BONUS question: LGSSMs, EM and SSID.

Download the datafiles `ssm_spins.txt` and `ssm_spins_test.txt`. Both have been generated by an LGSSM:

$$\mathbf{y}_t \sim \mathcal{N}(A\mathbf{y}_{t-1}, Q) \quad [t = 2 \dots T] \qquad \mathbf{y}_1 \sim \mathcal{N}(0, I)$$
$$\mathbf{x}_t \sim \mathcal{N}(C\mathbf{y}_t, R) \quad [t = 1 \dots T]$$

using the parameters:

$$A = 0.99 \begin{pmatrix} \cos(\frac{2\pi}{180}) & -\sin(\frac{2\pi}{180}) & 0 & 0 \\ \sin(\frac{2\pi}{180}) & \cos(\frac{2\pi}{180}) & 0 & 0 \\ 0 & 0 & \cos(\frac{2\pi}{90}) & -\sin(\frac{2\pi}{90}) \\ 0 & 0 & \sin(\frac{2\pi}{90}) & \cos(\frac{2\pi}{90}) \end{pmatrix} \qquad Q = I - AA^{\mathsf{T}}$$

$$C = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0.5 & 0.5 & 0.5 & 0.5 \end{pmatrix} \qquad R = I$$

but different random seeds. We shall use the first as a training data set and the second as a test set.

(a) Run the function `ssm_kalman.m` we have provided (or a re-implementation in your favourite language if you prefer) on the training data. Warning: the function expects data vectors in columns; you will need to transpose the loaded matrices!

Make the following plots:

```
logdet = @(A)(2*sum(log(diag(chol(A)))));
[Y,V,~,L] = ssm_kalman(X',Y0,Q0,A,Q,C,R, 'filt');
plot(Y'):
plot(cellfun(logdet,V));

[Y,V,Vj,L] = ssm_kalman(X',Y0,Q0,A,Q,C,R, 'smooth');
plot(Y'):
plot(cellfun(logdet,V));
```

Explain the behaviour of `Y` and `V` in both cases (and the differences between them).

(b) Write a function to learn the parameters A, Q, C and R using EM (we will assume that the distribution on the first state is known *a priori*). The M-step update equations for A and C were derived in lecture. You should show that the updates for R and Q can be written:

$$R_{\text{new}} = \frac{1}{T}\Big[\sum_{t=1}^{T} \mathbf{x}_t\mathbf{x}_t^{\mathsf{T}} - \Big(\sum_{t=1}^{T} \mathbf{x}_t\langle\mathbf{y}_t^{\mathsf{T}}\rangle\Big)C_{\text{new}}^{\mathsf{T}}\Big]$$
$$Q_{\text{new}} = \frac{1}{T-1}\Big[\sum_{t=2}^{T} \langle\mathbf{y}_t\mathbf{y}_t^{\mathsf{T}}\rangle - \Big(\sum_{t=2}^{T} \langle\mathbf{y}_t\mathbf{y}_{t-1}^{\mathsf{T}}\rangle\Big)A_{\text{new}}^{\mathsf{T}}\Big]$$

where C_{new} and A_{new} are as in the lecture notes. Store the log-likelihood at every step (easy to compute from the fourth value returned by `ssm_kalman`) and check that it increases.

[Hint: the matlab code

```
cellsum=@(C)(sum(cat(3,C{:}),3))
```

defines an inline function `cellsum()` that sums the entries of a cell array of matrices.]

Run at least 50 iterations of EM starting from a number of different initial conditions: (1) the generating parameters above (why does EM not terminate immediately?) and (2) 10 random choices.

Show how the likelihood increases over the EM iterations (hand in a plot showing likelihood vs iteration for each run, plotted in the same set of axes). Explain the features of this plot that you think are salient.

[If your code and/or computer is fast enough, try running more EM iterations. What happens?]

(c) Write a function to learn A, Q, C and R using Ho-Kalman SSID. You should provide options to specify the maximum lag to include in the future-past Hankel matrix (H), and the latent dimensionality. You may like to plot the singular value spectrum of H to see how good a clue it provides to dimensionality.

