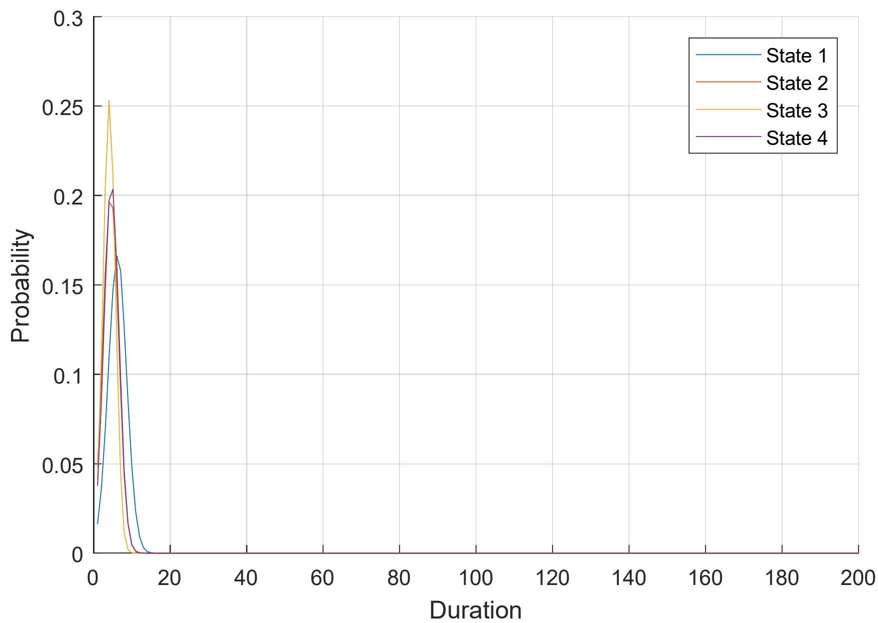


Figure 7.8: State duration Posterior distribution

8 Study Cases

8.1 Case 1: Two Dimension Data

This is a toy example to demonstrate the BSD functionality. The data is two dimensional and has four states. The state duration is around 100 ms (**Error! Reference source not found.** and **Error! Reference source not found.**).

