Bayesian Inference for ϵ-machines

Lecture 2

Christopher C. Strelioff

Complexity Sciences Center & Department of Physics
University of California at Davis

2013.05.16

Overview

- Today
    - Inferring structure (model topology)
        - Enumeration and model comparison for topological ϵ-machines
    - **Ex 3:** Infer *structure* of EvenOdd process
    - **Ex 4:** Survey of inferring Golden Mean, Even, Simple Nonunifilar Source (SNS)
    - Complications: out-of-class, non-stationary processes
- Previous Lecture
    - Goals of statistical inference
    - Introduction to Bayesian inference
    - **Ex 1:** Biased Coin
    - Unifilar HMMs and ϵ-machines
    - **Ex 2:** EvenOdd Process
        - Infer transition probabilities and start state

Review from last time

Bayes' Theorem: Update **Prior** Distribution to **Posterior** Distribution

- We inferred transition probabilities θ_i given data D and assumed model structure M_i

$$\mathbb{P}(\theta_i|\mathbf{D}, \sigma_{i,0}, M_i) = \frac{\mathbb{P}(\mathbf{D}|\theta_i, \sigma_{i,0}, M_i)\,\mathbb{P}(\theta_i|\sigma_{i,0}, M_i)}{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)}$$

- Also, we inferred the start state (or, hidden state path)

$$\mathbb{P}(\sigma_{i,0}|\mathbf{D}, M_i) = \frac{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)\,\mathbb{P}(\sigma_{i,0}|M_i)}{\mathbb{P}(\mathbf{D}|M_i)}$$

- Both of these inferences were made using **fixed** model topology M_i

Review from last time

Bayes' Theorem: Update **Prior** Distribution to **Posterior** Distribution

- We inferred transition probabilities θ_i given data D and assumed model structure M_i

$$\mathbb{P}(\theta_i|\mathbf{D}, \sigma_{i,0}, M_i) = \frac{\mathbb{P}(\mathbf{D}|\theta_i, \sigma_{i,0}, M_i)\boxed{\mathbb{P}(\theta_i|\sigma_{i,0}, M_i)}}{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)}$$

- Also, we inferred the start state (or, hidden state path)

$$\mathbb{P}(\sigma_{i,0}|\mathbf{D}, M_i) = \frac{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)\boxed{\mathbb{P}(\sigma_{i,0}|M_i)}}{\mathbb{P}(\mathbf{D}|M_i)}$$

- Both of these inferences were made using **fixed** model topology M_i

Review from last time

Bayes' Theorem: Update **Prior** Distribution to **Posterior** Distribution

- We inferred transition probabilities θ_i given data D and assumed model structure M_i

$$\boxed{\mathbb{P}(\theta_i|\mathbf{D}, \sigma_{i,0}, M_i)} = \frac{\mathbb{P}(\mathbf{D}|\theta_i, \sigma_{i,0}, M_i)\,\mathbb{P}(\theta_i|\sigma_{i,0}, M_i)}{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)}$$

- Also, we inferred the start state (or, hidden state path)

$$\boxed{\mathbb{P}(\sigma_{i,0}|\mathbf{D}, M_i)} = \frac{\mathbb{P}(\mathbf{D}|\sigma_{i,0}, M_i)\,\mathbb{P}(\sigma_{i,0}|M_i)}{\mathbb{P}(\mathbf{D}|M_i)}$$

- Both of these inferences were made using **fixed** model topology M_i

Inferring Structure

A Step Back. . .

- How is this related to other inference algorithms for HMMs?
- Properties of other approaches (very generally)
    - Usually infer parameters for fixed (assumed) HMM topology
    - Typically consider **nonunifiliar** topologies
    - Algorithms: expectation-maximization, Baum-Welch, etc.
- What we do differently
    - Restrict to unifilar HMM topologies
    - Use model comparison to infer model topology
    - Provide a distribution over candidate models
    - A different view of structural inference?
- What set of models $\mathcal{M}$ should we use?

Topological ϵ-machines

A set of candidate structures

- Algorithm to efficiently enumerate all topological ϵ-machines with specified alphabet and number of states
  - B. Johnson et al., *Enumerating Finitary Processes* `http://arxiv.org/abs/1011.0036`
- Use this algorithm to create our set of candidate model structures $\mathcal{M}$
  - A brute-force method to infer structure
    - Try all structures within time/computational limits

Finite-state, edge-labeled HMMs

Definition

A finite-state, edge-labeled, hidden Markov model (HMM) consists of:

1. *A finite set of hidden states* $\boldsymbol{\mathcal{S}} = \{\sigma_1, \ldots, \sigma_n\}$
2. *A finite output alphabet* $\mathcal{X}$
3. *A set of* $N \times N$ *symbol-labeled transition matrices* $T^{(x)}$, $x \in \mathcal{X}$, *where* $T^{(x)}_{i,j}$ *is the probability of transitioning from state* σ_i *to state* σ_j *on symbol* x. *The corresponding overall state-to-state transition matrix is denoted* $T = \sum_{x \in \mathcal{X}} T^{(x)}$.

Finite-state ϵ-machine

Definition

A finite-state ϵ-machine is a finite-state, edge-labeled, hidden Markov model with the following properties:

1. *Unifilarity: For each state $\sigma_i \in \boldsymbol{\mathcal{S}}$ and each symbol $x \in \mathcal{X}$ there is at most one outgoing edge from state σ_i that outputs symbol x.*
2. *Probabilistically distinct states: For each pair of distinct states $\sigma_k, \sigma_j \in \boldsymbol{\mathcal{S}}$ there exists some finite word $w = x_0 x_1 \ldots x_{L-1}$ such that:*

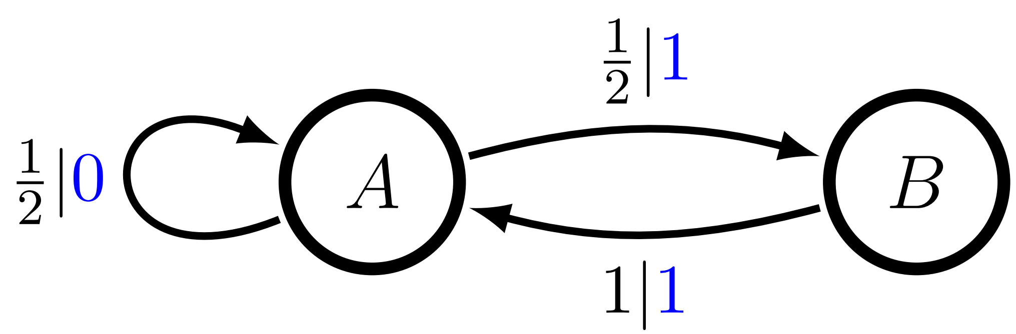
$$\mathbb{P}(w|\sigma_0 = \sigma_k) \neq \mathbb{P}(w|\sigma_0 = \sigma_j)$$

Topological ϵ-machines

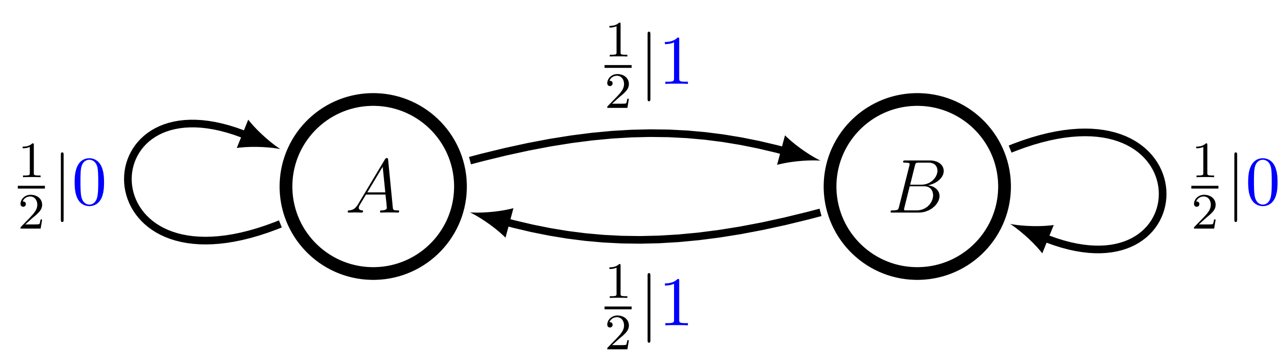
Definition

A topological ϵ-machine is a finite-state ϵ-machine where the transition probabilities for each state are equal for all outgoing edges.

Okay



Nope



Topological ϵ-machines

How many are there?

States, n	Edges, k	$F_{n,2}$
1		1
	2	1
2		7
	2	1
	3	6
3		78
	3	2
	4	22
	5	54
4		1,388
	4	3
	5	68
	6	403
	7	914

Number of

- full alphabet
- binary
- topological ϵ-machines

History vs Generator ϵ-machines

How to think about this approach to structural inference

- You have studied the **history** formulation for ϵ-machines using the equivalence relation
  - A process determined the ϵ-machine structure through the equivalence relation
- An alternative is the **generator** formulation developed by Travers and Crutchfield*
  - An ϵ-machine defines the process that can be produced by the given structure
  - Formulations recently proved to be equivalent

*Travers & Crutchfield, *Exact synchronization for finite-state sources (2011)*; *Asymptotic synchronization for finite-state sources (2011)*; *Equivalence of history and generator ϵ-machines (2011)*.

Inferring Structure

Model Comparison for Topology: Update **Prior** to **Posterior**

- Choose a set of candidate models $\boldsymbol{\mathcal{M}}$
- Bayes' Theorem at the level of structure, or model topology

$$\mathbb{P}(M_j|\mathbf{D},\boldsymbol{\mathcal{M}}) = \frac{\mathbb{P}(\mathbf{D}|M_j,\boldsymbol{\mathcal{M}})\,\mathbb{P}(M_j|\boldsymbol{\mathcal{M}})}{\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}})}$$

where

$$\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}}) = \sum_{M_i \in \boldsymbol{\mathcal{M}}} \mathbb{P}(\mathbf{D}|M_i,\boldsymbol{\mathcal{M}})\mathbb{P}(M_i|\boldsymbol{\mathcal{M}})$$

- As before, we have to specify a prior at this level
- We also use $\mathbb{P}(\mathbf{D}|M_j,\boldsymbol{\mathcal{M}}) = \mathbb{P}(\mathbf{D}|M_j)$
  - $\mathbb{P}(\mathbf{D}|M_j)$ came from inferring start state

Inferring Structure

Model Comparison for Topology: Update **Prior** to **Posterior**

- Choose a set of candidate models $\boldsymbol{\mathcal{M}}$
- Bayes' Theorem at the level of structure, or model topology

$$\mathbb{P}(M_j|\mathbf{D}, \boldsymbol{\mathcal{M}}) = \frac{\mathbb{P}(\mathbf{D}|M_j, \boldsymbol{\mathcal{M}})\boxed{\mathbb{P}(M_j|\boldsymbol{\mathcal{M}})}}{\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}})}$$

where

$$\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}}) = \sum_{M_i \in \boldsymbol{\mathcal{M}}} \mathbb{P}(\mathbf{D}|M_i, \boldsymbol{\mathcal{M}})\mathbb{P}(M_i|\boldsymbol{\mathcal{M}})$$