[Hints: if `M` is a cell array of the lagged correlation matrices M_τ, you can build the Hankel matrix using:

```
H = cell2mat(M(hankel([1:maxlag], [maxlag:2*maxlag-1])));
```

The matrices Ξ (`Xi`) and Υ (`Ups`) are found using SVD (assuming `K` latent dimensions):

```
[Xi,SV,Ups] = svds(H, K);

Xi = Xi*sqrt(SV);
Ups = sqrt(SV)*Ups';
```

and these can be used to find $\widehat{C}$ and $\widehat{A}$ and $\widehat{\Pi}$ by regression (`/` or `\` in MATLAB), e.g.:

```
Ahat = Xi(1:end-DD,:)\Xi(DD+1:end,:);
```

(The code for C and Π will take a little more thought!).

To find $\widehat{Q}$ and $\widehat{R}$ recall that, as Π is the stationary (prior) latent covariance:

$$A\Pi A^{\mathsf{T}} + Q = \Pi$$

and

$$C\Pi C^{\mathsf{T}} + R = \mathbb{E}_{t\to\infty}\left[\mathbf{x}_t\mathbf{x}_t^{\mathsf{T}}\right]$$

Run your code on the training data using a maximum lag of 10 and the true latent dimensionality. Evaluate the likelihood of this model using `ssm_kalman.m` and show it on the likelihood figure. Now use the SSID parameters as initial values for EM and show the resulting likelihood curve.

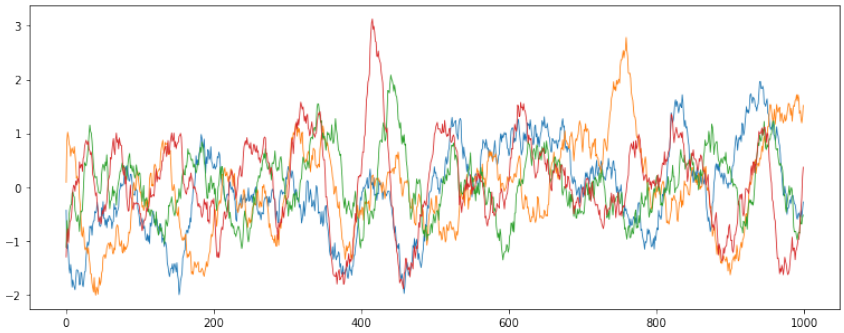
Comment on the advantages and disadvantanges of SSID as compared to EM.

(If you have time and inclination, you might like to explore how your results vary with different choices of lag and dimensionality)

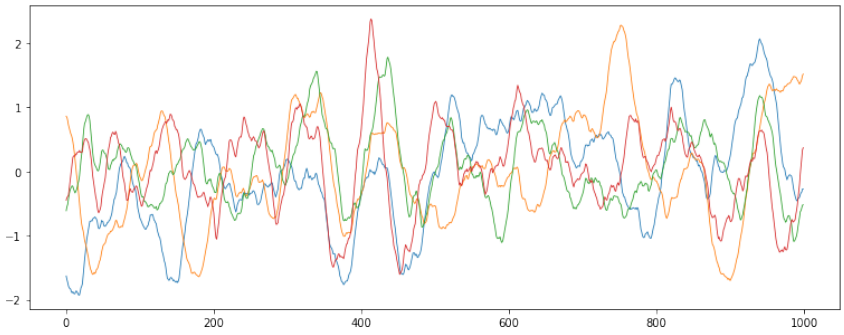
(d) Evaluate the likelihood of the test data under the true parameters, and all of the parameters found above (EM initialised at the true parameters, random parameters and the SSID parameters, as well as the SSID parameters without EM). Show these numbers on or next to the training data likelihoods plotted above. Comment on the results.

