Notations (common)

Each row is the recording for neuron i, $\boldsymbol{y}_i = (y_{i1}, \dots, y_{iT})'$, $i = 1, \dots N$. Denote the cluster index for neuron i as $z_i \in \{1, \dots\}$. The number of neurons in cluster j is $n_j = \sum_{i=1}^{N} I(z_i = j)$, and $\sum_{j=1,2,\dots} n_j = N$.

Model

Denote the latent vector in cluster j as $\boldsymbol{x}_t^{(j)} \in R^{p_j}$. For simplicity, assume all $p_j = p$. Each observation follows a Poisson distribution as follows:

$$\log \lambda_{it} = d_i + \boldsymbol{c}_i' \boldsymbol{x}_t^{(z_i)}$$

$$y_{it} \sim Poisson(\lambda_{it})$$

Where $\boldsymbol{c}_i \in R^p$ and $\boldsymbol{x}_t^{(z_i)} \in R^p$.

Denote all latent states as $\boldsymbol{x}_t = \left(\boldsymbol{x}_t^{(1)'}, \boldsymbol{x}_t^{(2)'}, \dots\right)'$ and they evolve linearly with Gaussian noise:

$$\boldsymbol{x}_1 \sim N(\boldsymbol{x}_0, \boldsymbol{Q}_0)$$

$$\boldsymbol{x}_{t+1} | \boldsymbol{x}_t \sim N(\boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{b}, \boldsymbol{Q})$$

To simplify, assume $\boldsymbol{Q}_0$ is known (e.g. $\boldsymbol{Q}_0 = \boldsymbol{I}$).

If we assume block diagonal (as in Joshua et al., 2020) for process noise covariance, we can write things as:

$$\boldsymbol{x}_{t+1}^{(j)} | \boldsymbol{x}_t^{(1)}, \boldsymbol{x}_t^{(2)}, \dots \sim N\left(\sum_{l=1,\dots} \boldsymbol{A}_{j \leftarrow l} \boldsymbol{x}_t^{(l)} + \boldsymbol{b}_j, \boldsymbol{Q}^{(j)}\right)$$

Notice $\{\boldsymbol{A}_{j \leftarrow l}\}$ forms the full transition matrix as:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_{1\leftarrow 1} & \boldsymbol{A}_{1\leftarrow 2} & \dots \\ \boldsymbol{A}_{2\leftarrow 1} & \boldsymbol{A}_{2\leftarrow 2} & \dots \\ \dots & \dots & \dots \end{pmatrix}$$

If the j^{th} row block of $\boldsymbol{A}$ is $\boldsymbol{A}_j = (\boldsymbol{A}_{j\leftarrow 1} \quad \boldsymbol{A}_{j\leftarrow 2} \quad \dots)$. Then, $\sum_{l=1,\dots} \boldsymbol{A}_{j \leftarrow l} \boldsymbol{x}_t^{(l)} + \boldsymbol{b}_j = \boldsymbol{A}_j \boldsymbol{x}_t + \boldsymbol{b}_j$.

If we further let $\boldsymbol{Q}$ be diagonal: denote the k^{th} row of $\boldsymbol{x}_t, \boldsymbol{A}, \boldsymbol{b}$ as $x_{kt}, \boldsymbol{a}_k, b_k$. The corresponding process noise variance as q_k. Then:

$$x_{k,t+1} | x_{kt} \sim N(\boldsymbol{a}_k' \boldsymbol{x}_t + b_k, q_k)$$

The parameters need to estimate:

(1) Latent vectors: $\boldsymbol{x}_t$
(2) Initials: $\boldsymbol{x}_0$
(3) Linear mapping for latent vectors: $\{d_i\}_{i=1}^N$ and $\{\boldsymbol{c}_i\}_{i=1}^N$
(4) Linear dynamics for latent vectors: $\boldsymbol{A}$ and $\boldsymbol{b}$
(5) Process noise: $\boldsymbol{Q}$

Since the progress noise is independent in the model, $f(\boldsymbol{y}_i|\boldsymbol{\Theta}_j) = \prod_{t=1}^{T} P(y_{it}|\boldsymbol{\Theta}_j)$, where $P(\cdot)$ is the Poisson density and $\boldsymbol{\Theta}_j$ is the parameters in cluster j.

