Quick Guide

I. About the software

This python-based software performs Bayesian inference for general state space models, using SMC^2 method presented in [1]. It returns the log score (the negative log likelihood) and H score (the Hyvarinen score [2], [3], [4]). The states and observations can be either continuous or discrete. This implementation of the SMC^2 engine has been built upon an open source previously developed by Jacob et al. [5]

II. Add user-specific models

First of all, users need to input key parameters in the userfile.py file. Some default values have been already set up. We highlight some important user-specific parameters below.

- MODEL: a string indicating the state space model being used (examples are "locallevel", "hiddenAR", "nestedLGSSM", "kangaroo1", "kangaroo2", "kangaroo3" which we shall elaborate in the next section)
- T: a positive integer indicating the length of sequentially observed data
- UseExistingSynthetic: a boolean (0/1) indicating whether we generate a new synthetic data. If it is 1, then it uses the existing synthetic data associated with MODEL; if it is 0 or there does not existing any synthetic data, it generates new synthetic data and store it in the file "data/MODEL-syntheticdata.R"

Second, if users need to implement a new state-space model that has not been included in the folder, they only need to add two simple files under the subfolder "models/..." We are frequently using "x" and "theta" to denote respectively state particles and parameter particles.

The first file is named "MODELx.py". It contains the following major elements.

- function "firstStateGenerator": generate x-particles at the initial time step
- function "observationGenerator": generate synthetic observations given hidden states
- function "transitionAndWeight": generate x-particles at the next time step, and to compute the log-likelihood of the current observation conditioning on each x-particle and theta-particle.
- function "computeHscore": compute the "compositeHScore", the composite H Score using the method proposed in [4]. We note that another output "simpleHScore" is not used in this version of software.
- constructor "modelx = SSM(name, xdimension, ydimension, HScoreInference, continuousObs)": name is a string indicating the model, xdimension and ydimension are the dimensions of states and

observations, HScoreInference is not used in this version of software and it is set to be zero by default, continuousObs is a boolean (0/1) indicating whether the data is discrete or continuous.

- function "setParameters": (optionally) input true parameters that are used to generate synthetic data, helpful when plotting the final results.

The second file is named "MODELtheta.py". It contains the following major elements.

- field "hyperparameters": parameters to generate theta-particles
- function "logdprior": compute log-likelihood of the generated theta-particles
- function "rprior": generate theta-particles
- constructor "modeltheta = ParameterModel(name, dimension)": name is a string indicating the model, dimension is the number of unknown parameters
- function "setTransformation": types of transformations in a list, corresponding to each parameter to be inferred (e.g., ["log", "none"])

Finally, in the terminal we "cd" to the software folder and run "python launch.py". The results will be automatically saved in the file "results/temp.RData". To plot the results in R, users may follow our sample code written in the file "results/userPlot.R'.

III. Three example state space models

A. The trajectory of Kangaroo population: nested model class

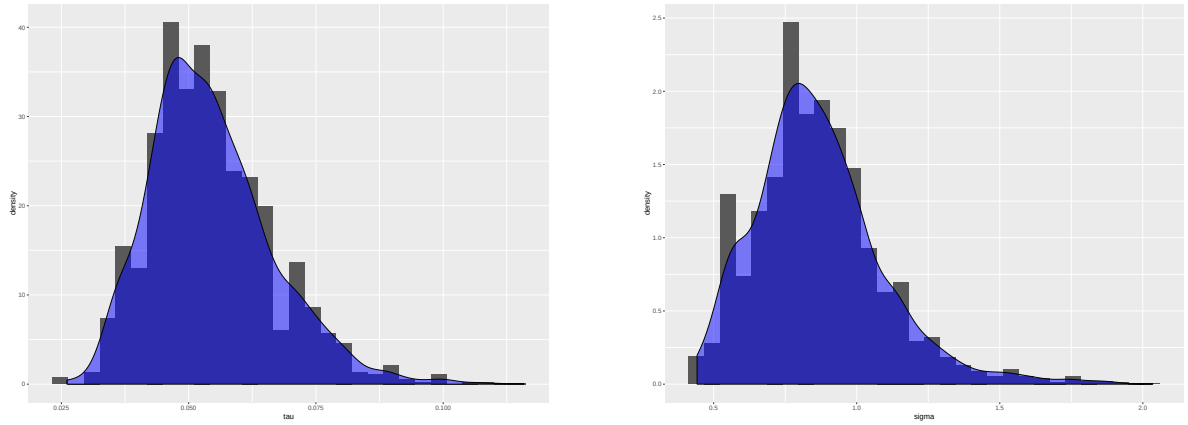
Following the work in [6], we study the log score and H score for three different models under different prior distributions.