(a)

The plot of $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:t}]$ and the log determinant of $\mathrm{Cov}\,[\mathbf{y}_t|\mathbf{x}_{0:t}]$ (Kalman Filter case) are shown in Figure 10-(a) and Figure 11-(a) respectively. The plot of $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:T}]$ and the log determinant of $\mathrm{Cov}\,[\mathbf{y}_t|\mathbf{x}_{0:T}]$ (Kalman Smoothing case) are shown in Figure 16-(b) and Figure 17-(b) respectively.



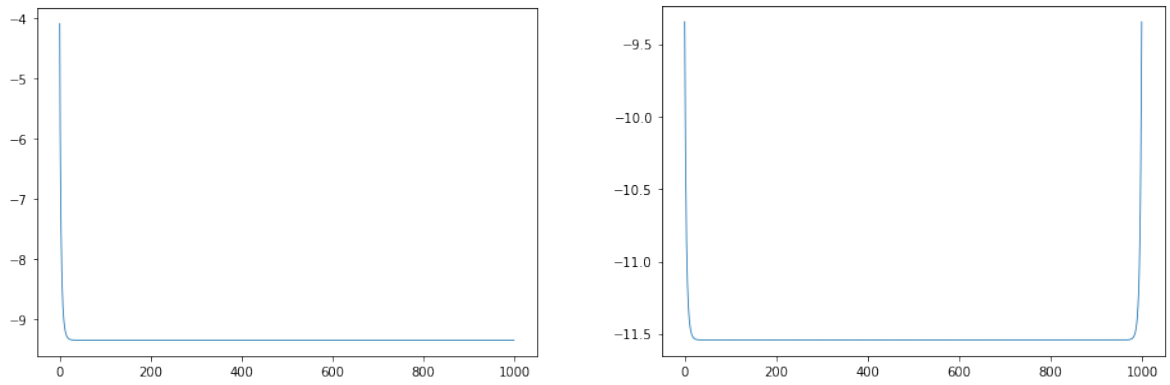
(a)



(b)

Figure 16: (a) $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:t}]$ obtained in Kalman Filtering; (b) $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:T}]$ obtained in Kalman Smoothing

It can been seen from Figure 10 that, while the overall shape of the lines of $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:t}]$ and $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:T}]$ is quite similar, the smoothness is rather different. The path of $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:t}]$ is much more zigzag and the path of $\mathrm{E}[\mathbf{y}_t|\mathbf{x}_{0:T}]$ is more smoothed.



(a) (b)

Figure 17: (a) log determinant of Cov $[\mathbf{y}_t|\mathbf{x}_{0:t}]$ obtained in Kalman Filtering; (b) log determinant of Cov $[\mathbf{y}_t|\mathbf{x}_{0:t}]$ obtained in Kalman Smoothing

From Figure 11, in the Kalman Filter case, the log determinant of Cov $[\mathbf{y}_t|\mathbf{x}_{0:t}]$ converges quickly, and the converged value is -9.3. In the Kalman Smoothing case, the log determinant of Cov $[\mathbf{y}_t|\mathbf{x}_{0:T}]$ also converges quickly, but diverges in the end, and the converged value is -11.5. This shows that the estimated covariance in the smoothing case is much smaller than in the filtering case.

The differences between the two cases can be explained by the difference in conditioning observation. To be specific, in the Kalman Filtering process, only the observation up to date $\mathbf{x}_{0:t}$ is incorporated as conditioning information, while in the Kalman Smoothing process, the whole observed sequence $\mathbf{x}_{0:T}$ is incorporated. As we have more information in Kalman Smoothing, the estimation of $\mathrm{E}[\mathbf{y}_t]$ and Cov $[\mathbf{y}_t]$ becomes more stable and thus smaller variance.

(b)

Free energy is

$$\mathcal{F}(q,\theta) = \int d\mathbf{y}_{1:T} q(\mathbf{y}_{1:T})(\log P(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}|\theta) - \log q(\mathbf{y}_{1:T}))$$

In the M Step, all the relevant variables are in $p(\mathbf{y}, \mathbf{x}|\theta)$.

[Update R_{new}]

Maximize $\mathcal{F}$ w.r.t Q is equivalent to $\max_Q \left\langle \sum_t \log p(\mathbf{x}_t|\mathbf{y}_t) \right\rangle_q$.

