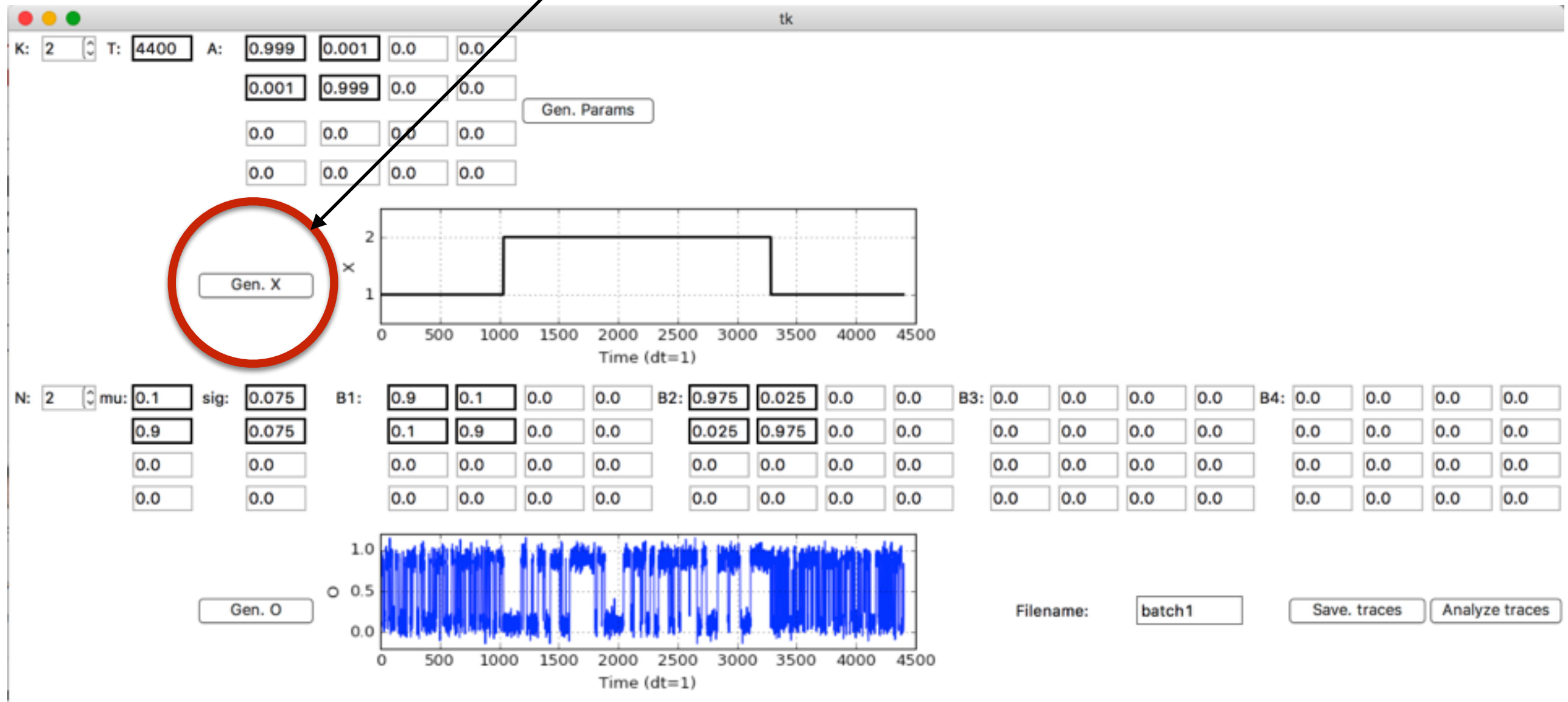


How to use VBDCMM

# **1. VBDCMM\_gui\_simul.py (short version)**

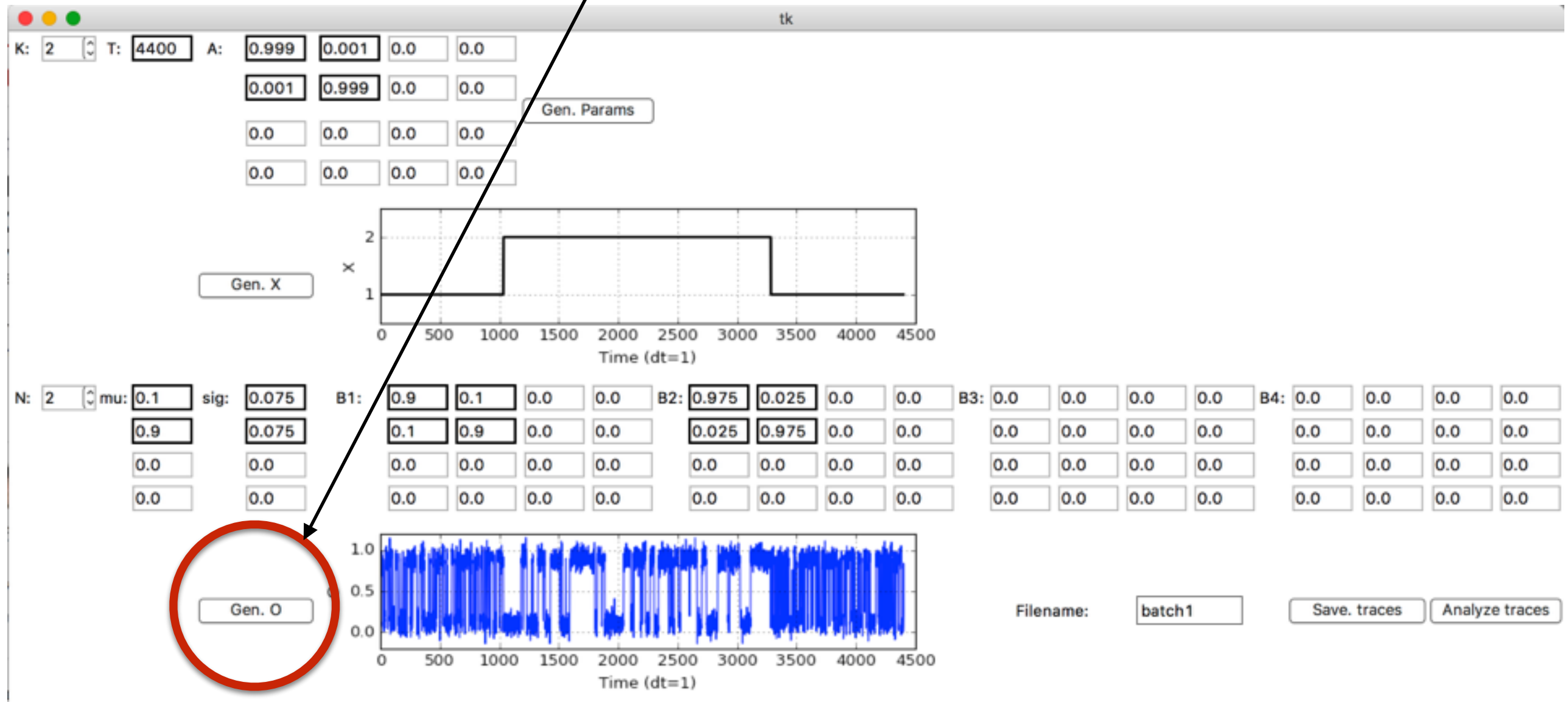
- (1) Open terminal.
- (2) Move to the folder where VBDCMM source files are downloaded),
- (3) Type: `python3 VBDCMM_gui_simul.py`