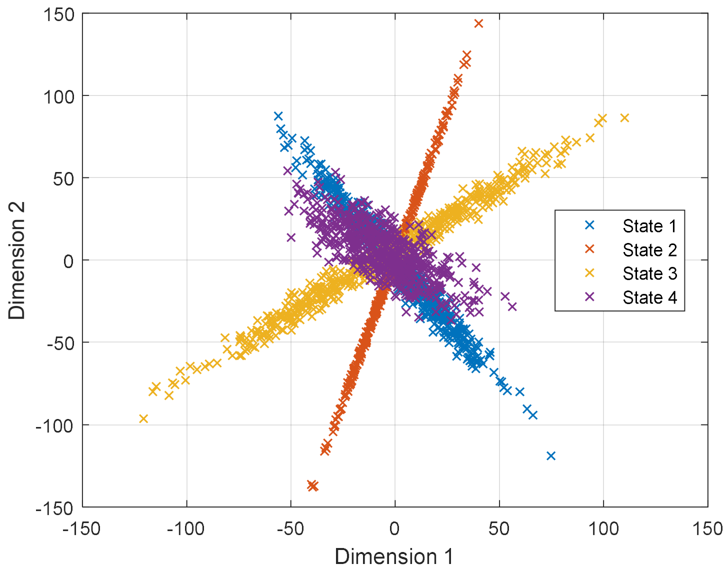


Figure 8.1: Case 1 Data Plot. There are 4 states, with duration of 100ms

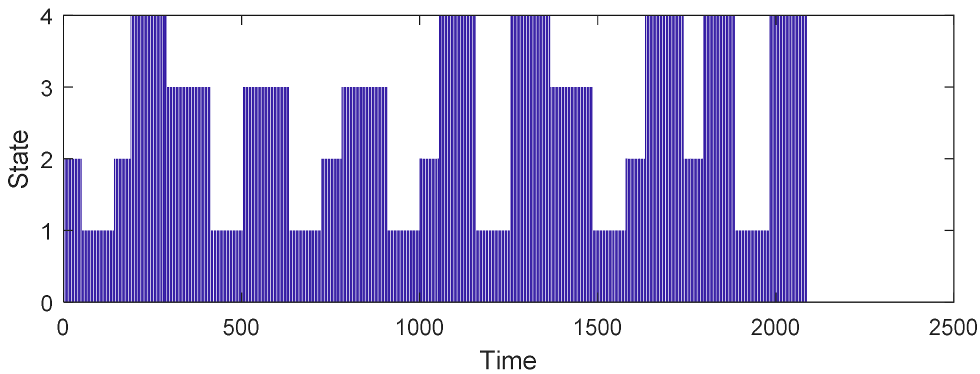


Figure 8.2: Case 1, actual State Sequence

<u>Load Data</u>

Data -> Import Data -> Matlab File

Open file “Example1_data2dim.mat” located in folder “Examples”

Select the variable “Data” and choose a name like “My data1”

Create Model

Model -> Create Model

Use the followings parameters in pop-up

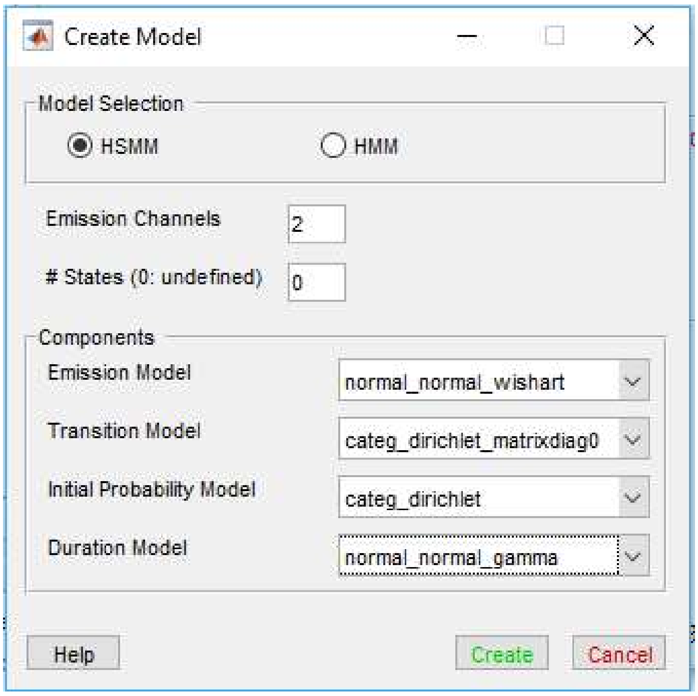


Figure 8.3: Create Model Window

Press Create

Prior Distribution

Use Non-Informative Prior (default)

Train Model

Inference -> Train Model

Use the followings parameters in pop-up window

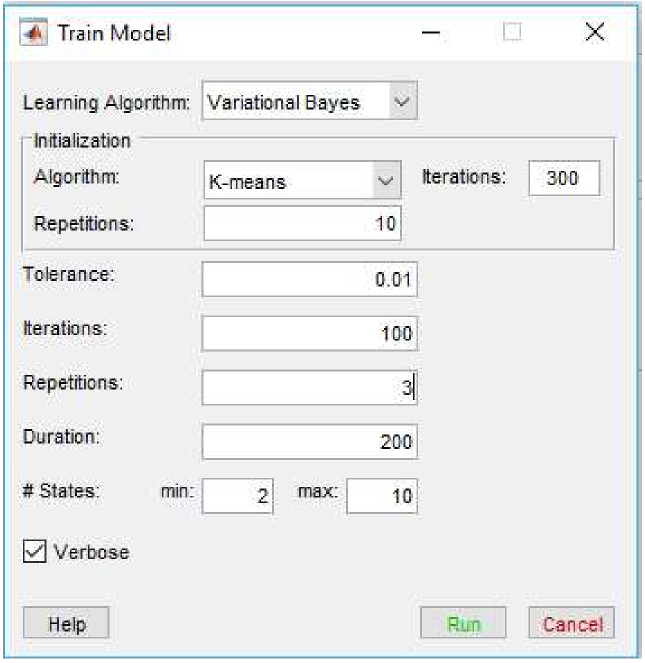


Figure 8.4: Train Model Window

Press Run

If we want to see the state estimating error:

```
load Examples/Example1_data2dim.mat;
s1= hSmm.study.viterbi_results;
s2=util.renameseq(stateseq,s1);
sum(s2~=stateseq)/length(s2)*100
```