Therefore,

$$\begin{aligned} R_{\text{new}} &= \arg\max_R \left\langle -\frac{1}{2}\sum_t (\mathbf{x}_t - C\mathbf{y}_t)^\top R^{-1}(\mathbf{x}_t - C\mathbf{y}_t) + \sum_t \log\left(\det(R)\right)^{-\frac{1}{2}} \right\rangle_q + \text{const} \\ &= \arg\max_R -\frac{1}{2}\sum_t \mathbf{x}_t{}^\top R^{-1}\mathbf{x}_t + \text{Tr}\left(\sum_t \mathbf{x}_t{}^\top R^{-1} C\langle \mathbf{y}_t\rangle_q\right) \\ &\quad -\frac{1}{2}\text{Tr}\left(\sum_t R^{-1} C\langle \mathbf{y}_t\mathbf{y}_t{}^\top\rangle_q\right) + \frac{T}{2}\log\det(R^{-1}) \end{aligned}$$

we have

$$\frac{\partial}{\partial R^{-1}} = -\frac{1}{2}\sum_t \mathbf{x}_t\mathbf{x}_t{}^\top + \left(\sum_{t=1}^T \mathbf{x}_t\langle \mathbf{y}_t{}^\top\rangle_q\right) C^\top - \frac{1}{2}C\sum_{t=1}^T \langle \mathbf{y}_t\mathbf{y}_t{}^\top\rangle_q C^\top + \frac{T}{2}R \tag{3}$$

Since

$$\begin{aligned} C_{\text{new}} &= \left(\sum_{t=1}^T \mathbf{x}_t\langle \mathbf{y}_t{}^\top\rangle_q\right)\left(\sum_{t=1}^T \langle \mathbf{y}_t\mathbf{y}_t{}^\top\rangle_q\right)^{-1} \\ \sum_{t=1}^T \langle \mathbf{y}_t\mathbf{y}_t{}^\top\rangle_q &= C_{\text{new}}^{-1}\left(\sum_{t=1}^T \mathbf{x}_t\langle \mathbf{y}_t{}^\top\rangle_q\right) \end{aligned} \tag{4}$$

Substitute (4) into (3), we get

$$R_{\text{new}} = \frac{1}{T}\left[\sum_{t=1}^T \mathbf{x}_t\mathbf{x}_t{}^\top - \left(\sum_{t=1}^T \mathbf{x}_t\langle \mathbf{y}_t{}^\top\rangle\right) C_{\text{new}}^\top\right]$$

[Update Q_{new}]

Maximize $\mathcal{F}$ w.r.t Q is equivalent to $\max_Q \left\langle \sum_t \log p(\mathbf{y}_t|\mathbf{y}_{t-1}) \right\rangle_q$.

Therefore,

$$\begin{aligned} Q_{\text{new}} &= \arg\max_Q \left\langle -\frac{1}{2}\sum_t (\mathbf{y}_t - A\mathbf{y}_{t-1})^\top Q^{-1}(\mathbf{y}_t - A\mathbf{y}_{t-1}) + \sum_t \log\left(\det(Q)\right)^{-\frac{1}{2}} \right\rangle_q + \text{const} \\ &= \arg\max_Q -\frac{1}{2}\sum_t \left\langle \mathbf{y}_t{}^\top Q^{-1}\mathbf{y}_t \right\rangle_q + \text{Tr}\left(\sum_t \left\langle \mathbf{y}_t{}^\top Q^{-1} A\mathbf{y}_{t-1}\right\rangle_q\right) \\ &\quad -\frac{1}{2}\text{Tr}\left(\sum_t Q^{-1}A\left\langle \mathbf{y}_{t-1}\mathbf{y}_{t-1}{}^\top\right\rangle_q\right) + \frac{T-1}{2}\log\det(Q^{-1}) \end{aligned}$$

we have

$$\frac{\partial}{\partial Q^{-1}} = -\frac{1}{2}\sum_{t=2}^T \left\langle \mathbf{y}_t\mathbf{y}_t{}^\top \right\rangle_q + \left(\sum_{t=2}^T \left\langle \mathbf{y}_t\mathbf{y}^\top{}_{t-1}\right\rangle_q\right) A^\top - \frac{1}{2}A\sum_{t=2}^T \left\langle \mathbf{y}_{t-1}\mathbf{y}_{t-1}^\top\right\rangle_q A^\top + \frac{T-1}{2}Q \tag{5}$$