- As before, we have to specify a prior at this level
- We also use $\mathbb{P}(\mathbf{D}|M_j, \boldsymbol{\mathcal{M}}) = \mathbb{P}(\mathbf{D}|M_j)$
  - $\mathbb{P}(\mathbf{D}|M_j)$ came from inferring start state

Inferring Structure

Model Comparison for Topology: Update **Prior** to **Posterior**

- Choose a set of candidate models $\boldsymbol{\mathcal{M}}$
- Bayes' Theorem at the level of structure, or model topology

$$\boxed{\mathbb{P}(M_j|\mathbf{D},\boldsymbol{\mathcal{M}})} = \frac{\mathbb{P}(\mathbf{D}|M_j,\boldsymbol{\mathcal{M}})\,\mathbb{P}(M_j|\boldsymbol{\mathcal{M}})}{\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}})}$$

where

$$\mathbb{P}(\mathbf{D}|\boldsymbol{\mathcal{M}}) = \sum_{M_i \in \boldsymbol{\mathcal{M}}} \mathbb{P}(\mathbf{D}|M_i,\boldsymbol{\mathcal{M}})\mathbb{P}(M_i|\boldsymbol{\mathcal{M}})$$

- As before, we have to specify a prior at this level
- We also use $\mathbb{P}(\mathbf{D}|M_j,\boldsymbol{\mathcal{M}}) = \mathbb{P}(\mathbf{D}|M_j)$
  - $\mathbb{P}(\mathbf{D}|M_j)$ came from inferring start state

Prior for Model Topologies

Inferring Structure

- We choose a simple, single parameter, prior over model topologies

$$\mathbb{P}(M_i|\boldsymbol{\mathcal{M}}) = \frac{\exp\left(-\beta\, f(M_i)\right)}{\sum_{M_j \in \boldsymbol{\mathcal{M}}} \exp\left(-\beta\, f(M_j)\right)}$$

- The function is chosen to penalize for larger structure
  - Number of states in HMM– this is the CMPy default
  - Number of edges in HMM
  - Other ideas?

What do we do with the posterior at this level?

Sampling vs MAP

- Application of Bayes' theorem provides the posterior distribution over models in $\boldsymbol{\mathcal{M}}$

$$\mathbb{P}(M_j | D, \boldsymbol{\mathcal{M}}) \quad , \quad \sum_{M_j \in \boldsymbol{\mathcal{M}}} \mathbb{P}(M_j | D, \boldsymbol{\mathcal{M}}) = 1$$

- **Use 1:** Sample from posterior over models
  - Quantify uncertainty in structure
  - Can also quantify uncertainty in start state and transition probabilities as seen previously
  - Estimate mean and credible interval for any function of interest: h_μ, C_μ, etc.

- **Use 2:** Choose a single *maximum a posteriori* (MAP) structure

$$M_{\text{map}} = \underset{M_j \in \boldsymbol{\mathcal{M}}}{\operatorname{argmax}} \, \mathbb{P}(M_j | \mathbf{D}, \boldsymbol{\mathcal{M}})$$

Sampling Algorithms

Whole posterior vs MAP

ALGORITHM 1: Sample using all topologies in $\boldsymbol{\mathcal{M}}$

---

for n in $(1, N_s)$ **do**:

$M_i \sim \mathbb{P}(M_i|\mathbf{D}, \boldsymbol{\mathcal{M}})$ # sample topology
$\sigma_{i,0} \sim \mathbb{P}(\sigma_{i,0}|\mathbf{D}, M_i)$ # sample start state
$\theta_i \sim \mathbb{P}(\theta_i|\mathbf{D}, \sigma_{i,0}, M_i)$ # sample parameters
$f_n = f(\{p(x|\sigma_i)\})$ # store sample

ALGORITHM 2: Sample using MAP topology

$M_{\text{map}} = \text{argmax}_{M_i \in \boldsymbol{\mathcal{M}}} P(M_i|\mathbf{D}, \boldsymbol{\mathcal{M}})$ # find MAP topology

for n in $(1, N_s)$ **do**:

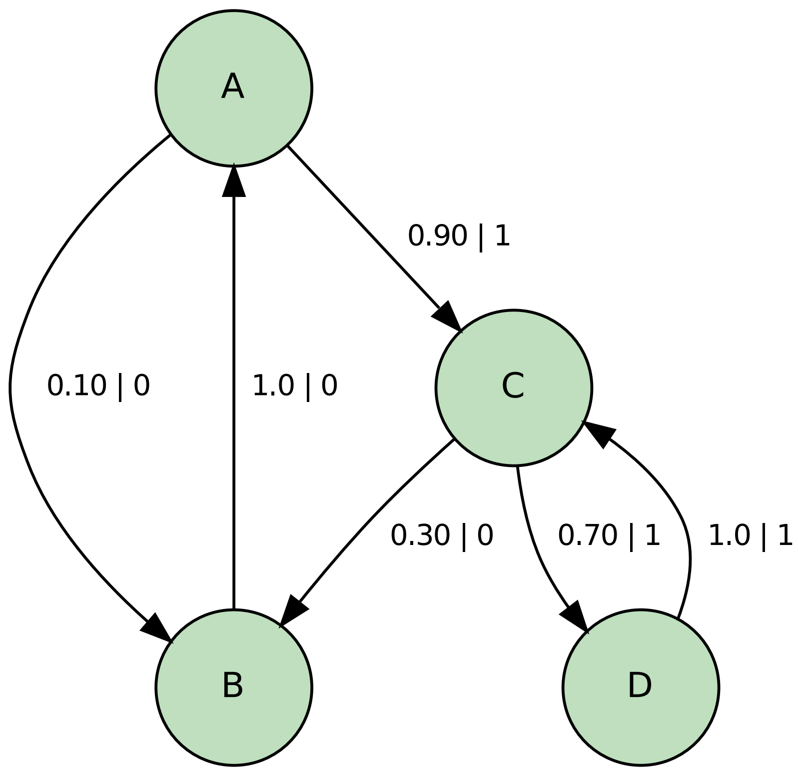
$\sigma_{i,0} \sim \mathbb{P}(\sigma_{i,0}|\mathbf{D}, M_{\text{map}})$ # sample start state
$\theta_i \sim \mathbb{P}(\theta_i|\mathbf{D}, \sigma_{i,0}, M_{\text{map}})$ # sample parameters
$f_n = f(\{p(x|\sigma_i)\})$ # store sample

Ex 3: Inferring the Structure of EvenOdd Process

EvenOdd Process

Use CMPy to generate data

```
import cmpy
eo_str = """A B 0 0.1;A C 1 0.9;B A 0 1.0;
            C B 0 0.3;C D 1 0.7;D C 1 1.0"""
eomachine = cmpy.machines.from_string(eo_str,name='biased EvenOdd',
                                      style=1)
# draw machine
eomachine.draw(filename='figures/evenodd.pdf', show=False)
```



Prior over all 1- to 4-state topological ϵ-machines

Instantiate in CMPy

```
# import inference code
import cmpy.inference.bayesianem as bayesem

# get set of 1- to 4-state topological epsilon machines
modelset1 = bayesem.LibraryGenerator(2,[1,2,3,4])
# declare prior over models
prior = bayesem.ModelComparisonEM(modelset1, beta=4., verbose=True)
```

```
*Infer Machine Topology (InferEM)
 * Model Prior- beta: 4.00000
 * Inferring all machines...
  ** 1474 machines considered, 1474 possible
 * Calculating log evidence for all machines...
 * Calculating model probabilities for all machines...
```

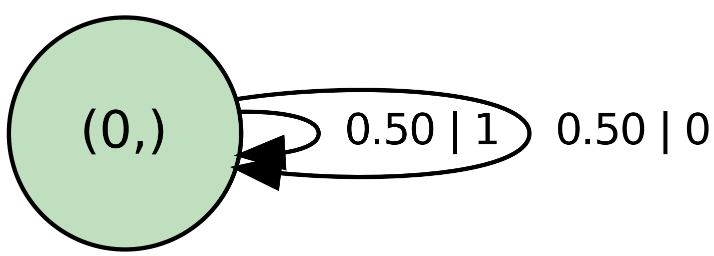
Prior

Most probable topology (a priori)

- Get MAP topology and use prior mean for transition probabilities
- Average over uncertainty in start state

```
# get machine from prior
pr, eo_prior_em = prior.get_MAP_PM_machine()[0]

# draw machine
eo_prior_em.draw(filename='figures/eo_prior_mach.pdf',
                 show=False)
```



Posterior given data from EvenOdd Process

Instantiate in CMPy

```
# generate data
eo_data = eomachine.symbols(5000)

# get set of 1- to 4-state topological epsilon machines
modelset2 = bayesem.LibraryGenerator(2,[1,2,3,4])
# declare posterior over models
posterior = bayesem.ModelComparisonEM(modelset2, eo_data, beta=4.,
                                      verbose=True)
```

```
*Infer Machine Topology (InferEM)
 * Model Prior- beta: 4.00000
 * Inferring all machines...
  ** 1474 machines considered, 175 possible
 * Calculating log evidence for all machines...
 * Calculating model probabilities for all machines...
```

Posterior

MAP topology

- Get MAP topology and use posterior mean for transition probabilities
- Average over uncertainty in start state (if any)

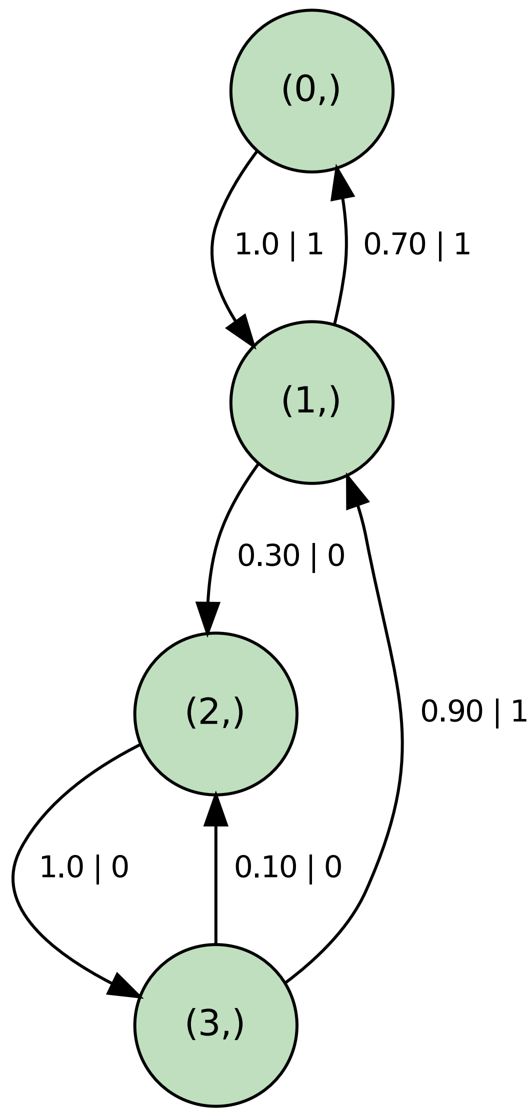
```
# get machine from posterior
pr, eo_post_em = posterior.get_MAP_PM_machine()[0]

# draw machine
eo_post_em.draw(filename='figures/eo_posterior_mach.pdf',
                show=False)
```