8.2 Case 2: EEG Topography

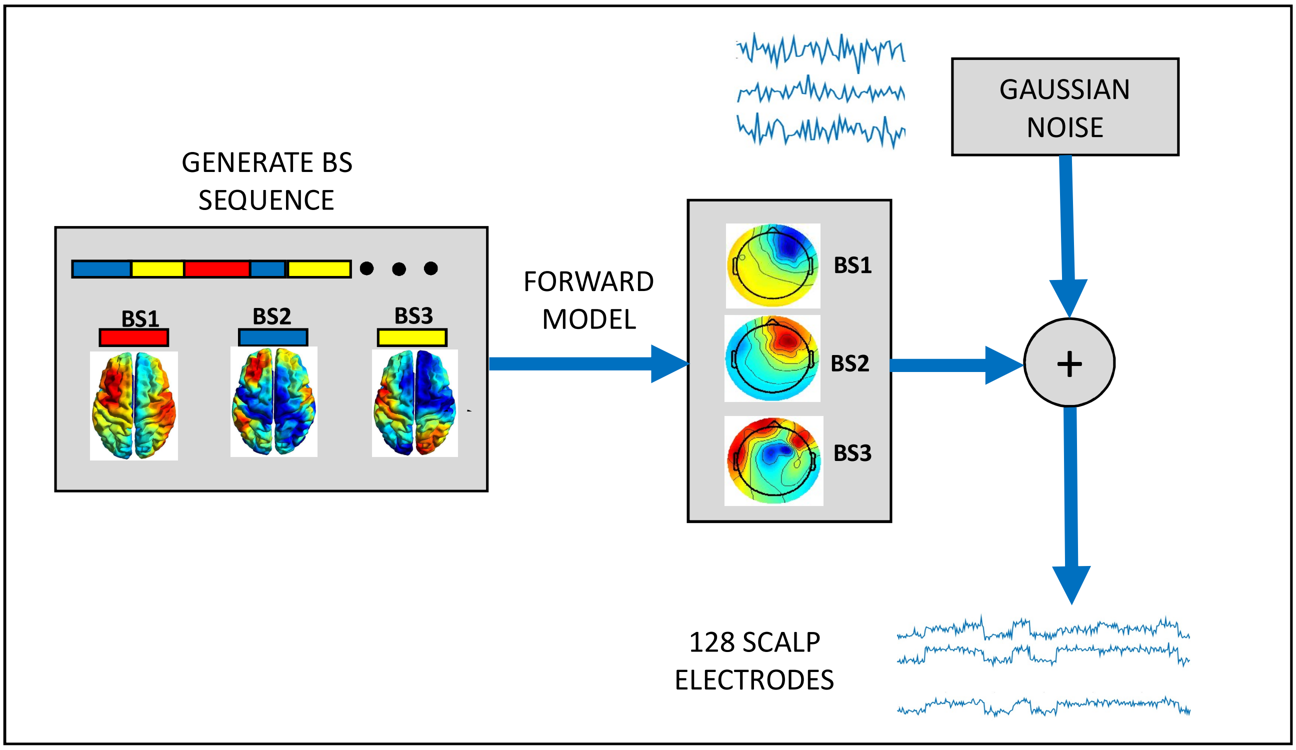


Figure 8.5: Pipeline data generation

Load Data

Data -> Import Data -> Matlab File

Open file "Example2_topography.mat" located in folder "Examples"

Select the variable "Data" and choose a name like "My data2"