Since

$$A_{\text{new}} = \left(\sum_{t=2}^{T}\left\langle \mathbf{y}_t \mathbf{y}^{\top}{}_{t-1}\right\rangle_q\right)\left(\sum_{t=2}^{T}\left\langle \mathbf{y}_{t-1}\mathbf{y}_{t-1}^{\top}\right\rangle_q\right)^{-1}$$

$$\left(\sum_{t=2}^{T}\left\langle \mathbf{y}_{t-1}\mathbf{y}_{t-1}^{\top}\right\rangle_q\right) = A_{\text{new}}^{-1}\left(\sum_{t=2}^{T}\left\langle \mathbf{y}_t \mathbf{y}^{\top}{}_{t-1}\right\rangle_q\right) \tag{6}$$

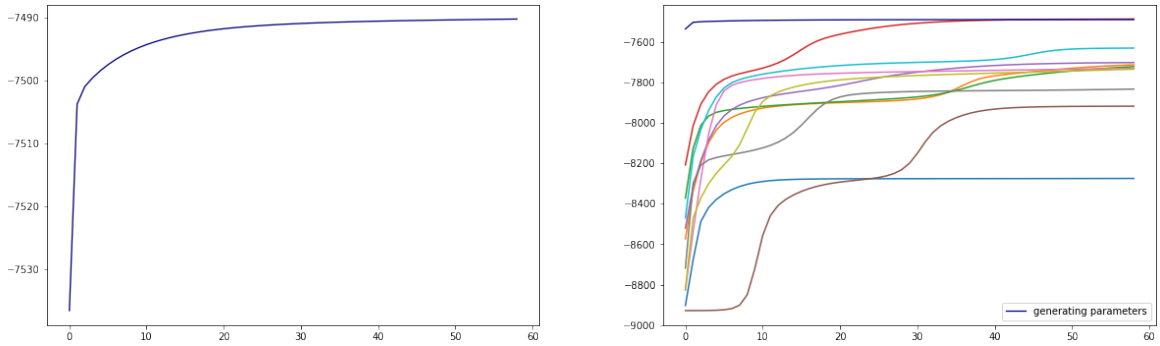
Substitute (6) into (5), we get

$$Q_{\text{new}} = \frac{1}{T-1}\left[\sum_{t=2}^{T}\left\langle \mathbf{y}_t \mathbf{y}_t^{\top}\right\rangle - \left(\sum_{t=2}^{T}\left\langle \mathbf{y}_{t-1}\mathbf{y}_{t-1}^{\top}\right\rangle\right) A_{\text{new}}^{\top}\right]$$

When running 60 iterations of EM algorithm starting from (1) the generating parameters and (2) 10 random choices of parameters, we get the following results of log likelihood as are shown in Figure 18.

As can be seen from Figure 18(b), when using the generating parameters as initial condition, the log likelihood starts at a relatively larger value than those obtained when using random choices as initial conditions. On the other hand, the increase rate of log likelihood under the case of using random initialization is much higher than under the case of using generating parameters. Thus, overall, the speeds of convergence in both cases seem to be quite similar.

Furthermore, a distinctive trait of the log likelihood curves obtained when using random initialization choices is that, some curves are rather 'wavy', meaning if we increase the number of iterations, log likelihood might get out of a local maxima and continues to climb to another local maxima.



(a) (b)

Figure 18: (a) Log likelihoods obtained when starting from (1) the generating parameters; (b) Comparison of log likelihoods obtained when starting from (2) 10 random choices of parameters and (1)