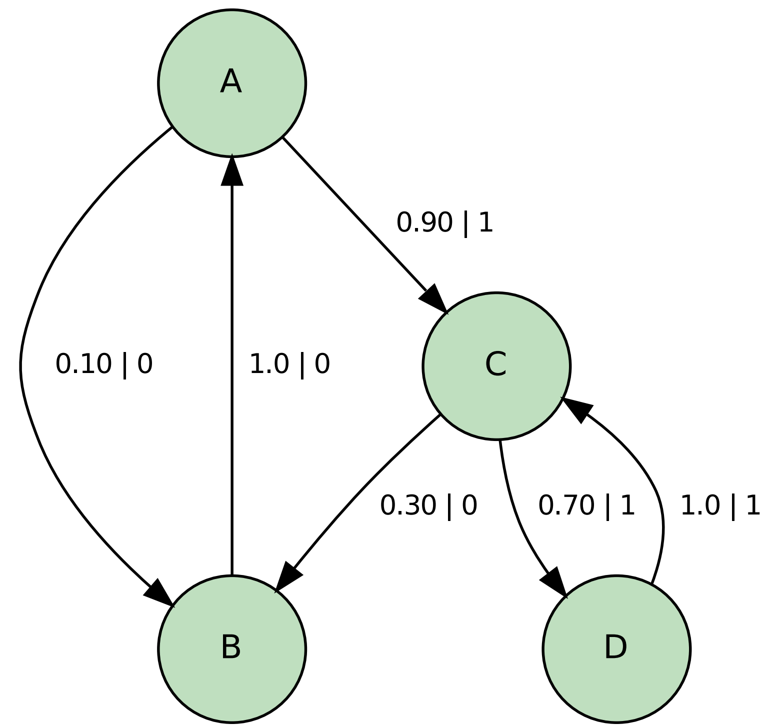
Posterior

MAP topology

Inferred Topology



True Topology



Prior, Posterior and C_μ, h_μ

Sample from prior & posterior

```
num_samples = 2000
eo_prior_hmu = []; eo_prior_Cmu = [];
eo_posterior_hmu = []; eo_posterior_Cmu = [];

# generate and store samples
for n in range(num_samples):
    # prior
    (node,machine) = prior.generate_sample()
    hmu = machine.entropy_rate()
    Cmu = machine.statistical_complexity()
    eo_prior_hmu.append(hmu); eo_prior_Cmu.append(Cmu)

    # posterior
    (node,machine) = posterior.generate_sample()
    hmu = machine.entropy_rate()
    Cmu = machine.statistical_complexity()
    eo_posterior_hmu.append(hmu); eo_posterior_Cmu.append(Cmu)
```

Prior vs Posterior, h_μ

Plot h_μ samples

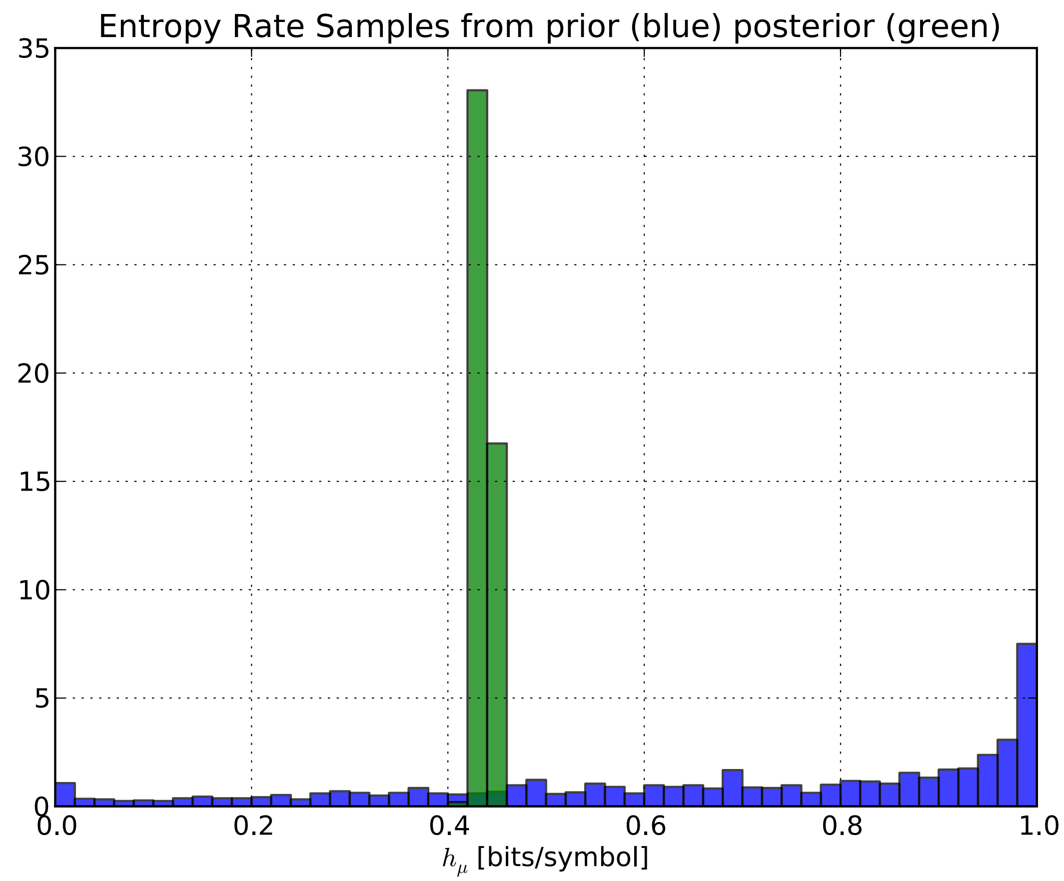
```
import pylab as plt
# prior hmu -- blue
n, bins, patches = plt.hist(eo_prior_hmu, 50, range=[0.0,1.0],
                            normed=1, facecolor='blue', alpha=0.75,
                            cumulative=False)

# posterior hmu -- green
n, bins, patches = plt.hist(eo_posterior_hmu, 50, range=[0.0,1.0],
                            normed=1, facecolor='green', alpha=0.75,
                            cumulative=False)

plt.xlabel(r'$h_{\mu}$ [bits/symbol]')
plt.title('Entropy Rate Samples from prior (blue) posterior (green)')
plt.grid(True)
plt.savefig('figures/eo_hmu_hist.pdf')
```

Prior vs Posterior, h_μ

Plot h_μ samples




```
true hmu:  0.438432153702  [bits/symbol]
```

Prior vs Posterior, C_μ

Plot C_μ samples

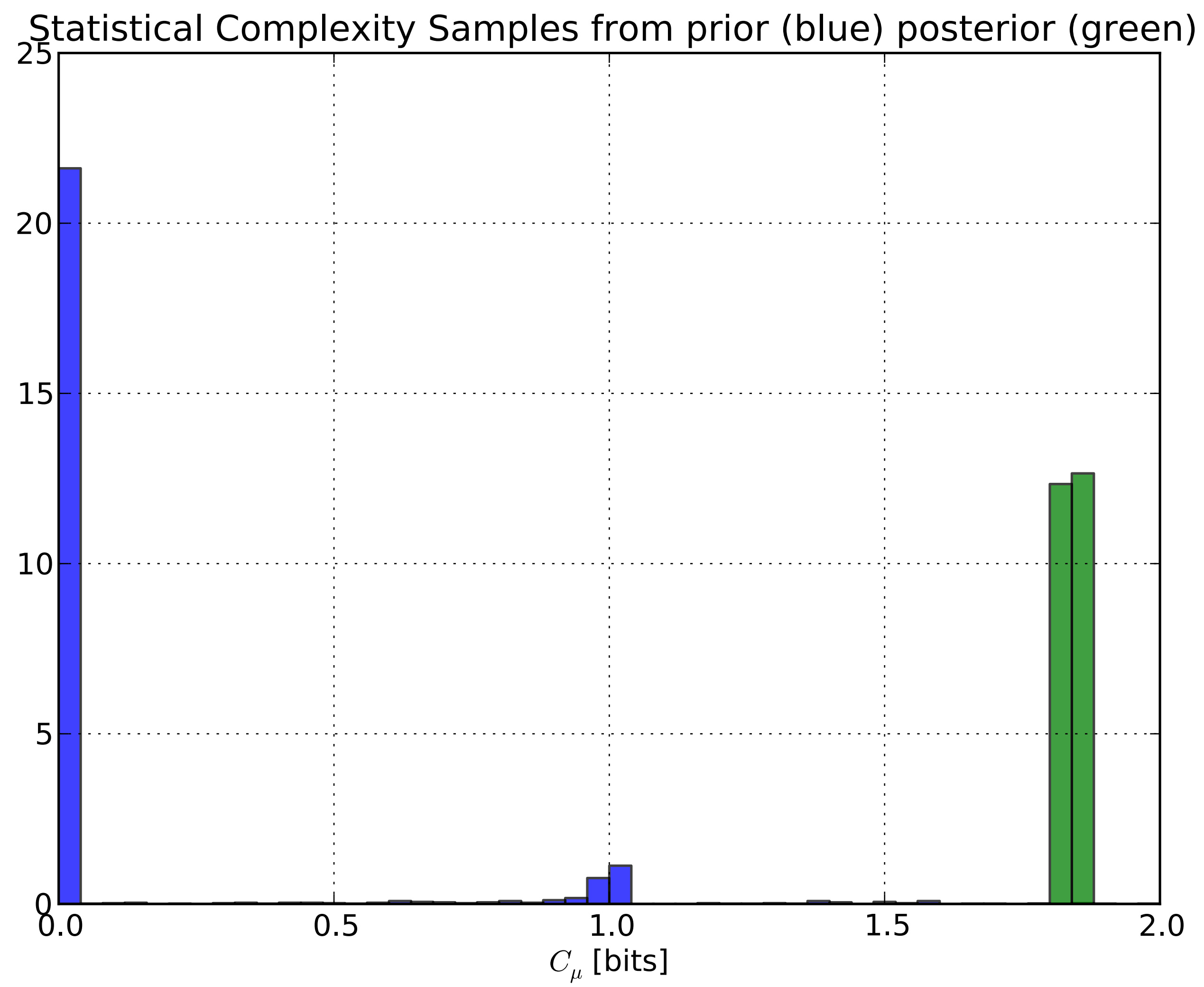
```
plt.clf()
# prior Cmu -- blue
n, bins, patches = plt.hist(eo_prior_Cmu, 50, range=[0.0,2.0],
                            normed=1, facecolor='blue', alpha=0.75,
                            cumulative=False)

# posterior Cmu -- green
n, bins, patches = plt.hist(eo_posterior_Cmu, 50, range=[0.0,2.0],
                            normed=1, facecolor='green', alpha=0.75,
                            cumulative=False)

plt.xlabel(r'$C_{\mu}$ [bits]')
plt.title('Statistical Complexity Samples from prior (blue) posterior
n)')
plt.grid(True)
plt.savefig('figures/eo_Cmu_hist.pdf')
```