Create Model

Model -> Create Model

Use the followings parameters in pop-up

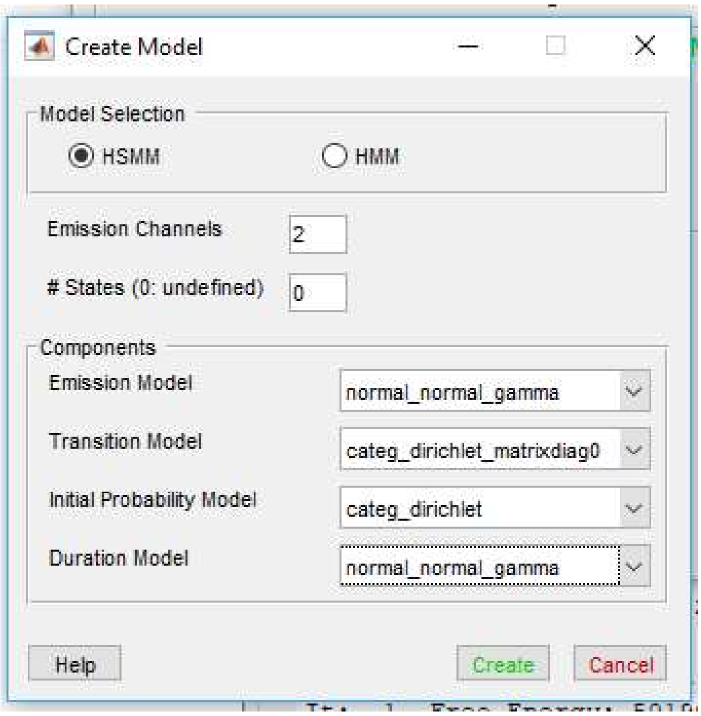


Figure 8.6: Create Model Windows

Press Create

<u>Prior Distribution</u>

Use Non-Informative Prior (do not change)