# Click this



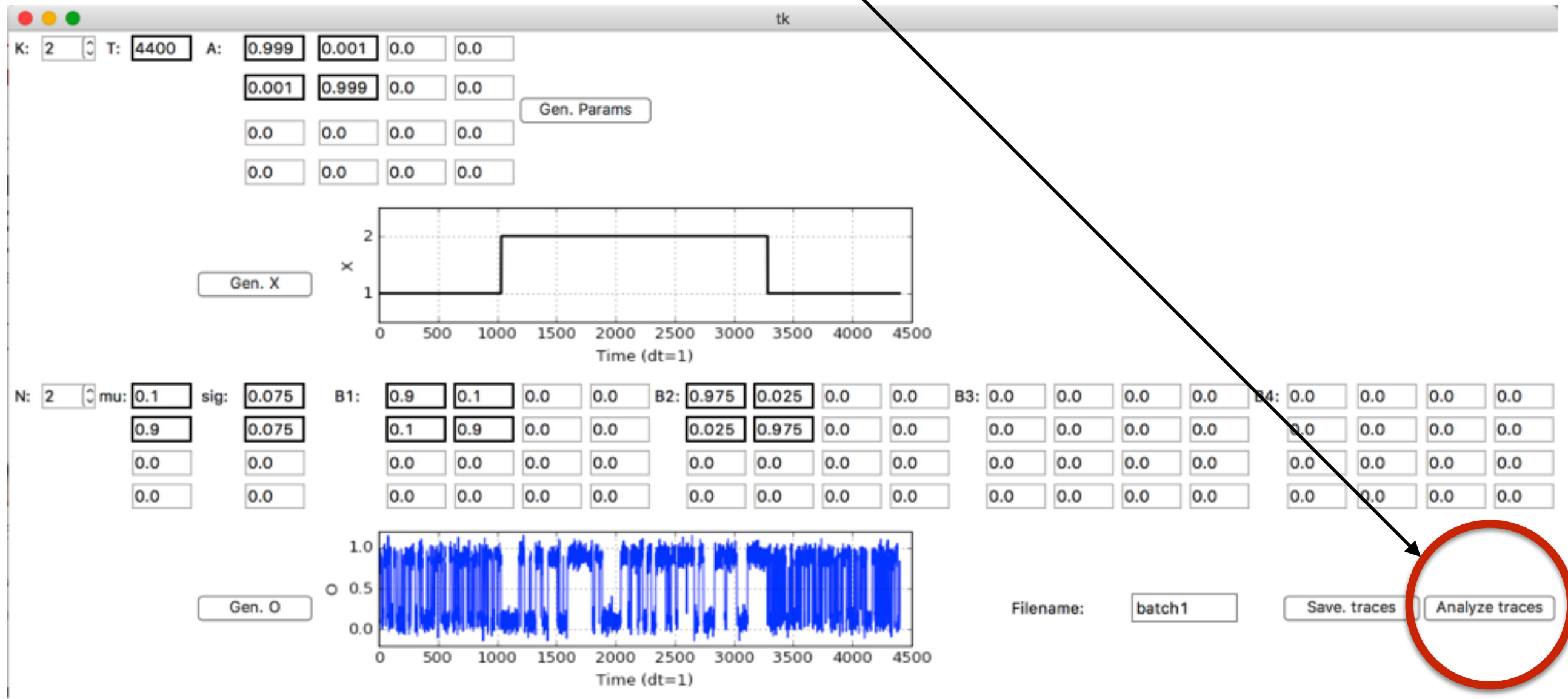
(to generate the sequence of (hidden) internal states. Repeat as much as you like)

# Click this



(to generate the sequence of observable states. Repeat as much as you like)

# Click this



(to start VBDCMM analysis)

# Click this

tk #4

N:  mu:  sig:  B:     mu\_esti:  sig\_esti:

Time (dt=1)

K min:  K max:  gamma:  n\_repeats:  F methods:

Time (dt=1)

Evidence

K

Best K:

A:     B1:     B2:

B3:     B4:

Filename:

(to start filter the noise)

# Click this

tk #4

N:  mu:  sig:  B:     mu\_esti:  sig\_esti:

HMM!