Prior vs Posterior, C_μ

Plot C_μ samples




```
true Cmu:  1.84152253296  [bits]
```

Prior vs Posterior, C_μ, h_μ

Plot C_μ and h_μ samples

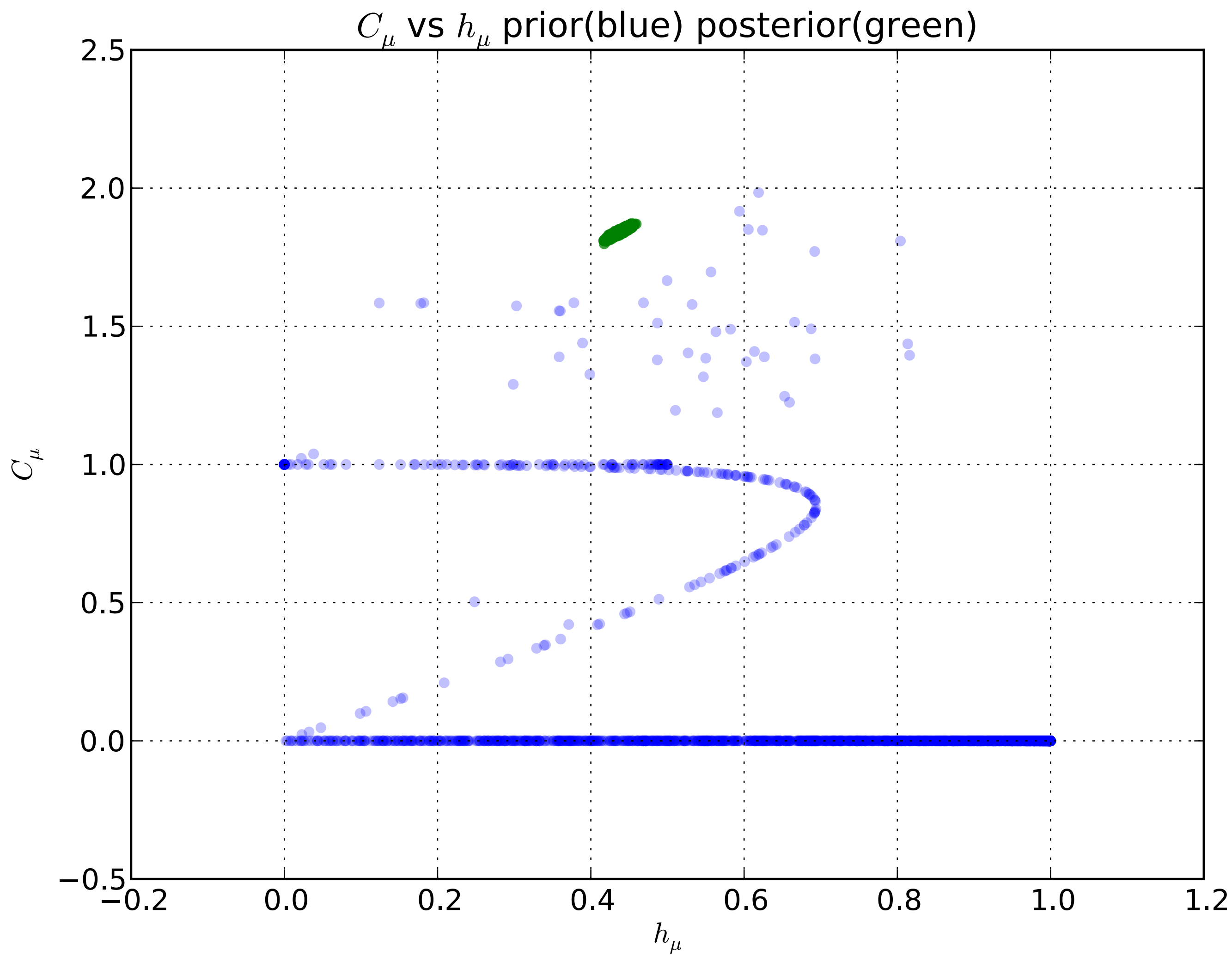
```
# prior - blue
plt.clf()
plt.scatter(eo_prior_hmu, eo_prior_Cmu,s=20,
facecolor='blue', edgecolor='none',alpha=0.25)

# posterior - green
plt.scatter(eo_posterior_hmu, eo_posterior_Cmu,s=20,
            facecolor='green', edgecolor='none',alpha=0.75)

plt.ylabel(r'$C_{\mu}$')
plt.xlabel(r'$h_{\mu}$')
plt.title(r'$C_{\mu}$ vs $h_{\mu}$ prior(blue) posterior(green)')
plt.grid(True)
plt.savefig('figures/eo_Cmuhmu.pdf')
```

Prior vs Posterior, C_μ, h_μ

Plot C_μ and h_μ samples




```
true Cmu:  1.84152253296  [bits]
true hmu:  0.438432153702  [bits/symbol]
```

Ex 4: Golden Mean, Even, SNS

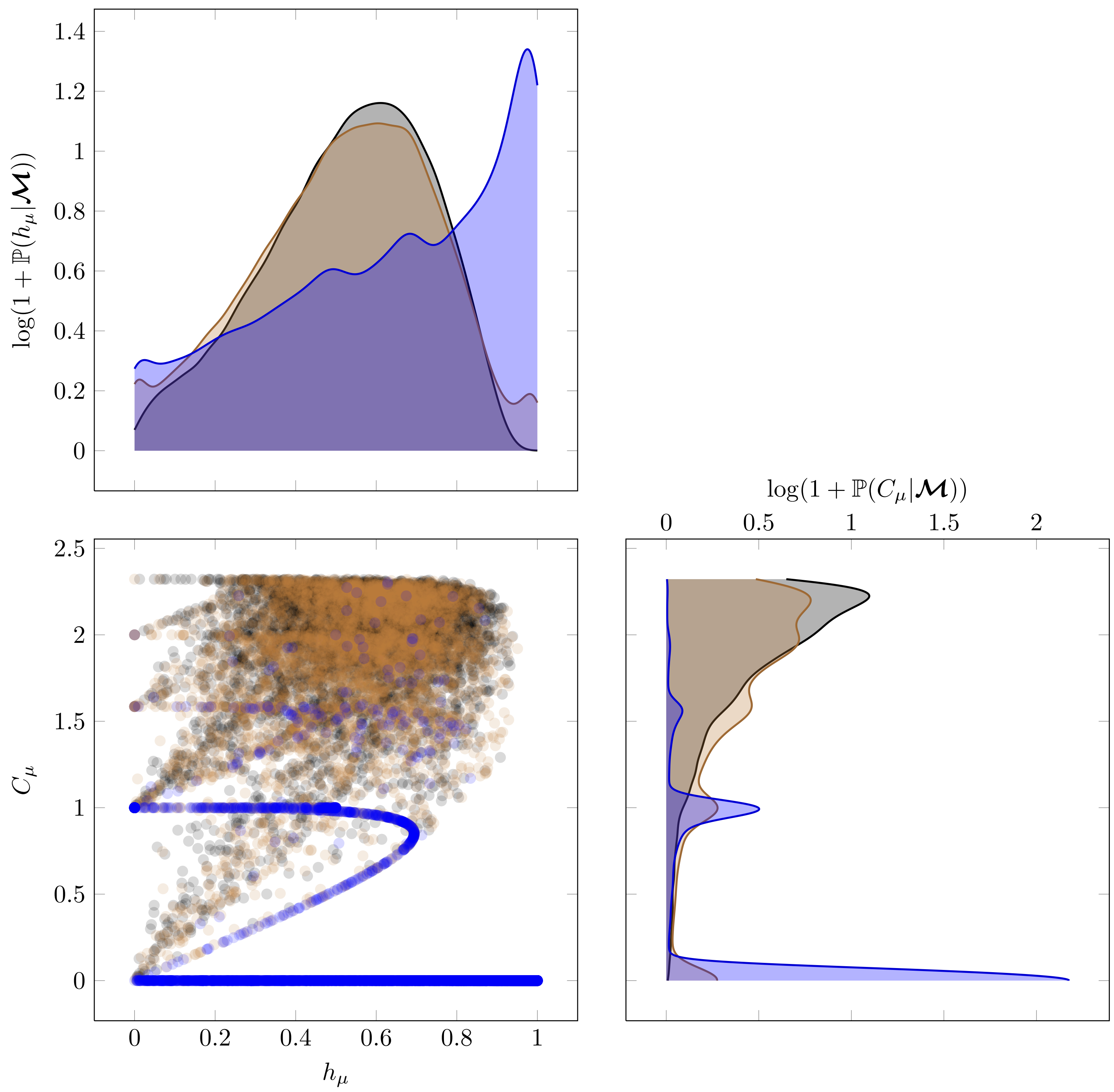
Survey

Golden Mean, Even, SNS

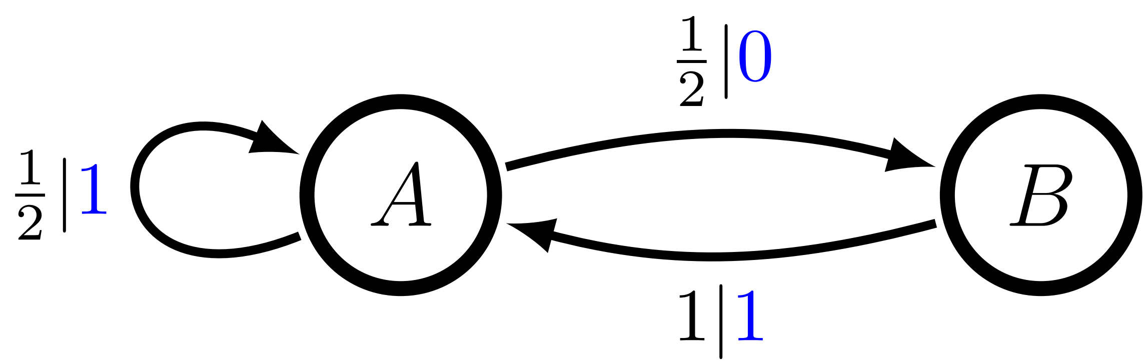
- A single time series of length $T = 2^{17}$ is generated
- Sub-strings of the time series are analyzed at lengths $L = 2^i$ for $i = 0, 1, 2, \ldots, 17$
  - Notate these substrings as $D_{:L}$
- Use 36 660 models that make up all 1- to 5-state binary, topological ϵ-machines
  - Use $\beta = 4$ in all examples
  - Use $50\,000$ samples for each L
- Consider how inference converges as L increases
- Look at h_μ and C_μ as proxies for models inferred

Prior over models

Plot C_μ and h_μ samples: $\beta = 0$ (black), $\beta = 2$ (brown), $\beta = 4$ (blue)