<u>Train Model</u>

Inference -> Train Model

Use the followings parameters in pop-up

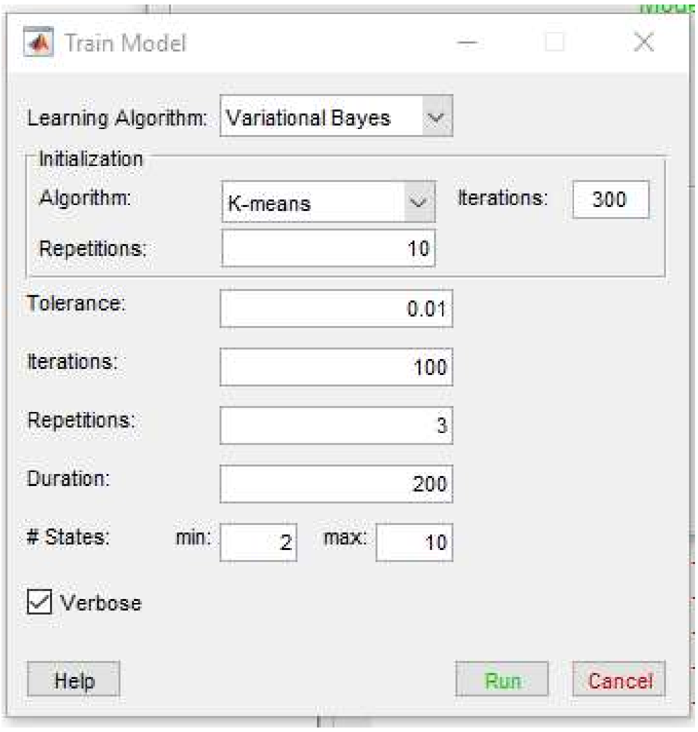


Figure 8.7: Train Model Window

Press Run

Case 3: Multi-Blocks and Multi-Condition

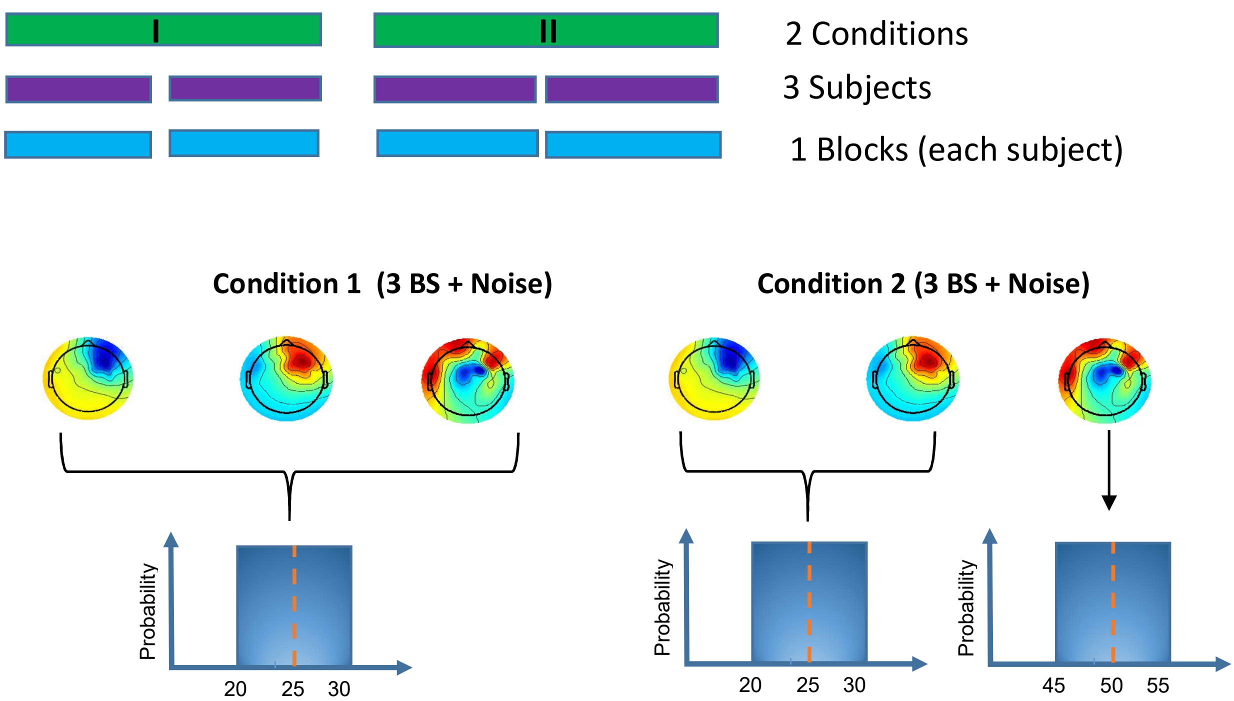


Figure 8.8: Pipeline data generation

Load Data

Data -> Import Data -> Matlab File

Open file “Example3_Multiblock.mat” located in folder “Examples”

Create Model

Model -> Create Model

Use the followings parameters in pop-up

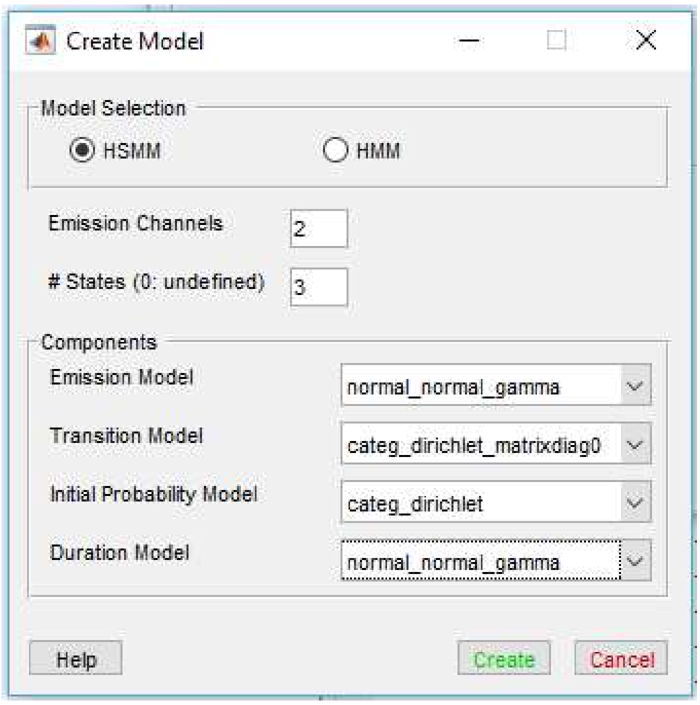


Figure 8.9: Create Model Window

Press Create

<u>Prior Distribution</u>

Use Non-Informative Prior (do not change)

