EXERCISE ON MCMC FILTERING

(1) (*JFV problem*) R code that can be used to solve this problem is available on the course website, as `pf.R` and `kf2.R`. The particle filter code just does the filter, under the model's distributional assumptions. You can write your own code for the problem if you like. If you use the R code, you will need in part 1h to understand enough of it to make the modification to it that is suggested in that part. In R, to set the random number seed so you ensure random number sequences are different, you use `set.seed()`. R can generate t-distributed random variables with the `rt()` function and evaluate the t density with the `dt()` function.

Consider the following state space model:

$$s_t = \rho s_{t-1} + \varepsilon_t + \eta \varepsilon_{t-1}, \text{ where } \varepsilon_t \sim \mathcal{N}(0, \sigma_{\sigma_\varepsilon})$$

and

$$y_t = \beta s_t + v_t, \text{ where } v_t \sim \mathcal{N}(0, \sigma_v).$$

As usual, we are assuming that ε is the innovation (one-step-ahead prediction error) in s and v is serially uncorrelated and uncorrelated with ε and s at all leads and lags.

Assume the following parametrization:

ρ	η	β	σ_ε	σ_v
0.9	-0.5	1.3	0.5	1

(a) Generate a sample of 200 observations from the model using $s_0 = \varepsilon_{t-1} = 0$.
(b) Use the Kalman filter to evaluate the likelihood function given the parameterization above and the random sample from step 1, and treating the initial conditions as known and non-random (i.e., conditioning on s_0 and ε).
(c) Use the particle filter with 10,000 particles to evaluate the likelihood.
(d) Repeat the exercise in step 1c 25 times, with different seeds for the random number generator of the simulation (note: you are keeping the sample in step 1 fixed. What you are doing is generating different particles, different resampling with replacement, etc). This will give you 25 different evaluations of the likelihood.
(e) Compare the exact evaluation you got from the Kalman filter in step 1b with the 25 evaluations of step 1d.
(f) Repeat exercises 1c to 1e with 50,000 particles. How does your answer in 1e change?
(g) (*CAS addition*) (This part of the question is not required if you have written your own code.) The `pf()` R code returns, in addition to the likelihood, `w`, `s0`, `s1`, and `yhat`. What are these objects? Plot a histogram for each of the first three. What do we learn from these histograms, if anything? `yhat` contains predictions of `ydata`. Is `yhat[it]` a prediction of `ydata[it]` based on the

whole sample, on the sample dated `it` and earlier, or on the sample dated `it-1` and earlier?

(h) (*CAS addition*) Alter the particle-filter code so that it assumes, instead of normal distributions for ε_t and v_t, t distributions with 5 degrees of freedom, with the same .5 and 1 scale parameters. Repeat the 10,000 particle part of the exercise with this new version of the particle filter, but with the same data (generated with normal distributions) that you used above. Since the true model has normal disturbances, you should get lower likelihood with this false model. Does the particle filter give you an accurate enough estimate of the likelihood to see that the t distribution is not correct? Note that with t shocks, the closed-form evaluation of the likelihood that we got from the Kalman filter is not available.

(i) (Bonus. Worthwhile if you are contemplating applying particle filtering to a serious application sometime soon and have not previously tried parallelizing code.) Try to parallelize your code:
   (i) If you are using `R`, `parallel` is a nice, simple package. Also, you can check `snow`, `snowfall`, and `multicore`.
   (ii) If you are using `Matlab`, the `parallel toolbox` should work nicely for you.
   (iii) If you are using `C++/Fortran`, `OpenMP` should be easier than `MPI` unless you have experience with the later.

(2) Here are some artificial data we'll use to gain insight into ways to model data that may be nasty to handle with usual linear models.

$$x = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 3 \\ 10 \end{bmatrix} \qquad y = \begin{bmatrix} 1 \\ 2 \\ 3 \\ .9 \\ 2.1 \\ 2.9 \\ 1.1 \\ 1.9 \\ 3.1 \\ 4 \end{bmatrix} .$$

(a) Obtain estimates and flat-prior posterior standard errors for α and β in the model

$$y = \alpha + x\beta + \varepsilon ,$$

where $\varepsilon \sim N(0, \sigma^2)$ and σ^2 is unknown.

(b) Use Rubin's original Bayesian bootstrap to obtain sample of 1000 draws from the posterior distribution on α and β, where we define α and β as the least squares fit, i.e.

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = (E[X'X])^{-1} E[X'y] ,$$

where X is the matrix with first column a vector of ones and second column our x above. Rubin's Bayesian bootstrap amounts to assuming an improper "Dirichlet(0)" prior on the probabilities of the 10 points in the observed sample, with density $\prod_{j=1}^{10} p_j^{-1}$. When multiplied by the likelihood, $\prod_{j=1}^{10} p_i$ (for this case with no observations repeating), this generates a uniform posterior for the $\vec{p}$ vector over the unit simplex (the set of $\vec{p}$'s that are non-negative and sum to one). This is also a symmetric Dirichlet(1) distribution, i.e. a Dirichlet($\vec{\theta}$) distribution with all 10 elements of the $\vec{\theta}$ vector equal to one. To sample from the posterior, you need to obtain 1000 draws from this posterior and for each draw calculate the implied value of the parameters from (2b).

(c) Repeat the calculations in the previous part, but now using a proper uniform prior on $\vec{p}$ (so that the prior times likelihood becomes a symmetric Dirichlet with parameter 2).

(d) Form 1000 draws from the frequentist bootstrap distribution for the OLS estimator of the regression parameters with this sample. That is, construct 1000 artificial samples from a distribution that puts equal weight on the 10 observed sample points, and for each artificial sample construct the OLS estimator.

(e) Construct a scatter plot of the α draws against the corresponding β draws for each of the three preceding exercises. Comment on what differences you see, and see if you can explain them.

(f) Try adding an x^2 term to the regression matrix. Estimate this expanded model by least squares. Plot the implied regression line (as a smooth line, with finely spaced values of x) and on the same graph a scatter of actual x against y (as points). This model will look like it fits mighty well. However, the posterior odds necessarily favor the Bayesian bootstrap (with proper prior, of course) by infinity to one. Explain why, and why this is an implicit criticism of both the Bayesian and frequentist bootstrap.