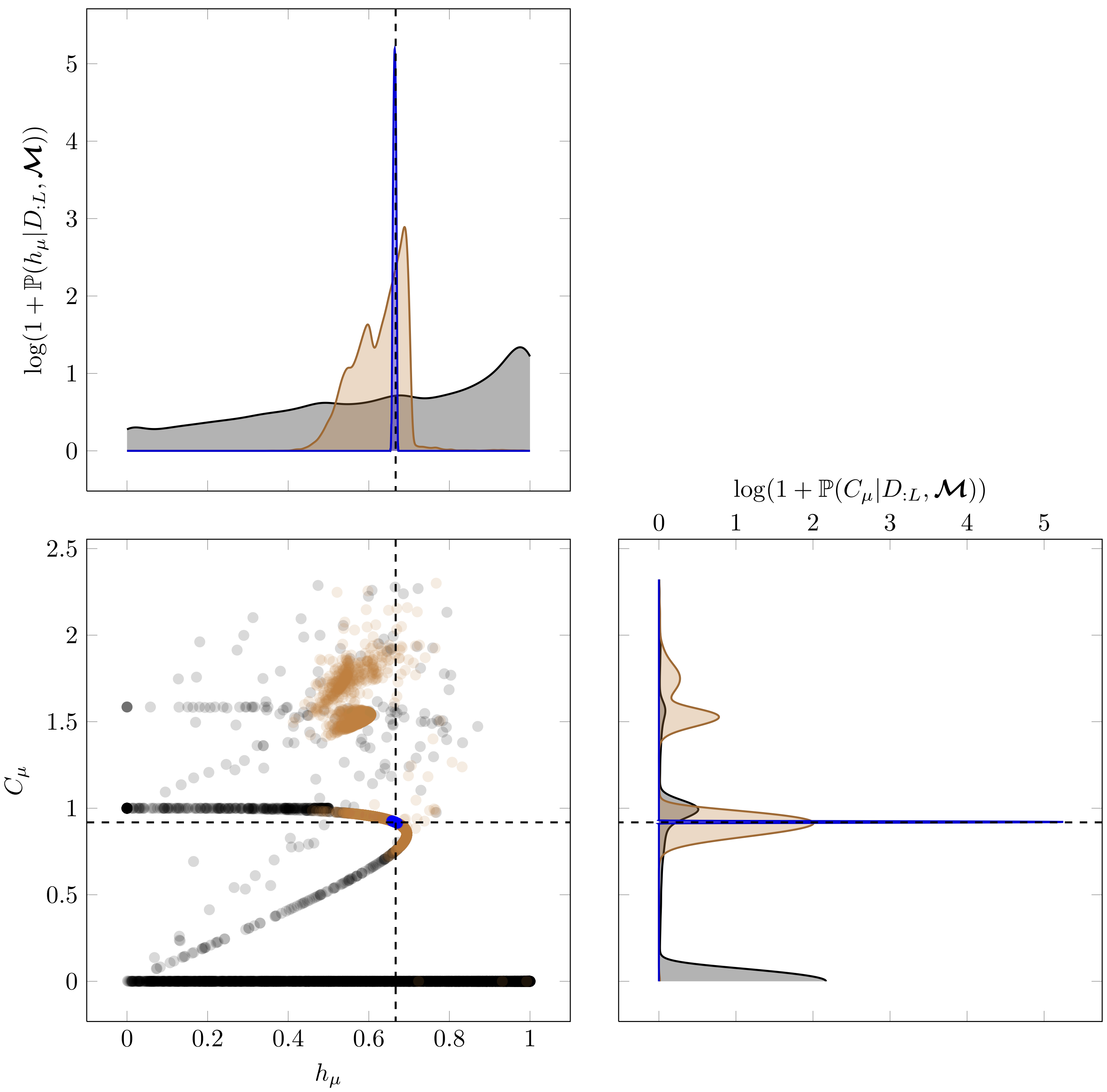


Golden Mean Process



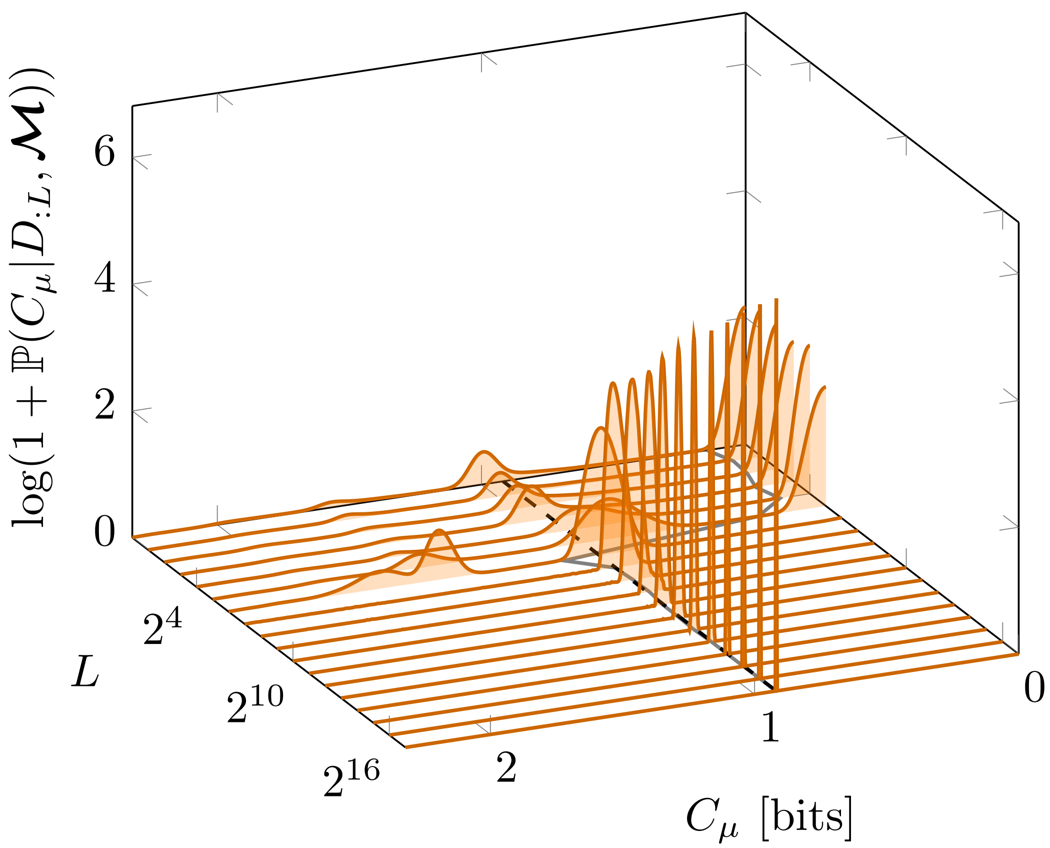
Posterior over models, GM data

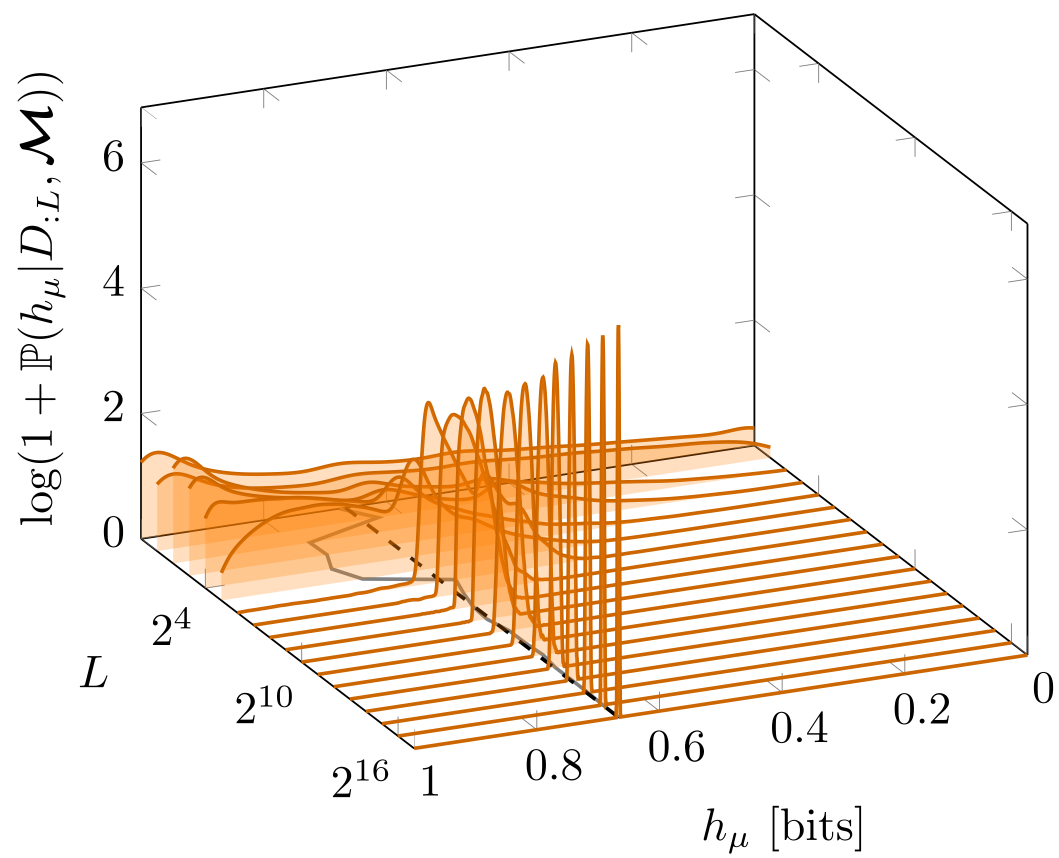
C_μ, h_μ samples: $L = 1$ (black), $L = 64$ (brown), $L = 16\,384$ (blue)