Conditional Priors

(1) Latent vectors $\boldsymbol{x}_t^{(j)}$: the conditional prior is defined by
(Assume there are J clusters)

$$\boldsymbol{x}_1 \sim N(\boldsymbol{x}_0, \boldsymbol{Q}_0)$$

$$\boldsymbol{x}_{t+1}|\boldsymbol{x}_t \sim N(\boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{b}, \boldsymbol{Q})$$

(2) Initials $\boldsymbol{x}_0$:
(Assume there are J clusters)

$$\boldsymbol{x}_0 \sim N(\boldsymbol{\mu}_{x_{00}}, \boldsymbol{\Sigma}_{x_{00}})$$

$\boldsymbol{\mu}_{x_{00}} = \boldsymbol{0}_{Jp}$ and $\boldsymbol{\Sigma}_{x_{00}} = 10^2 \boldsymbol{I}_{Jp}$

(3) Linear mapping for latent vectors $\{d_i\}_{i=1}^{N}$ and $\{\boldsymbol{c}_i\}_{i=1}^{N}$:

$$(d_i, \boldsymbol{c}_i')' \sim N(\boldsymbol{\mu}_{dc0}, \boldsymbol{\Sigma}_{dc0})$$

Where $\boldsymbol{\mu}_{dc0} = \boldsymbol{0}_{p+1}$ and $\boldsymbol{\Sigma}_{dc0} = \boldsymbol{I}_{p+1} \times 10^{-2}$

(4) Linear dynamics for latent vectors $\boldsymbol{A}$ and $\boldsymbol{b}$:
(if the number of cluster is J)

a. Assume $\boldsymbol{Q}$ be diagonal:

$$(b_k, \boldsymbol{a}_k')' \sim N(\boldsymbol{\mu}_{ba_k0}, \boldsymbol{\Sigma}_{ba0})$$

Where $\boldsymbol{\mu}_{ba_k0} = (0, \boldsymbol{e}_k')'$ and $\boldsymbol{\Sigma}_{ba0} = 0.25\boldsymbol{I}_{Jp+1}$

b. Assume $\boldsymbol{Q}$ be block-diagonal:

Denote $\widetilde{\boldsymbol{A}}_j = vec(\boldsymbol{A}_j)$

$$(\boldsymbol{b}_j', \widetilde{\boldsymbol{A}}_j')' \sim N(\boldsymbol{\mu}_{bA_0}, \boldsymbol{\Sigma}_{bA_0})$$

Where $\boldsymbol{\mu}_{bA_0} = \left(\boldsymbol{0}'_{p+(j-1)p^2}, vec(\boldsymbol{I}_p)', \boldsymbol{0}'_{(J-j)p^2}\right)'$ and $\boldsymbol{\Sigma}_{bA_0} = 0.25\boldsymbol{I}_{p(1+pJ)}$

c. No constraints on $\boldsymbol{Q}$:

Denote $\widetilde{\boldsymbol{A}} = vec(\boldsymbol{A})$

$$(\boldsymbol{b}', \widetilde{\boldsymbol{A}})' \sim N(\boldsymbol{\mu}_{BA_0}, \boldsymbol{\Sigma}_{BA_0})$$

Where $\boldsymbol{\mu}_{BA_0} = \left(\boldsymbol{0}'_{Jp}, vec(\boldsymbol{I}_{Jp})'\right)'$ and $\boldsymbol{\Sigma}_{BA_0} = 0.25\boldsymbol{I}_{Jp(1+Jp)}$