<u>Train Model</u>

Inference -> Train Model

Use the followings parameters in pop-up

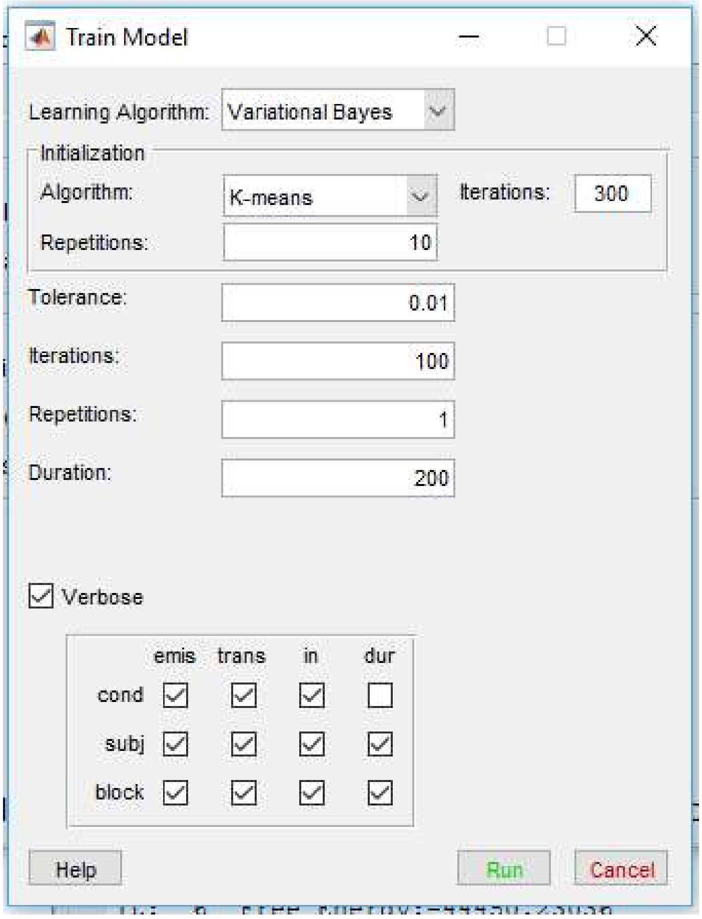


Figure 8.10: Train Model Window

Press Run

8.3 Case 4: Source Connectivity

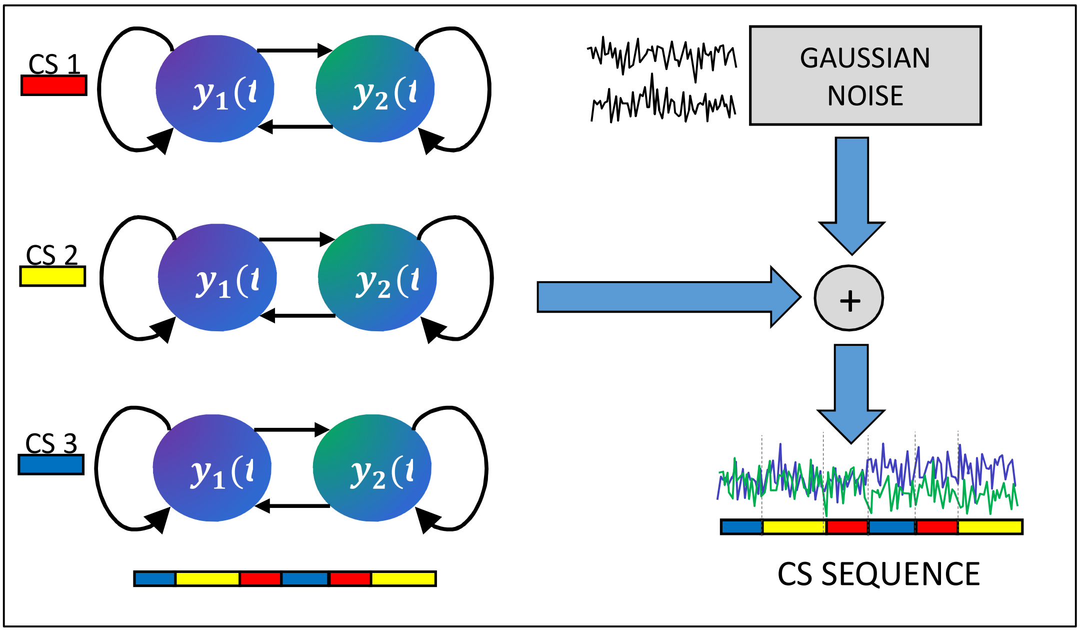


Figure 0.1: Pipeline data generation

Load Data

Data -> Import Data -> Matlab File

Open file “Example3_Multiblock.mat” located in folder “Examples”