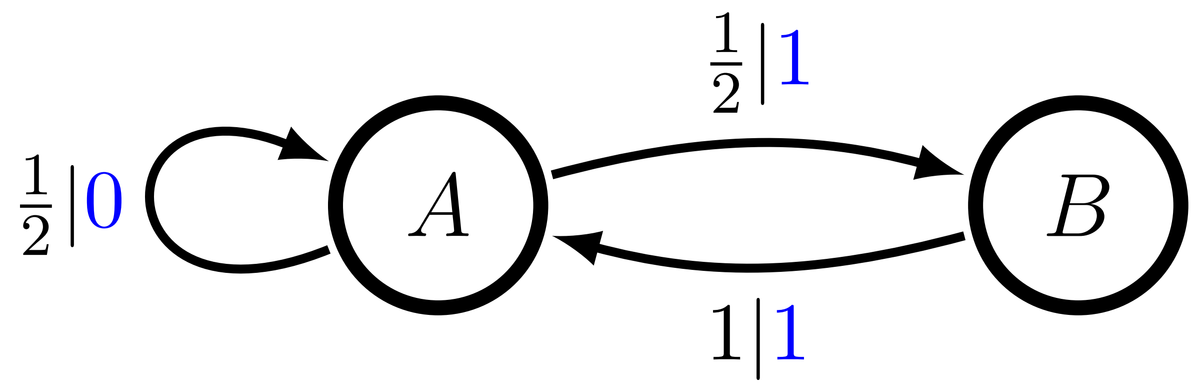
Posterior over models, GM data

C_μ, h_μ convergence



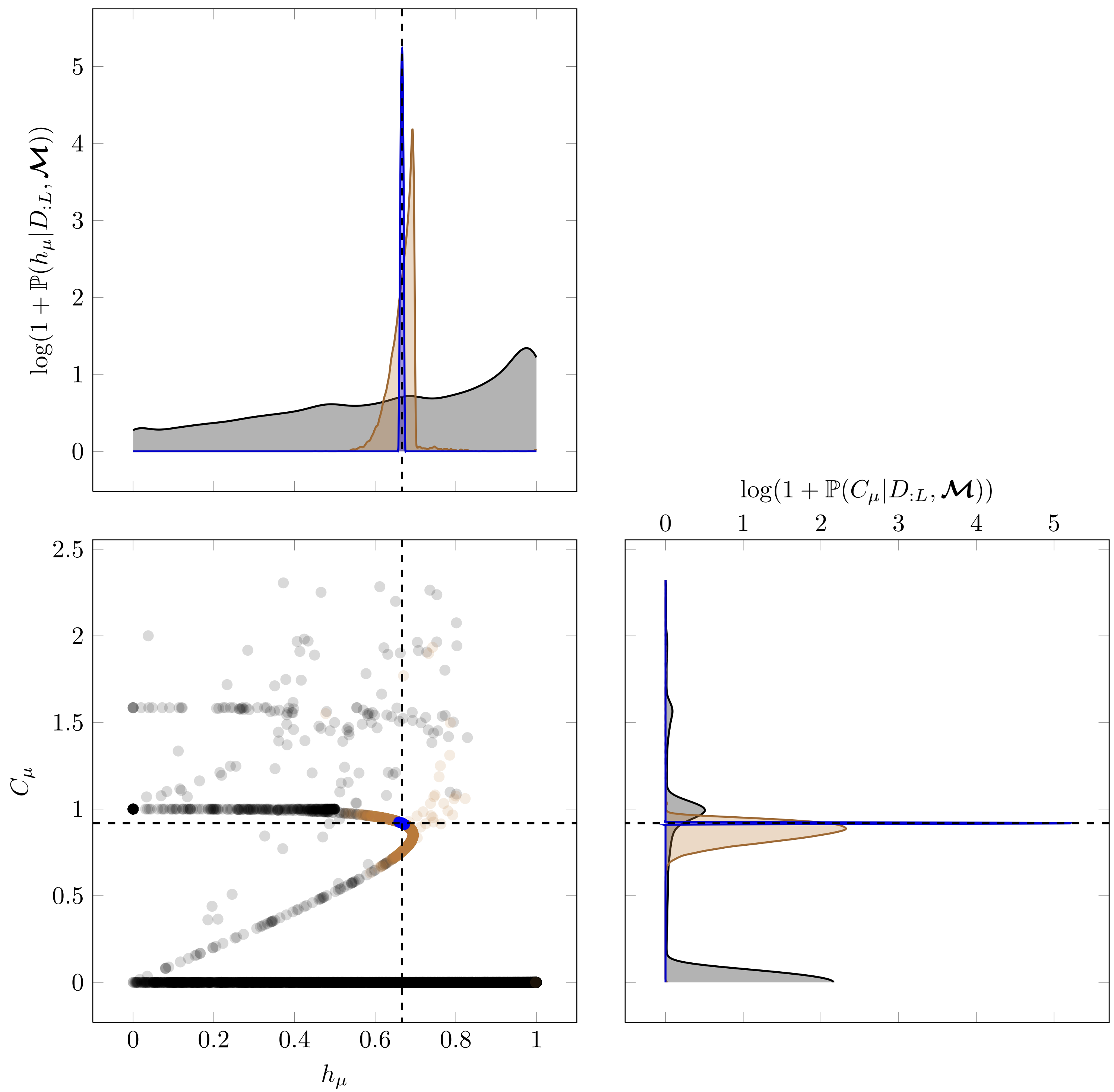


Even Process



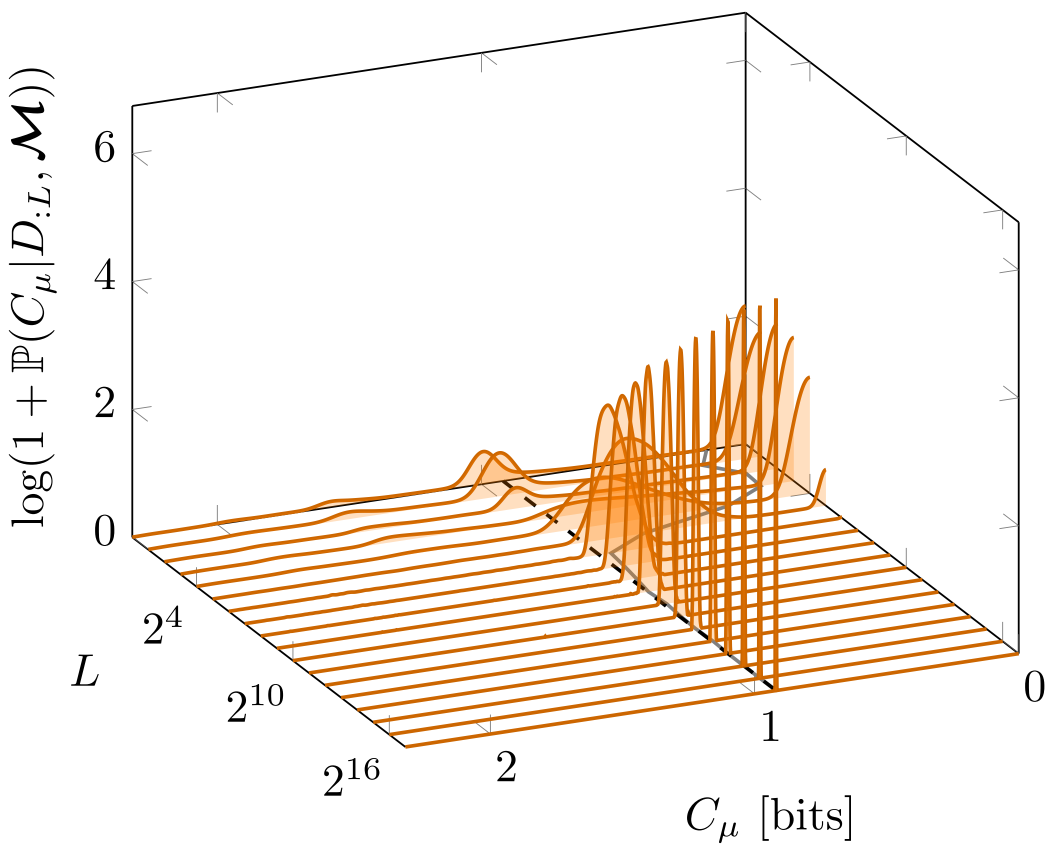
Posterior over models, Even data

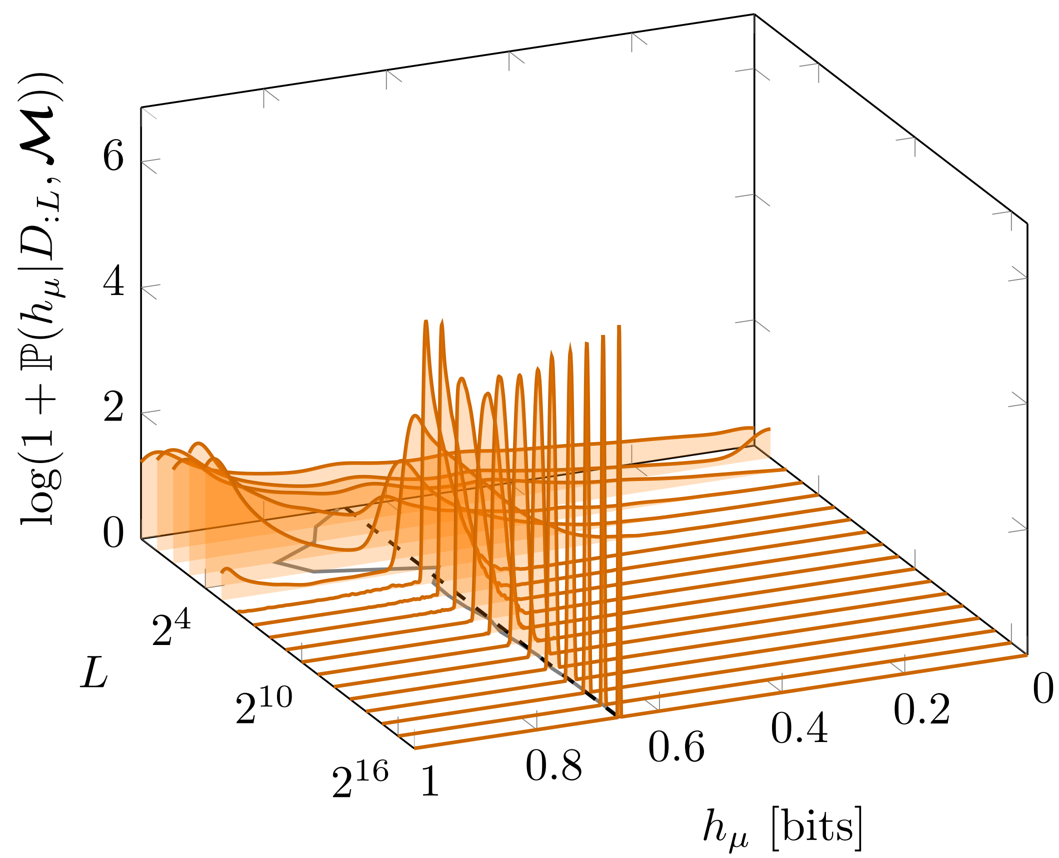
C_μ, h_μ samples: $L = 1$ (black), $L = 64$ (brown), $L = 16\,384$ (blue)