K min:  K max:  gamma:  n\_repeats:  F methods:

VB-DCMM!

x

Time (dt=1)

Evidence

K

Best K:  A:

B1:

B2:

B3:

B4:

Filename:

Save

(to find most probable (hidden) sequence of internal state & best model)



## **2. VBDCMM\_gui\_simul.py (longer version)**

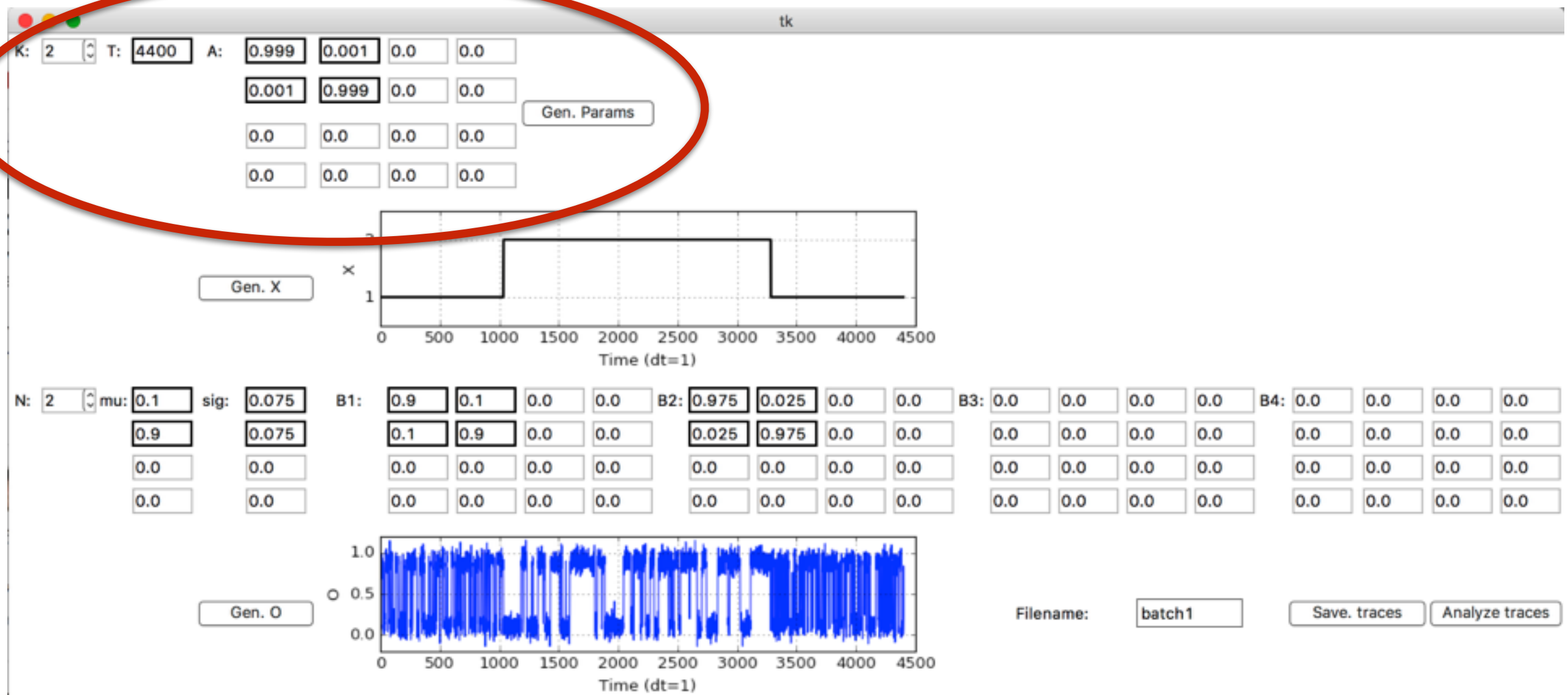
# 1. Set parameters for the generation of sequence of internal states

K: number of internal states

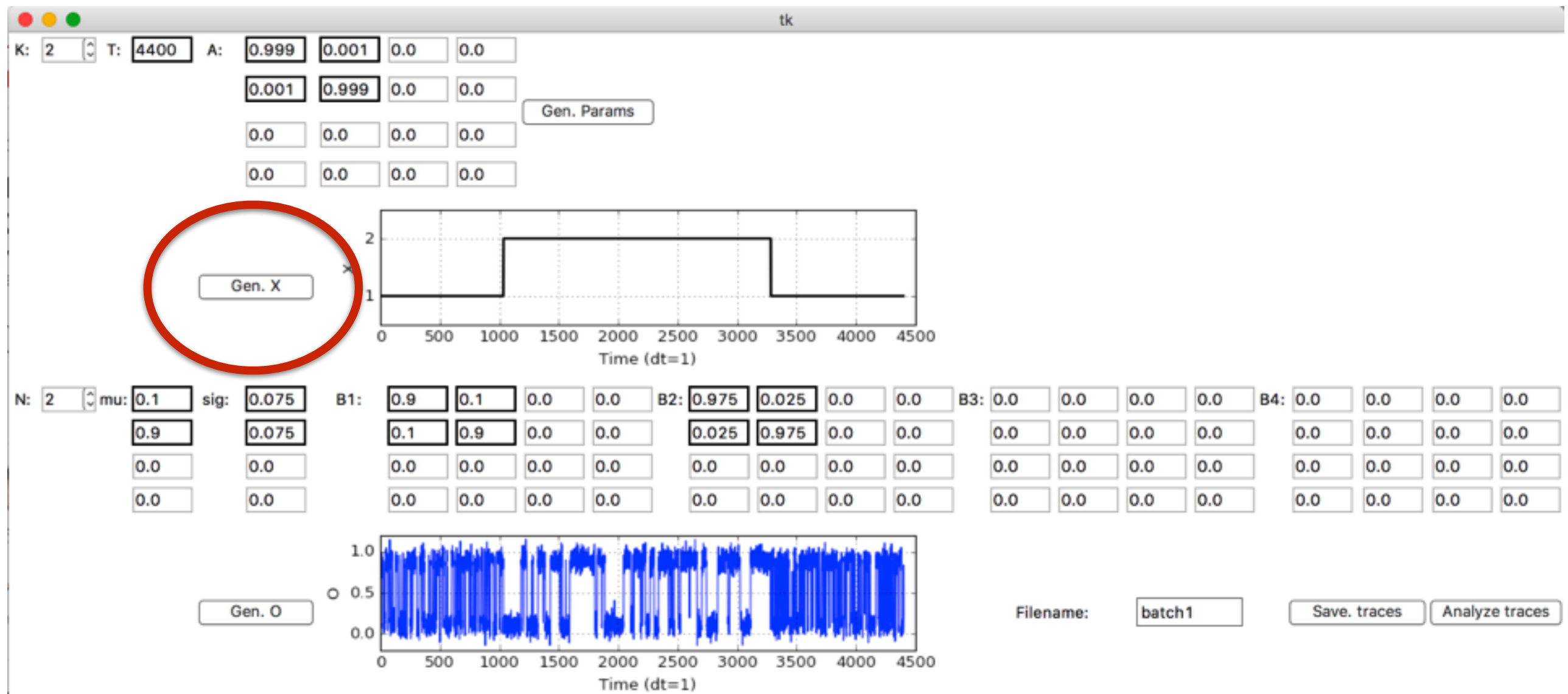
T: Observation time

A: Transition matrix

Gen. Params: Set parameters automatically.