Create Model

Model -> Create Model

Use the followings parameters in pop-up

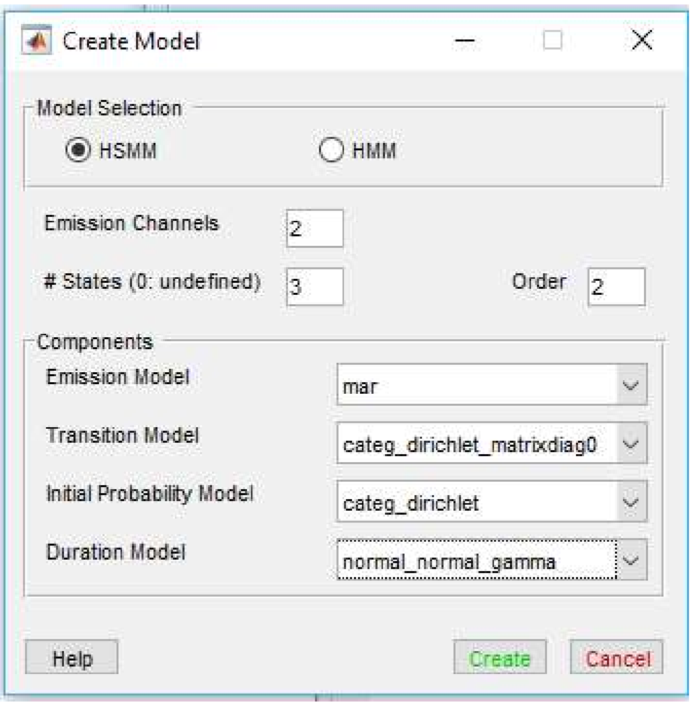


Figure 0.2: Create Model Window

Press Create

<u>Prior Distribution</u>

Use Non-Informative Prior (do not change)