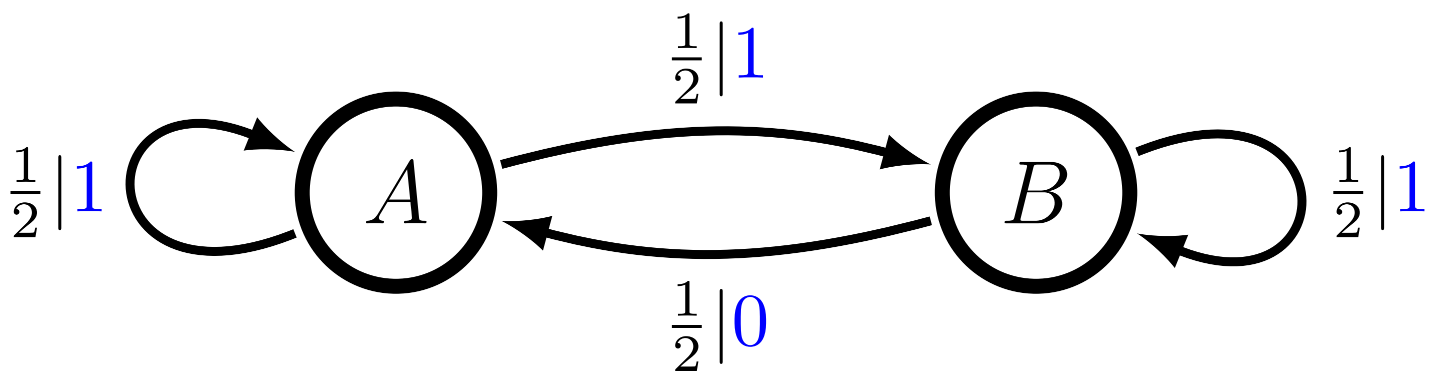
Posterior over models, Even data

C_μ, h_μ convergence



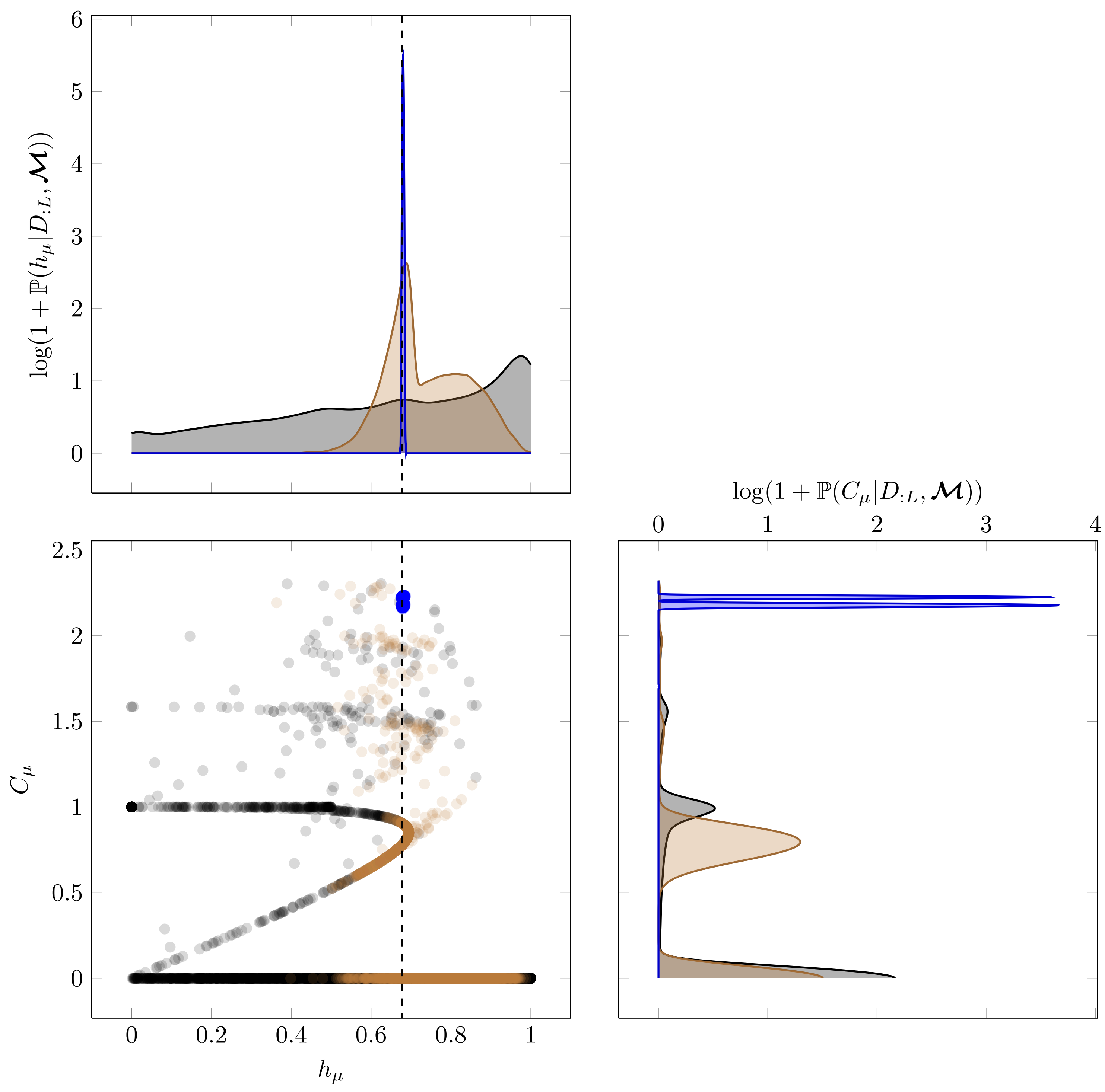


Simple Nonunifilar Source



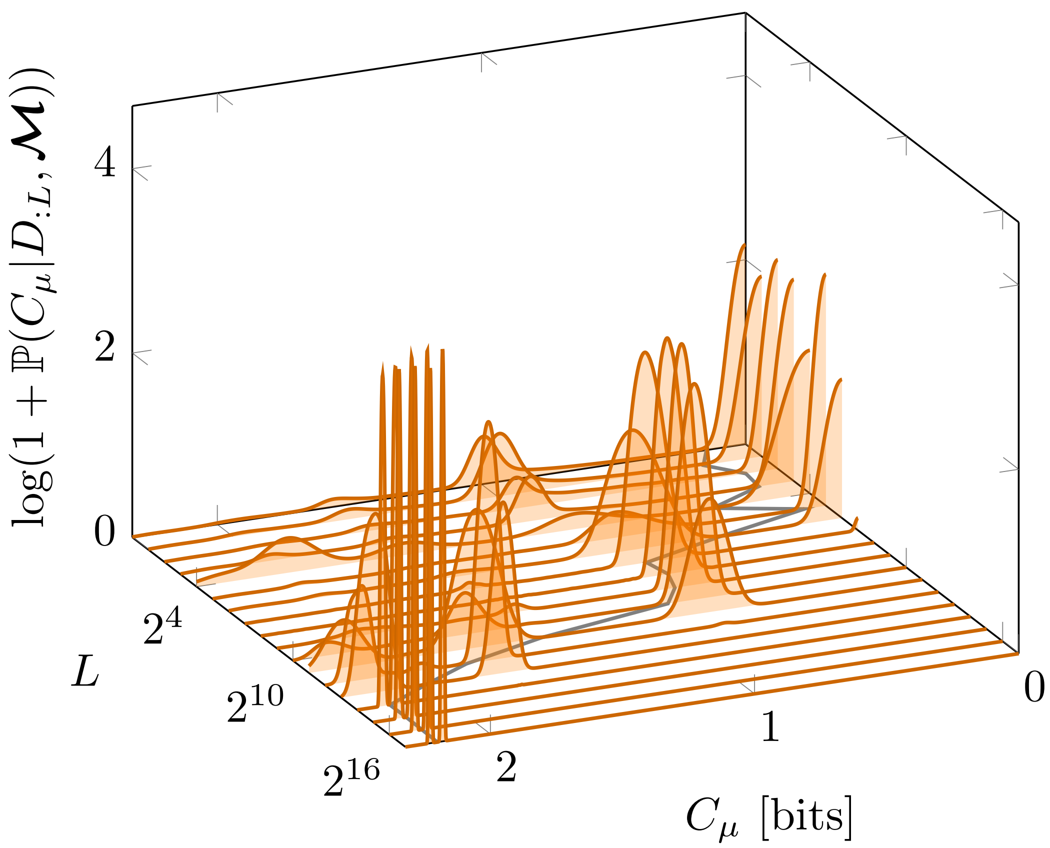
Posterior over models, SNS data

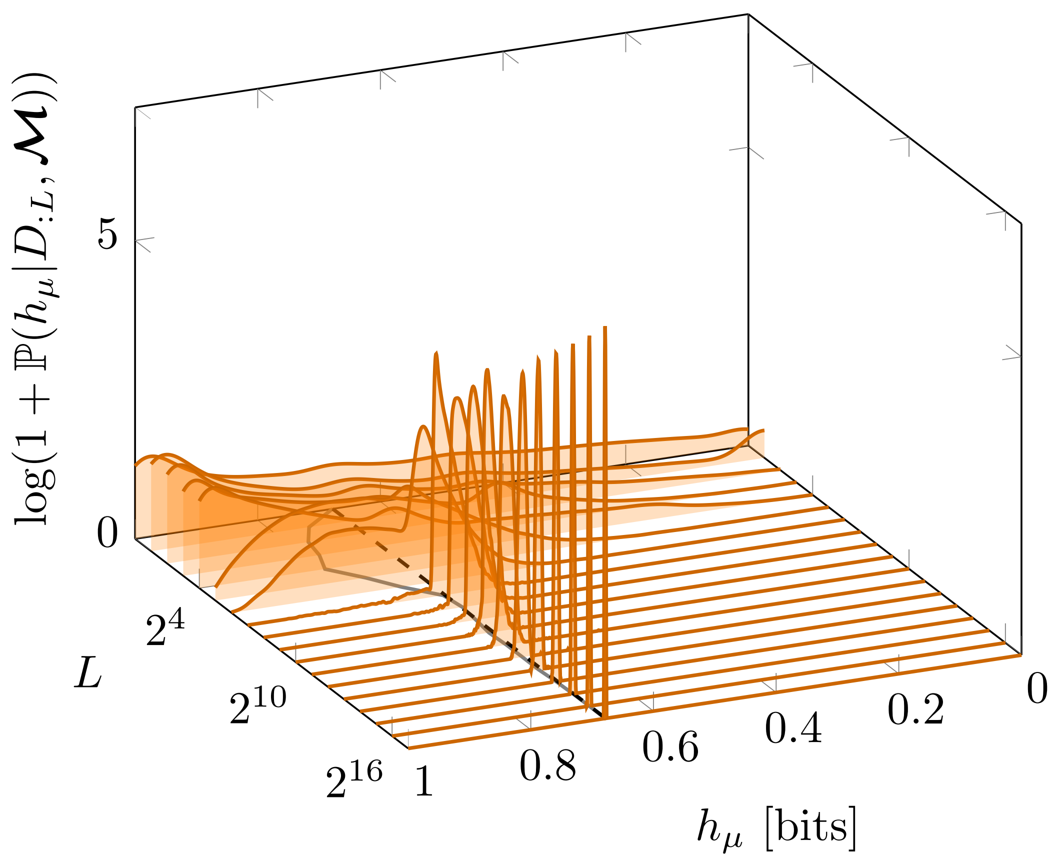
C_μ, h_μ samples: $L = 1$ (black), $L = 64$ (brown), $L = 16\,384$ (blue)