## 2. Generate a trace of internal state



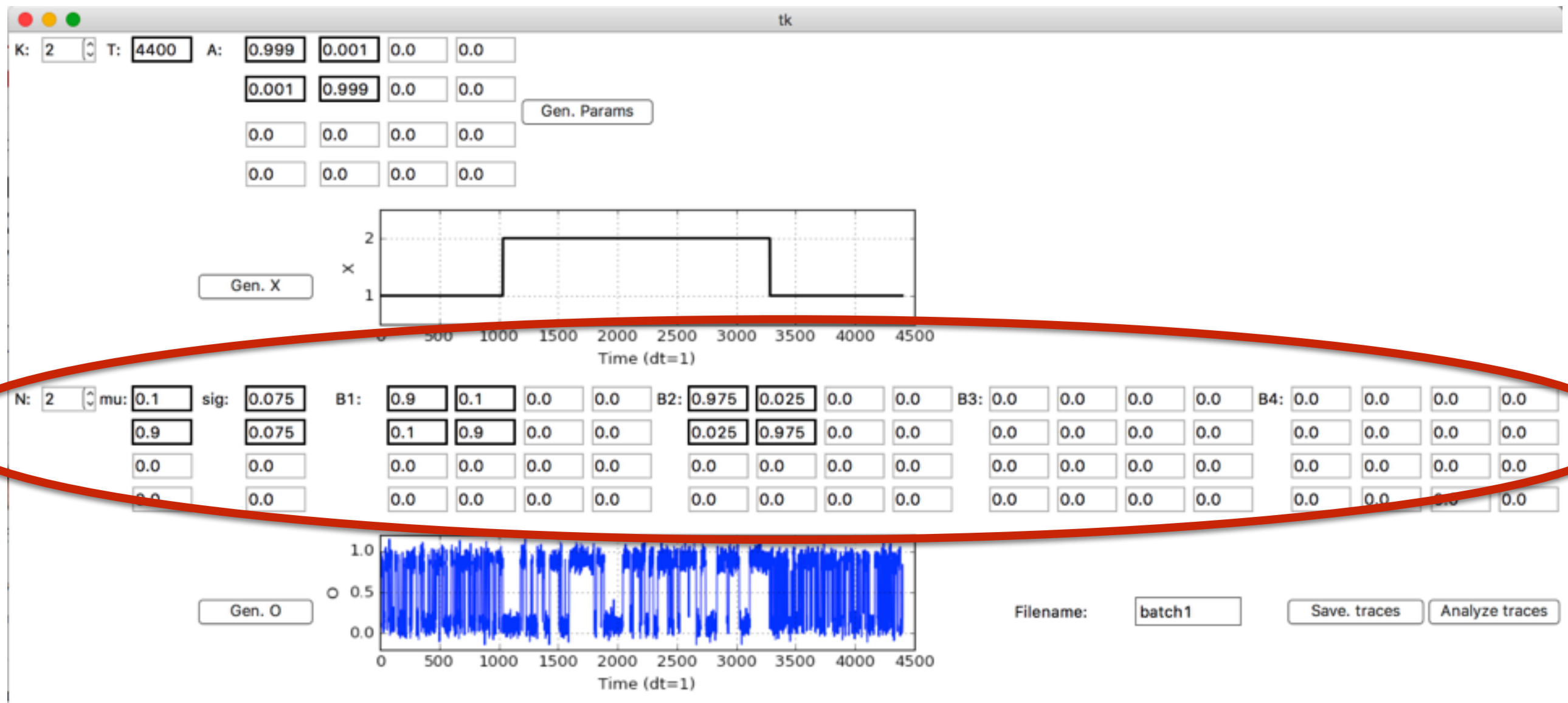
### 3. Set parameters for generation of sequence of observable states