Models M1 (logistic), M2 (exponential), M3 (random walk) in consideration are three nested models, with complexity from large to small. The time starts from 1973 Jan, with initial x-particles being generated from $\mathcal{N}(0, 100)$. In model M1, we used Euler discretization with 100 time steps per year. To ensure that the hidden states are always positive, we used log transformation which, by Ito's lemma, leads to the following stochastic equation

$$d\tilde{N}_t = (r - be^{\tilde{N}_t})dt + \sigma dW_t \tag{1}$$

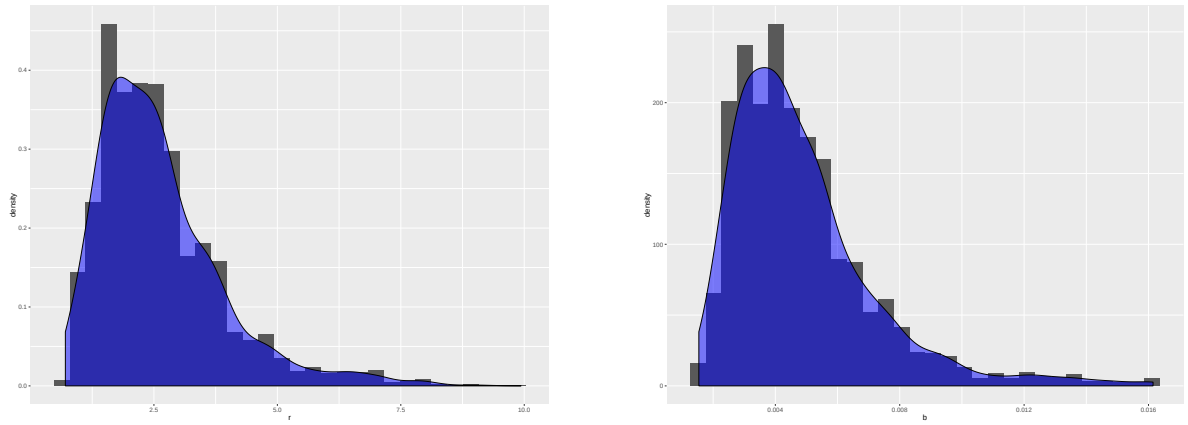
where $\tilde{N}_t \triangleq \log N_t$. The posterior of four parameters of M1 is plotted in Fig. 1, which is consistent with Fig. 1 of [6].

The results of model comparison are summarized in Tab. I. The standard errors of the scores are computed based on four independent experiments (in generating random particles). From the table, both the log and H scores select the smallest model M3. Note that for Unif[*,10] prior, M2 and M3 are not



(a) Posterior summary for τ

(b) Posterior summary for σ



(c) Posterior summary for r

(d) Posterior summary for b

Fig. 1: The computed posterior of τ, σ, r, b in Model M1

TABLE I: Scores given by different models for kangaroo data (where standard errors are in the parenthesis, * means -10 for real value r and 0 for positive τ, σ, b, the values under H score are rescaled by 10^4)

	log score		H score	
	Unif[*,10]	Unif[*,100]	Unif[*,10]	Unif[*,100]
M1	588.11 (10.53)	604.31 (0.23)	-6.88 (7.30)	4.55 (0.05)
M2	554.27 (0.51)	615.44 (1.95)	-30.21 (1.52)	4.23 (0.10)
M3	549.09 (0.34)	604.03 (0.18)	-29.51 (0.11)	3.89 (0.04)

well distinguishable by H score (due to the large Monte Carlo errors), but distinguishable under the flatter Unif[*,100] prior.

B. Local level model: non-overlapping class

In this synthetic data experiment, we consider the following local level model

$$X_t = X_{t-1} + \sigma\epsilon_t$$
$$Y_t = X_t + \tau\eta_t$$

where $t = 1, 2, \ldots$, $X_0 \sim \mathcal{N}(0, 1)$ and ϵ_t, η_t are independent $\mathcal{N}(0, 1)$ noises. The unknown parameters to be inferred are σ^2, τ^2.