Posterior over models, SNS data

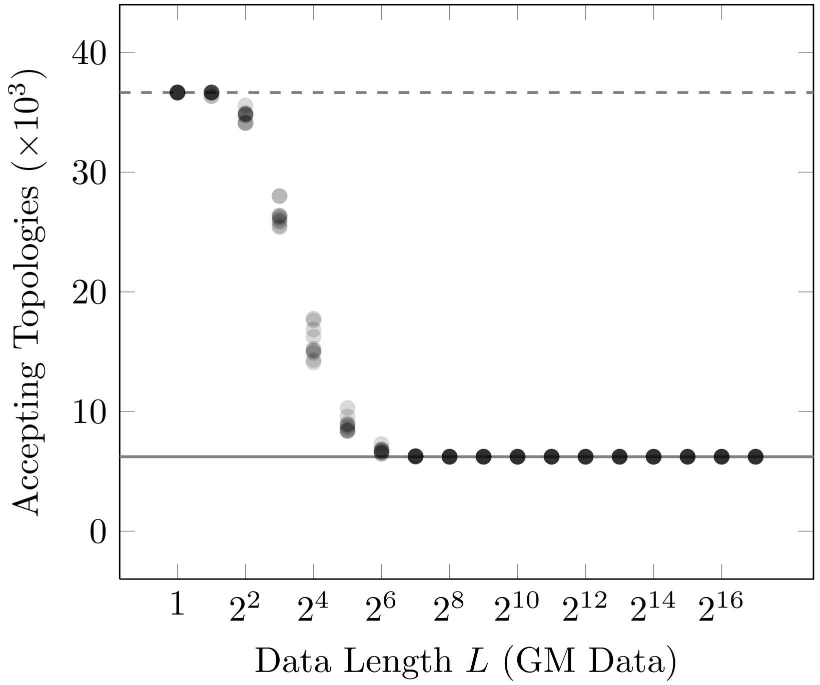
C_μ, h_μ convergence

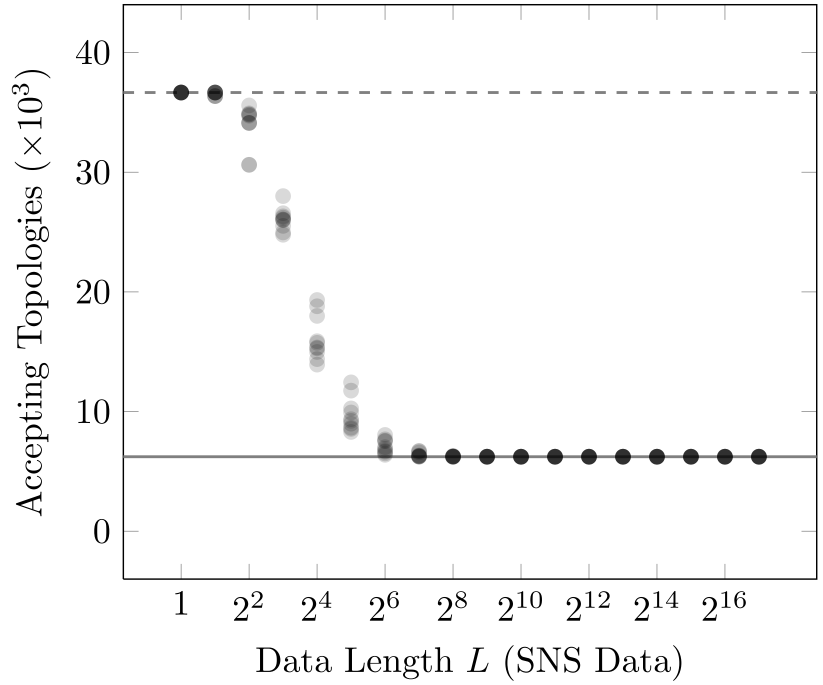


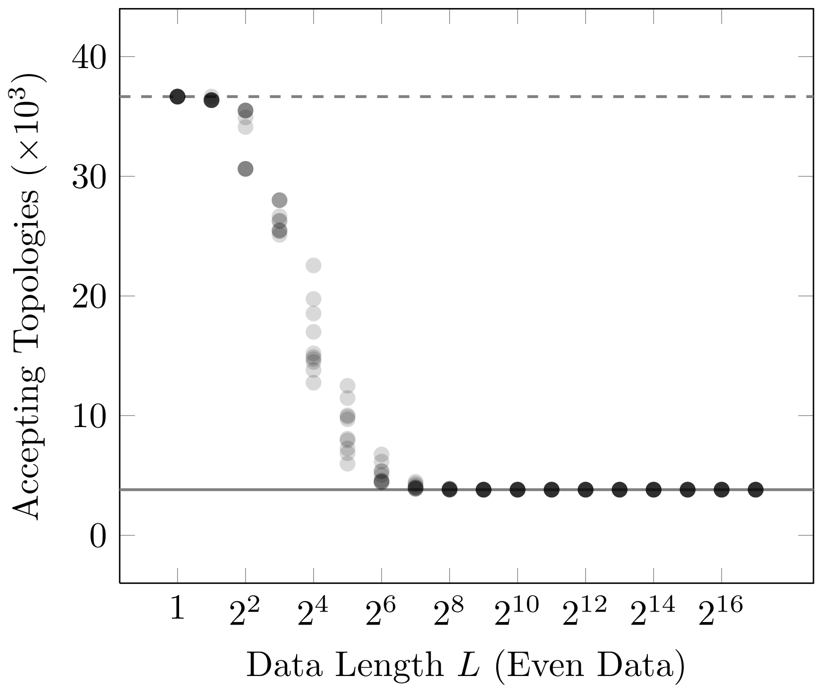


Accepting Topologies

How many topologies accept data from GM, Even, SNS?







- GM, SNS: $6\,225$ of $36\,660$ accept data
- Even: $3\,813$ of $36\,660$ accept data

Complications

Things to think about. . .

- True model topology may not be in $\mathcal{M}$
  - We've seen analysis of SNS data– out-of-class
  - We might also not have machines with enough states in our set
- We assume stationary process– structure and transition probabilities do not change with time/space
  - Resulting inference might be unclear
- Many others issues
  - Check the inferred model(s) as much as possible!

Ex: Not enough states in $\mathcal{M}$

Posterior given data from EvenOdd Process

Use all 1- to 3-state machines

```
# get set of 1- to 3-state topological epsilon machines
modelset3 = bayesem.LibraryGenerator(2,[1,2,3])
# declare posterior over models
posterior2 = bayesem.ModelComparisonEM(modelset3, eo_data, beta=4.,
                                        verbose=True)
```

```
*Infer Machine Topology (InferEM)
 * Model Prior- beta: 4.00000
 * Inferring all machines...
  ** 86 machines considered, 10 possible
 * Calculating log evidence for all machines...
 * Calculating model probabilities for all machines...
```

Posterior

MAP topology– out-of-class

- Get MAP topology and use posterior mean for transition probabilities
- Average over uncertainty in start state (if any)

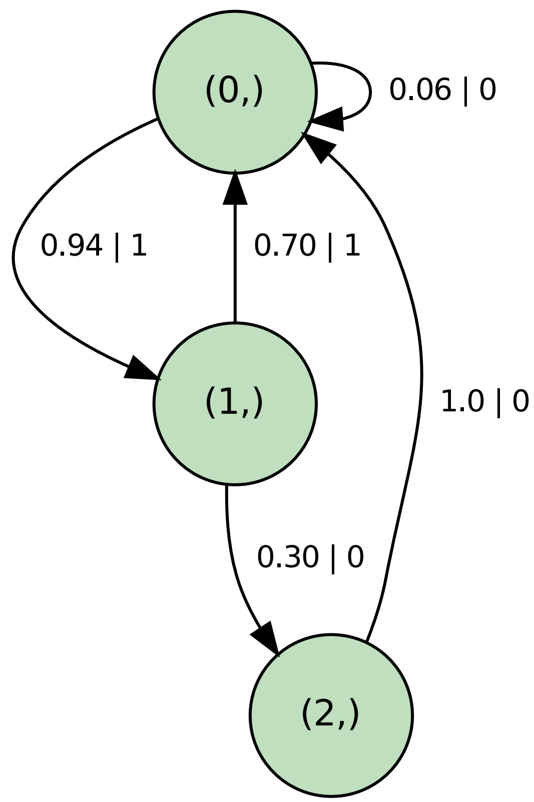
```
# get machine from posterior
pr, eo_post_em = posterior2.get_MAP_PM_machine()[0]

# draw machine
eo_post_em.draw(filename='figures/eo_posterior_mach2.pdf',
                show=False)
```

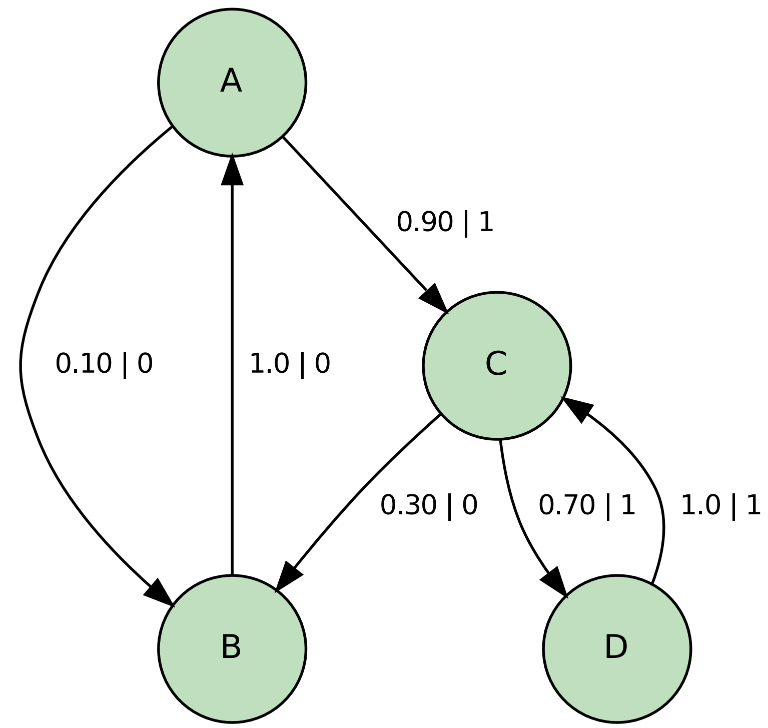
Posterior

MAP topology– out-of-class

Inferred Topology



True Topology



Ex: Non-stationary Process

Make non-stationary: GM then Even

Create data first

```
# get machines
gm = cmpy.machines.GoldenMean()
even = cmpy.machines.Even()

# GM data first, Even Next
ns_data = gm.symbols(4000)
ns_data = ns_data + even.symbols(6000)
```

Posterior given data from non-stationary source

Use all 1- to 3-state machines

```
# get set of 1- to 3-state topological epsilon machines
modelset4 = bayesem.LibraryGenerator(2,[1,2,3])
# declare posterior over models
posterior3 = bayesem.ModelComparisonEM(modelset4, ns_data, beta=4.,
                                        verbose=True)
```

```
*Infer Machine Topology (InferEM)
 * Model Prior- beta: 4.00000
 * Inferring all machines...
  ** 86 machines considered, 3 possible
 * Calculating log evidence for all machines...
 * Calculating model probabilities for all machines...
```

Posterior

MAP topology– non-stationary source

- Get MAP topology and use posterior mean for transition probabilities
- Average over uncertainty in start state (if any)

```
# get machine from posterior
pr, ns_post_em = posterior3.get_MAP_PM_machine()[0]

# draw machine
ns_post_em.draw(filename='figures/ns_posterior_mach.pdf',
                show=False)
```

Posterior

MAP topology– non-stationary source