N: The number of observable states

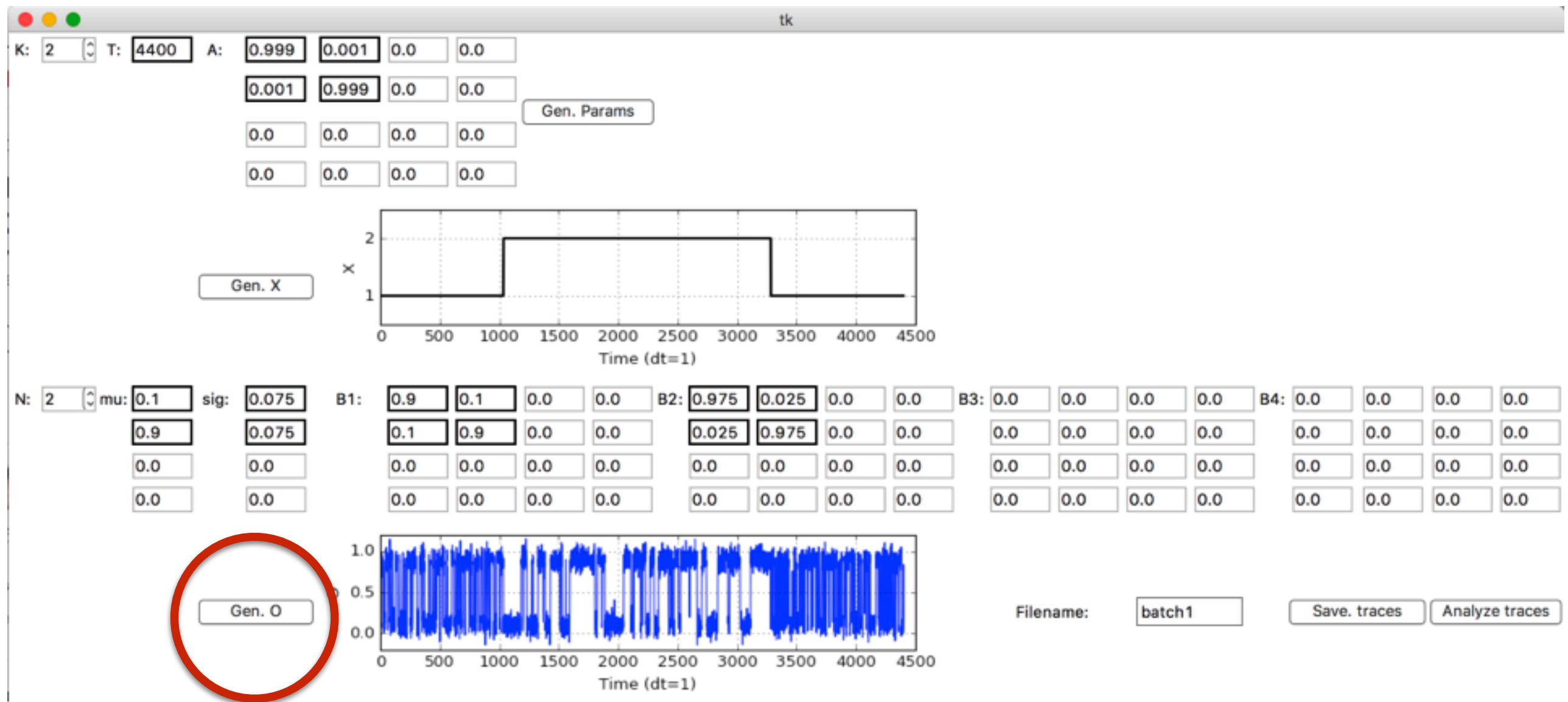
mu: Mean value of each observable state

sig: Std of each observable state

B1—4: Transition matrix of observable state corresponding to internal state 1—4.



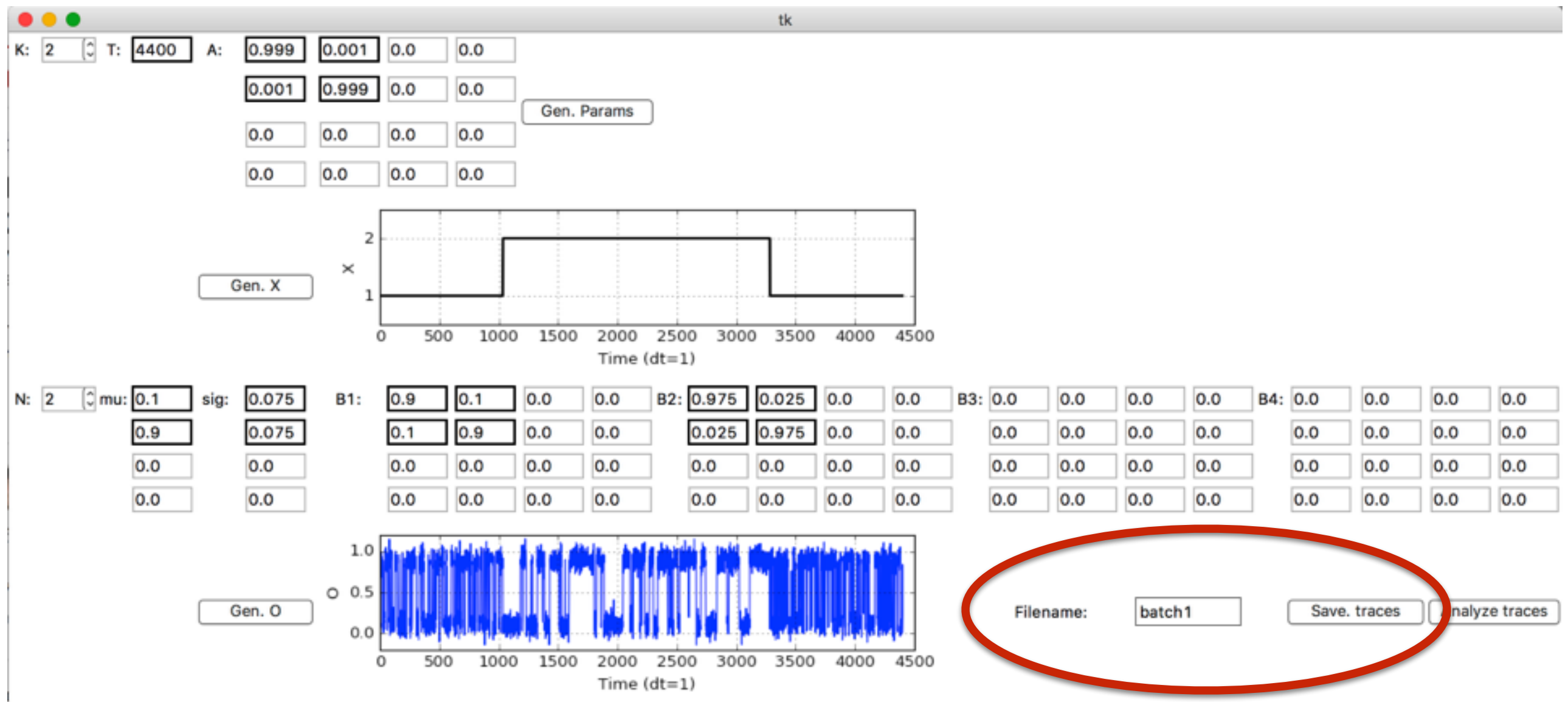
## 5. Gen. O: Generate the trace of observable state.