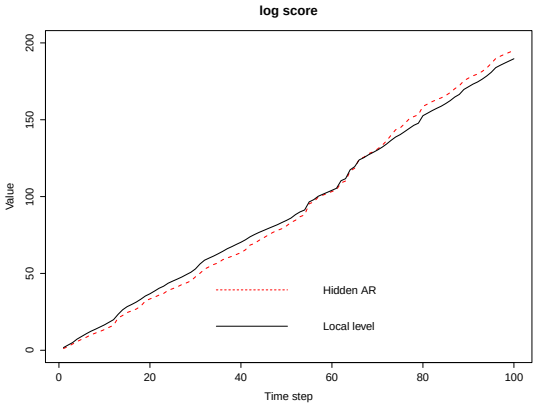
We generated 100 data from true parameters $\sigma^2 = 0.5, \tau^2 = 1$, and then apply our proposed method to compute the scores of two candidate models: first is the well-specified local level model, second is the mis-specified hidden autoregressive model described by

$$X_t = \psi X_{t-1} + 0.5\epsilon_t$$
$$Y_t = X_t + \eta_t$$

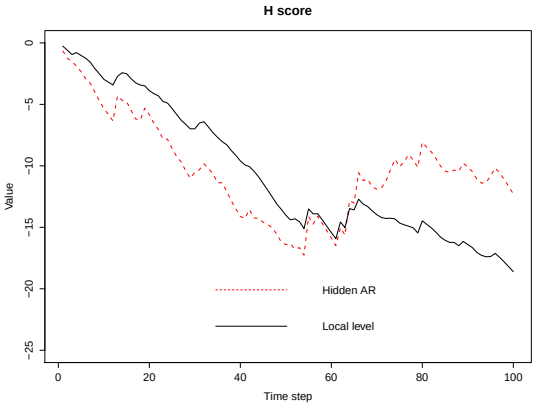
where $t = 1, 2, \ldots$, $X_0 \sim \mathcal{N}(0, 1)$ and ϵ_t, η_t are independent $\mathcal{N}(0, 1)$ noises. The unknown parameter to be inferred is ψ.

For the mis-specified model, we set prior of the parameter ψ to be Unif [0.01,0.99] (and logit transform has been applied). For the well-specified model, the priors are $\sigma^2, \tau \sim$Inv-Gamma with varying shapes and scales. We summarize the scores in Fig. 2.

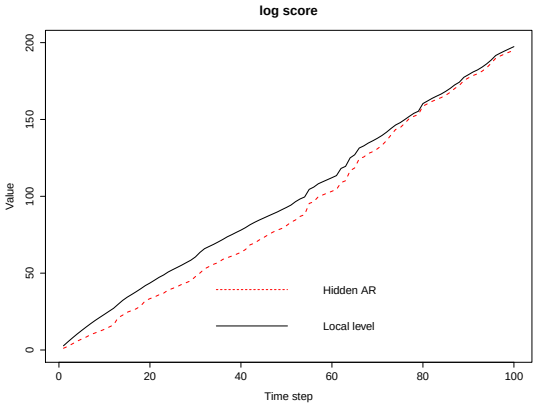
Hidden AR is the smaller model (in terms of dimension), thus flavored for small data size. From the results, log score can barely choose the true model, especially for flatter priors. On the other hand, however, H score is able to quickly identify the true model, and the switching point is insensitive to the specified prior.



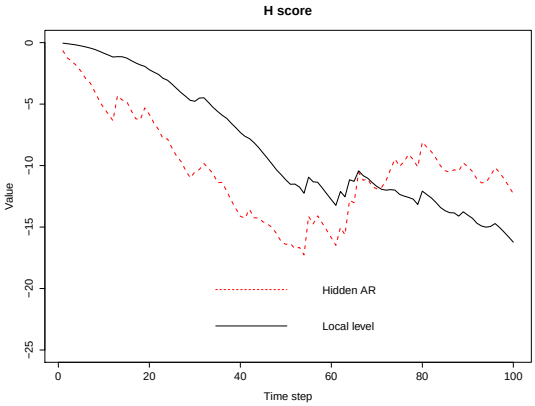
(a) Scale= 1, shape= 1



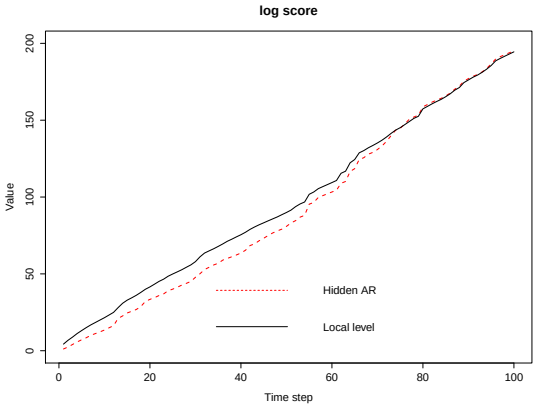
(b) Scale= 1, shape= 1



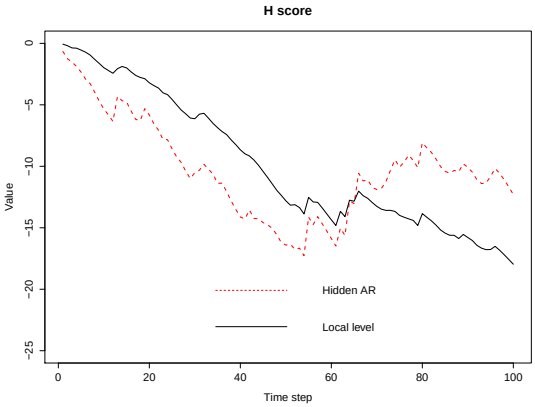
(c) Scale= 0.1, shape= 1



(d) Scale= 0.1, shape= 1



(e) Scale= 1, shape= 0.1



(f) Scale= 1, shape= 0.1

Fig. 2: Model comparison under log score (left) and H score (right), given different priors