<u>Train Model</u>

Inference -> Train Model

Use the followings parameters in pop-up

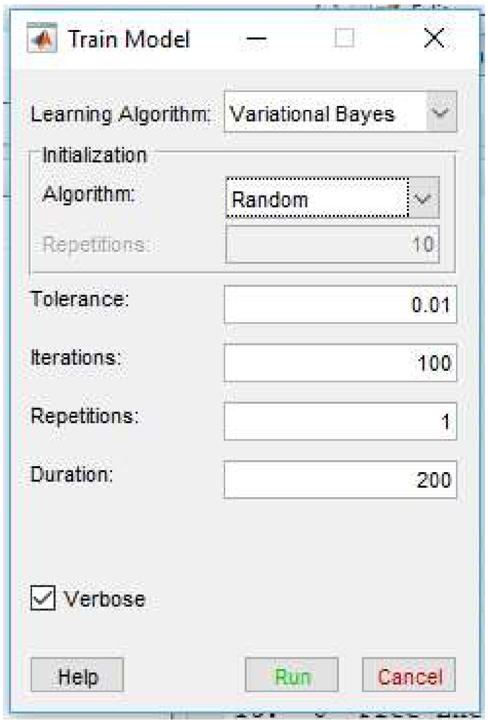
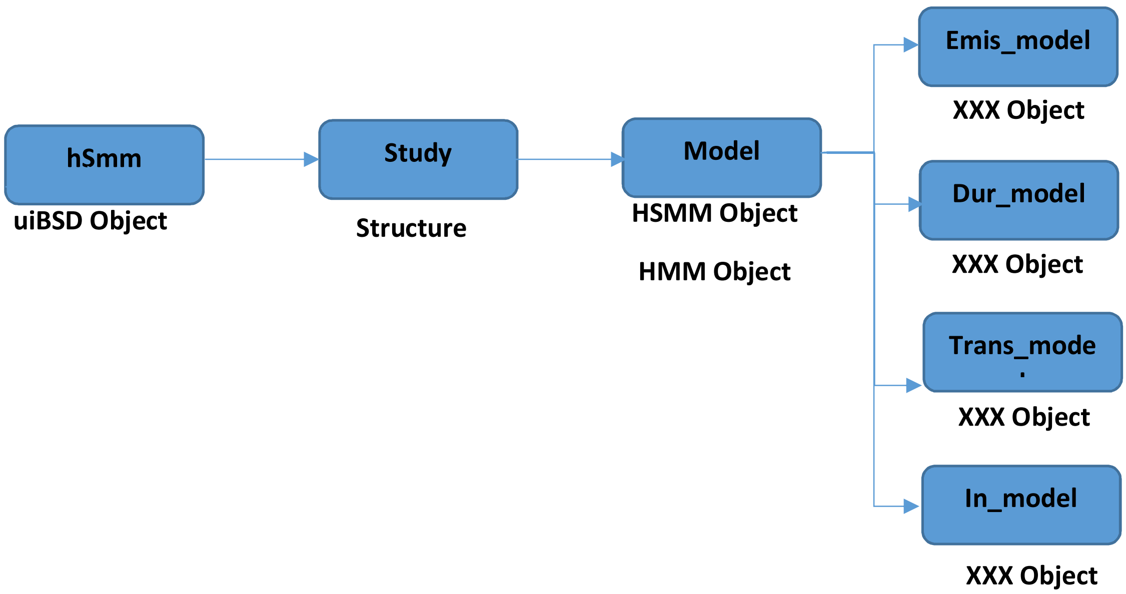
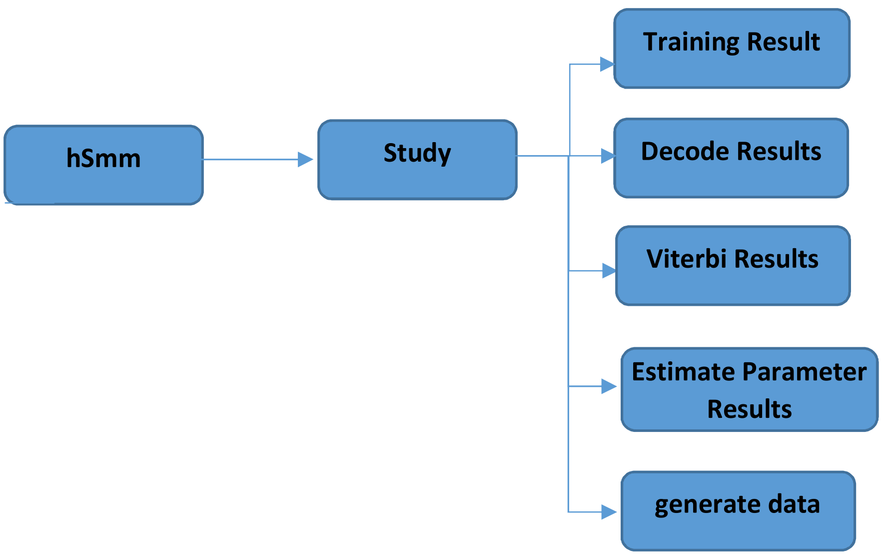


Figure 0.3: Train Model Window

APPENDIX 1: MODEL AND DATA STRUCTURE


hSmm
uiBSD Object
Study
Structure
Model
HSMM Object
HMM Object
Emis_model
XXX Object
Dur_model
XXX Object
Trans_mode
.
XXX Object
In_model
XXX Object


hSmm
Study
Training Result
Decode Results
Viterbi Results
Estimate Parameter Results
generate data

uiBSD object

Variable	Type
gui:	struct
study:	struct
default_eeg:	array
is_saved:	boolean
ws_filename:	string

Study Structure

Variable	Type
data:	double
metadata:	struct
generated_data:	double array
generated_metadata:	struct
viterbi_results:	integer array
viterbi_metadata:	struct
training_results:	struct
training_metadata:	struct
decode_results:	struct
decode_metadata:	struct
model:	Hsmm object
opt:	struct

HSMM Object

Variable	Type
ndim	real
nstates	real
in_model	obj
trans_model	obj
emis_model	obj
dur_model	obj

HSMM Component Object (emis_model, trans_model, in_model, dur_model)

Variable	Type
prior	struct
posterior	struct
parsampl	struct
divkl	real
ndim	real
nstates	real

Normal_normal_gamma Object (emis_model)

Prior

Variable	Type
mean_normal	struct
prec_gamma	struct

Posterior

Variable	Type
mean_normal	struct
prec_gamma	struct

mean_normal

Variable	Type
Mean	real array
Prec	real matrix

prec_gamma

Variable	Type
shape	real array
Scale	real array

Posterior

Variable	Type
mean_normal	struct
prec_gamma	struct

parsampl

Variable	Type
mean	real array
prec	real matrix