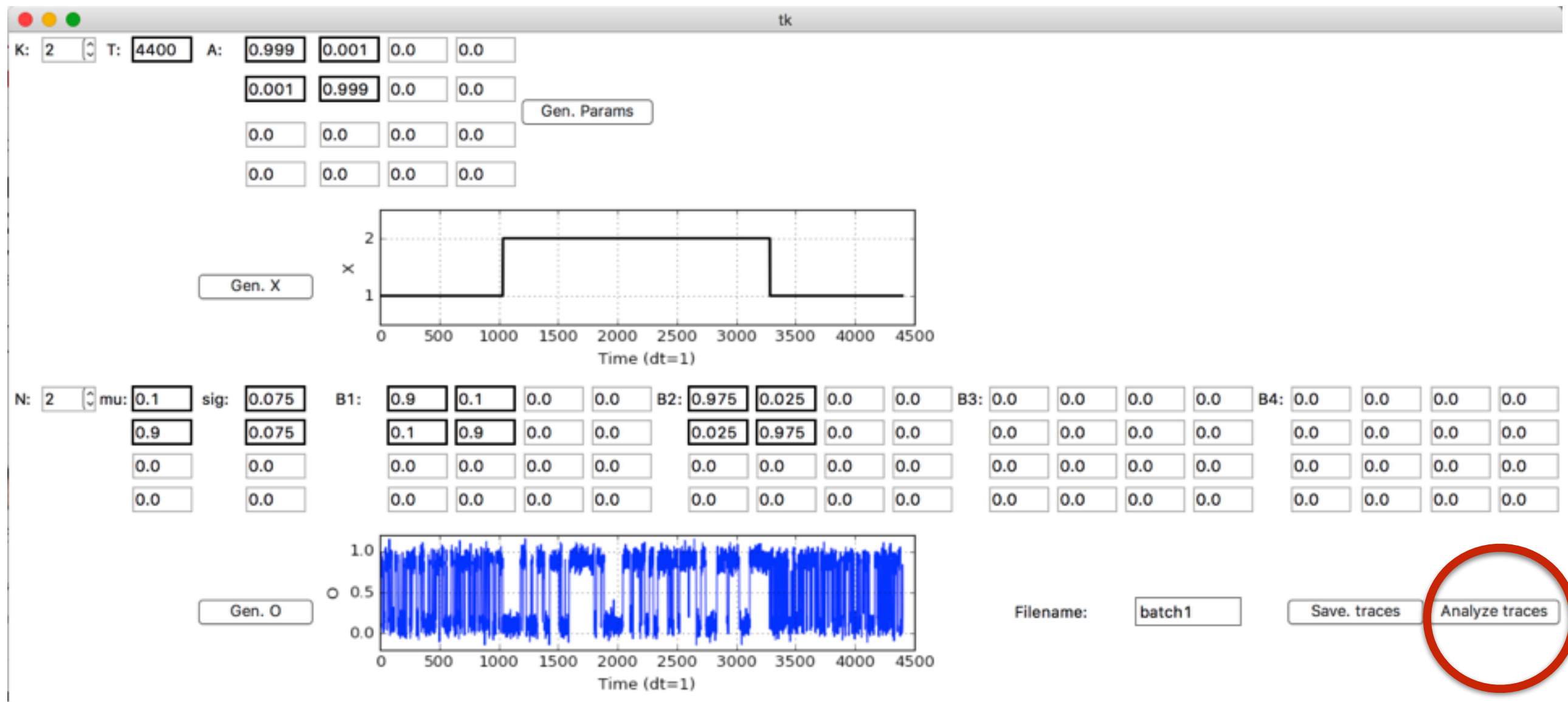
## 6. Save traces

Filename: The name of save-files.

Save. traces: Save data. Files will be saved in the same folder with “VBDCMM\_gui\_simul.py” file



## 7. Start VBDCMM





## 8. Filter the noise via Hidden Markov Modelling (HMM)

N: The number of (guessed) observable states.

mu: The initial guess of mean value of each observable state.

sig: The initial guess of std value of each observable state.

B: The initial guess of transition matrix.

N:

2

⬇

mu:

0.1

sig:

0.075

B:

0.99

0.01

0.0

0.0

0.01

0.99

0.0

0.0

0.0

0.0

0.0

0.0

m\_esti:

0.1033

sig\_esti:

0.0770

0.9299

0.0700

0

0

0.9007

0.0768

0.0553

0.9446

0

0

0

0

0

0

0

0

HMM!

tk #4

o

1.0

0.5

0.0

0

500

1000

1500

2000

2500

3000

3500

4000

4500

Time (dt=1)

K min:

1

⬇

K max:

2

⬇

gamma:

0.001

n\_repeats:

1

F methods:

Based on G

⬇

VB-DCMM!

x

2

1

0

500

1000

1500

2000

2500

3000

3500

4000

4500

Time (dt=1)

Evidence

-980

-1025

1

2

K

Best K:

2

A:

0.9994

0.0005

0

0

0.0004

0.9995

0

0

0

0

0

0

B1:

0.8950

0.1049

0

0

0.0949

0.9050

0

0

0

0

0

0

B2:

0.9687

0.0312

0

0

0.0214

0.9785

0

0

0

0

0

0

B3:

0

0

0

0

0

0

0

0

0

0

0

0

B4:

0

0

0

0

0

0

0

0

0

0

0

0

Filename:

batch1

Save



## 9. Result of Hidden Markov Modelling (HMM)

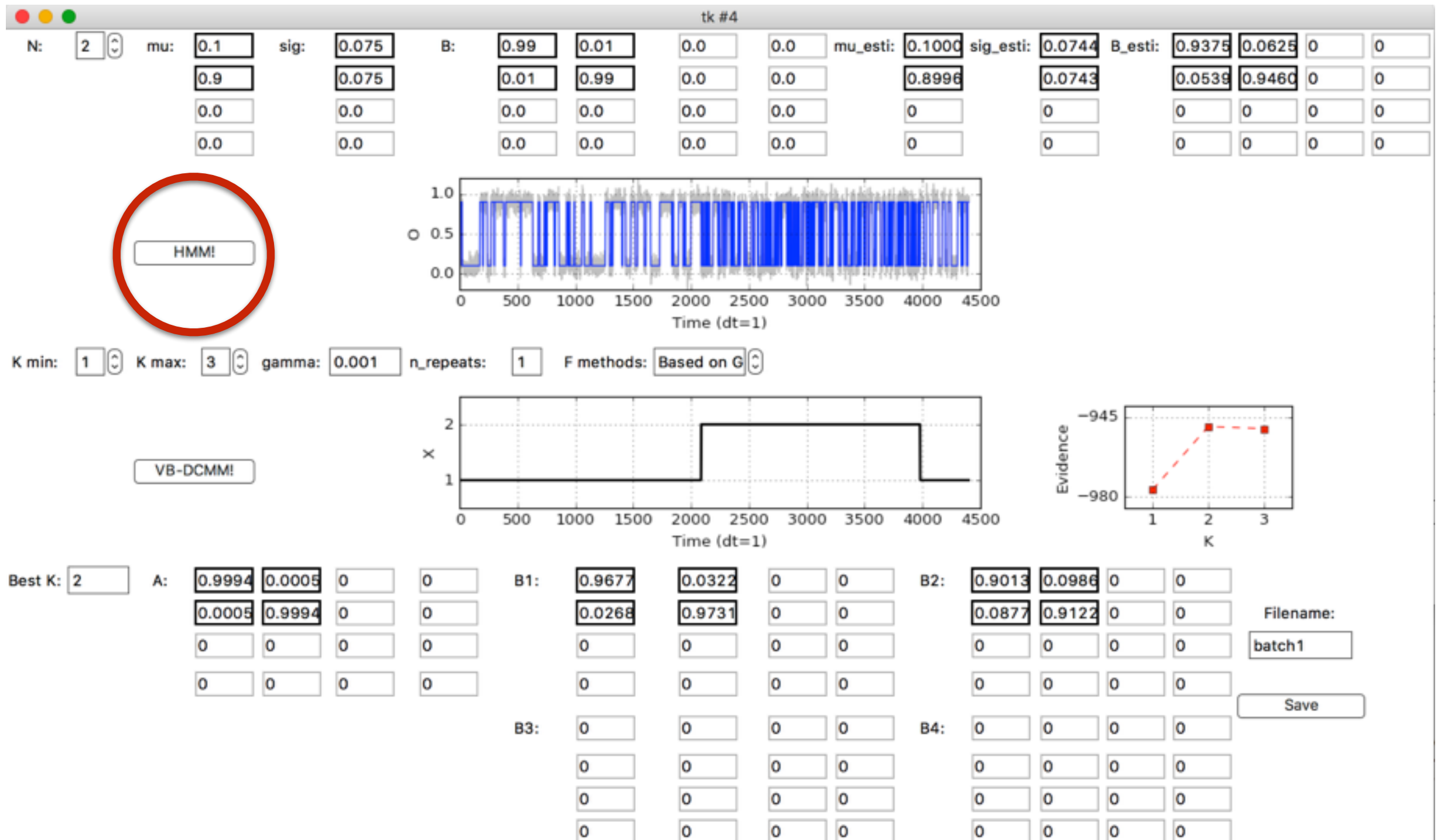
mu\_esti: The estimated mean value of each observable state.

sig\_esti: The estimated std value of each observable state.

B\_esti: The estimated transition matrix (assuming the presence of one internal state).



## 11. HMM analysis



## 12. Parameters for VB-DCMM analysis

K min: Minimum number of internal states

K max: Maximum number of internal states

gamma: Initial guess of transition rate of internal states. “0.001” means roughly 1 transition is observed during 1000 sec observation

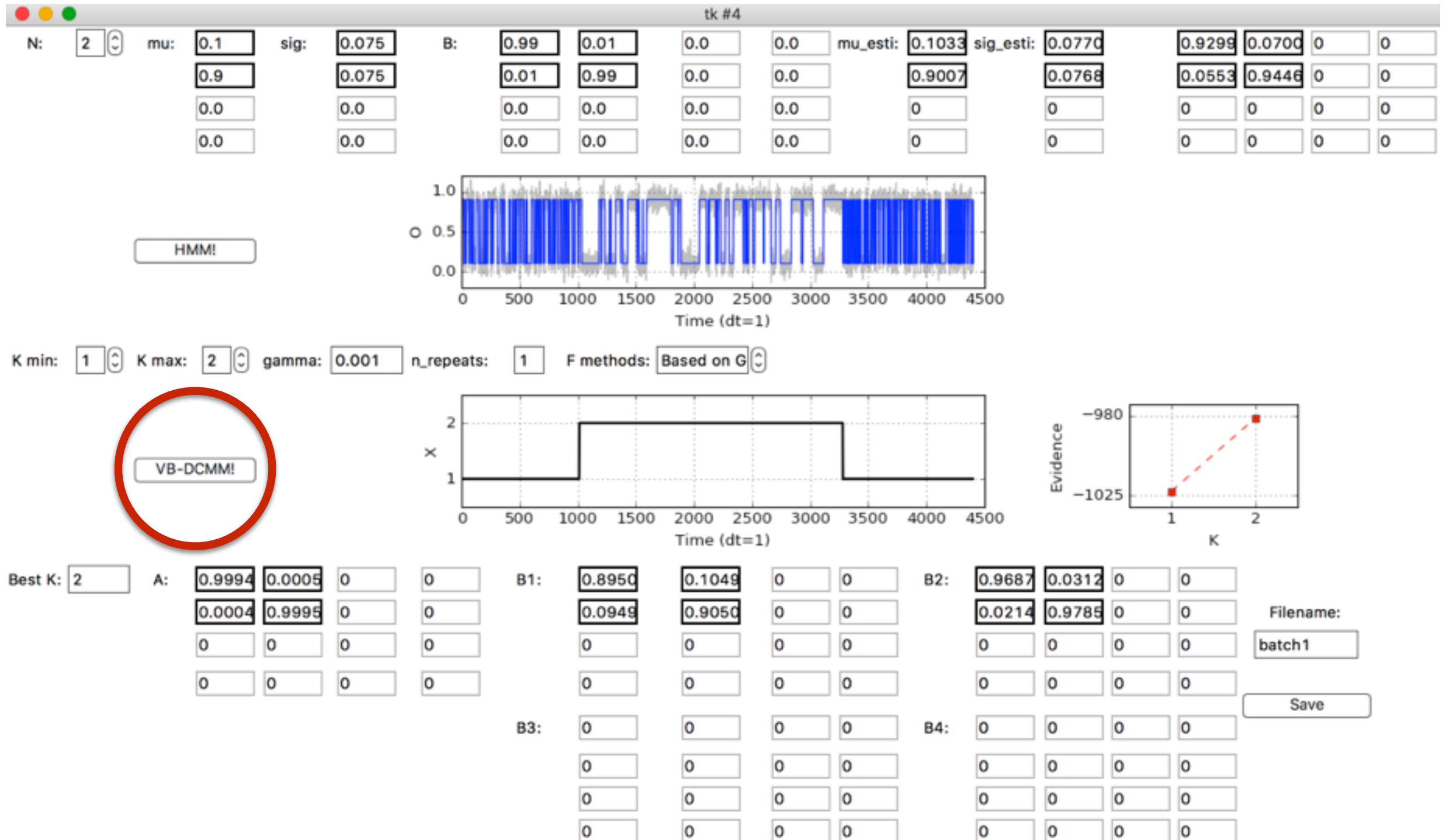
n\_repeats: The number of repeats VB-DCMM in each model. To avoid local minimum n\_repeats=20 recommended.