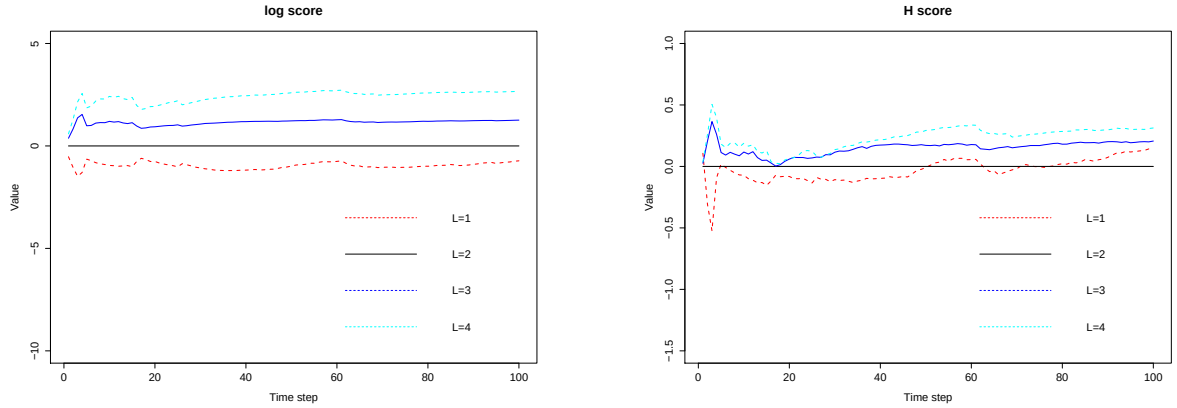
C. Linear Gaussian model: nested class

In this synthetic data experiment, we consider the following linear Gaussian model

$$\boldsymbol{X}_t = \boldsymbol{X}_{t-1} + \boldsymbol{\eta}_t, \quad Y_t = \sum_{d=1}^{L} \psi_d X_{t,d} + \varepsilon_t, \tag{2}$$

where $t = 1, 2, \ldots$, $\boldsymbol{X}_0 \sim \mathcal{N}(0, I_4)$, $\boldsymbol{\eta}_t \sim \mathcal{N}(0, I_4)$ and $\varepsilon_t \sim \mathcal{N}(0, 0.5)$ are independent noises. The unknown parameters are $\{\psi_1, \ldots, \psi_L\}$, and the model under comparison are $L = 1, L = 2, L = 3, L = 4$.

We generate true data from $L = 2$ and $\psi_1 = 3, \psi_2 = 5$, namely $\psi_3 = \psi_4 = 0$ in (2). The prior for each unknown parameter is exponential with rate 0.2. The results of model comparison are given in Fig. 3, where the cumulated scores of model L minus those of model $L = 2$ are plotted. Therefore, a line below the black (zero) line indicates that the corresponding model is more flavored compared with the true model. From the results, log score wrongly selects the model $L = 1$, while H score correctly selects the true model $L = 2$ within 100 data points.



(a) log score (b) H score

Fig. 3: Model comparison under log score (left) and H score (right), given different priors

REFERENCES

[1] N. Chopin, P. Jacob, O. Papaspiliopoulos *et al.*, "Smc2: A sequential monte carlo algorithm with particle markov chain monte carlo updates," *JR Stat. Soc. B*, 2011.

[2] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 695–709, 2005.

[3] A. P. Dawid, M. Musio *et al.*, "Bayesian model selection based on proper scoring rules," *Bayesian Analysis*, vol. 10, no. 2, pp. 479–499, 2015.

[4] S. Shao, J. Ding, V. Tarokh, and P. Jacob, "Model selection for state-space models," 2017.

[5] P. Jacob, "py-smc2: a python package for smc2," *https://sites.google.com/site/pierrejacob/software*, 2017.

[6] J. Knape and P. De Valpine, "Fitting complex population models by combining particle filters with markov chain monte carlo," *Ecology*, vol. 93, no. 2, pp. 256–263, 2012.