F methods: The method to treat model degeneracy. See the paper for the detail.



### 13. Find a most probable sequence of internal state

VB-DCMM!: Start VB-DCMM



## 14. Estimated sequence of internal state





## 15. Estimated lower bound of evidence.

K with maximum value will be selected as a best model.



## 16. Estimated parameters

Best K: Best model

A: Estimated internal state transition matrix.

B1—4: Estimated observable transition matrices.



## 17. Save the results

Filename: The name for save-files.

Save: Save data.

tk #4

N: 2

mu: 0.1  
0.9  
0.0  
0.0

sig: 0.075  
0.075  
0.0  
0.0

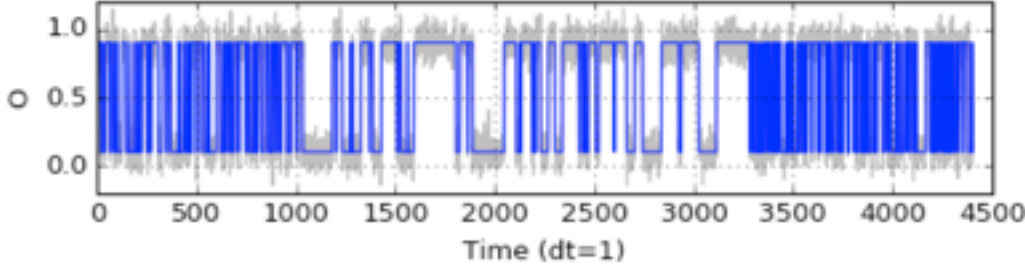
B: 0.99 0.01 0.0 0.0  
0.01 0.99 0.0 0.0  
0.0 0.0 0.0 0.0  
0.0 0.0 0.0 0.0

mu\_esti: 0.1033  
0.9007  
0  
0

sig\_esti: 0.0770  
0.0768  
0  
0

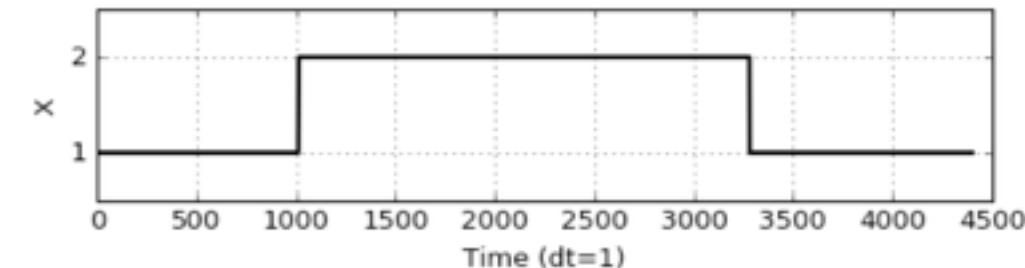
0.9299 0.0700 0 0  
0.0553 0.9446 0 0  
0 0 0 0  
0 0 0 0

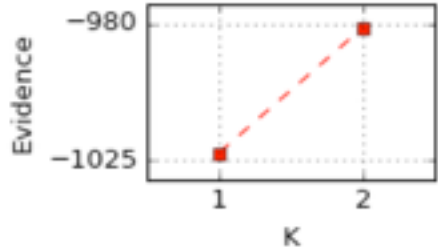
HMM!



K min: 1 K max: 2 gamma: 0.001 n\_repeats: 1 F methods: Based on G

VB-DCMM!





Best K: 2

A: 0.9994 0.0005 0 0  
0.0004 0.9995 0 0  
0 0 0 0  
0 0 0 0

B1: 0.8950 0.1049 0 0  
0.0949 0.9050 0 0  
0 0 0 0  
0 0 0 0

B2: 0.9687 0.0312 0 0  
0.0214 0.9785 0 0  
0 0 0 0  
0 0 0 0

B3: 0 0 0 0  
0 0 0 0  
0 0 0 0  
0 0 0 0

B4: 0 0 0 0  
0 0 0 0  
0 0 0 0  
0 0 0 0

Filename:  
batch1

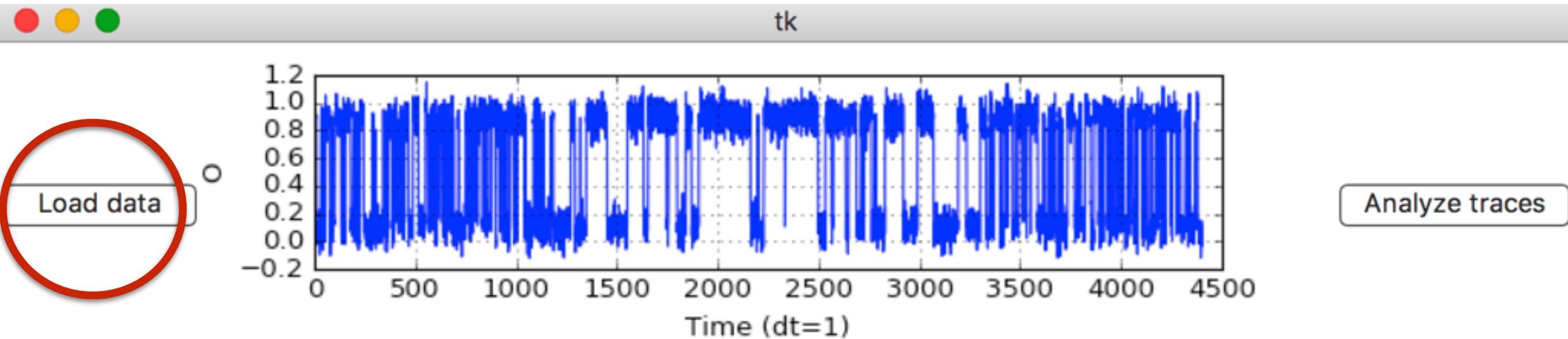
Save



**3. VBDCMM\_gui\_load.py**  
**(to load experimental data)**

# 1. Load experimental data

Load data: Click to load data. Data should be 1-D column vector saved in text file.  
For example, if data contains 1000 data points, it should be **1000 x 1** matrix.



## 2. Analyse traces

Analyse traces: Start VB-DCMM