(5) Process noise $\boldsymbol{Q}$:

a. Assume $\boldsymbol{Q}$ be diagonal:

$$q_k \sim IG(\frac{\nu_0}{2}, \frac{\nu_0 \sigma_0^2}{2})$$

Where $\nu_0 = 4$ and $\sigma_0^2 = 10^{-4}$

b. Assume $\boldsymbol{Q}$ be block-diagonal:

$$\boldsymbol{Q}^{(j)} \sim W^{-1}(\Psi_0, \nu_0)$$

Where $\nu_0 = p + 2$ and $\Psi_0 = \boldsymbol{I}_p \times 10^{-4}$.
(To make the mean of $\boldsymbol{Q}^{(j)}$ loosely centered around $\boldsymbol{I}_p \times 10^{-4}$)

c. No constraints on $\boldsymbol{Q}$:

(if the number of cluster is J)

$$\boldsymbol{Q} \sim W^{-1}(\Psi_{\boldsymbol{Q}_0}, \nu_{\boldsymbol{Q}_0})$$

Where $\nu_{\boldsymbol{Q}_0} = Jp + 2$ and $\Psi_{\boldsymbol{Q}_0} = \boldsymbol{I}_{Jp} \times 10^{-4}$.

MCMC iteration

(1) Update $\boldsymbol{x}_t^{(j)}$:

use local Normal approximation at prior, i.e. update by adaptive smoothing.

Notice: adaptive filtering is not the "exact Laplace approximation". In Laplace approximation, we are evaluating at the mode/ maximum of posterior, but in adaptive filtering, it is evaluated at the prior…

An IMPROTANT drawback for adaptive filtering/ smoothing is that it don't give the covariance between different time steps. This will ruin the sampling and estimations for other parameters. But the mean estimations for adaptive filtering/ smoothing is perfect.

An easy solution: Only use the posterior mean from adaptive smoothing (discard the variance estimation), and then calculate the covariance matrix by the Hessian as follows ("THE EXACT LAPLACE APPROXIMATION").

THE EXACT LAPLACE APPROXIMATION

Denote t^{th} column of mean firing rate and observation as $\tilde{\boldsymbol{\lambda}}_t = (\lambda_{1t}, \dots, \lambda_{Nt})'$ and $\tilde{\boldsymbol{y}}_t = (y_{1t}, \dots, y_{Nt})'$. The linear mapping matrix for all observations is $\boldsymbol{C}$, such that $\log \tilde{\boldsymbol{\lambda}}_t = \boldsymbol{d} + \boldsymbol{C}\boldsymbol{x}_t$. Let $\boldsymbol{x} = (\boldsymbol{x}_1', \dots, \boldsymbol{x}_T')'$and $f(\boldsymbol{x}) = \log P(\boldsymbol{x} | \{\boldsymbol{y}_i\}_{i=1}^N, \boldsymbol{C}, \boldsymbol{Q}_0, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{Q}, \dots)$

The first and second derivative with respect to $\boldsymbol{x}$

$$\frac{\partial f}{\partial \boldsymbol{x}_1} = \boldsymbol{C}'(\tilde{\boldsymbol{y}}_1 - \tilde{\boldsymbol{\lambda}}_1) - \boldsymbol{Q}_0^{-1}(\boldsymbol{x}_1 - \boldsymbol{x}_0) + \boldsymbol{A}'\boldsymbol{Q}^{-1}(\boldsymbol{x}_2 - \boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{b})$$

$$\frac{\partial f}{\partial \boldsymbol{x}_t} = \boldsymbol{C}'(\tilde{\boldsymbol{y}}_t - \tilde{\boldsymbol{\lambda}}_t) - \boldsymbol{Q}^{-1}(\boldsymbol{x}_t - \boldsymbol{A}\boldsymbol{x}_{t-1} - \boldsymbol{b}) + \boldsymbol{A}'\boldsymbol{Q}^{-1}(\boldsymbol{x}_{t+1} - \boldsymbol{A}\boldsymbol{x}_t - \boldsymbol{b})$$

$$\frac{\partial f}{\partial \boldsymbol{x}_T} = \boldsymbol{C}'(\tilde{\boldsymbol{y}}_T - \tilde{\boldsymbol{\lambda}}_T) - \boldsymbol{Q}^{-1}(\boldsymbol{x}_T - \boldsymbol{A}\boldsymbol{x}_{T-1} - \boldsymbol{b})$$

$$\frac{\partial^2 f}{\partial x_1 \partial x_1'} = -\boldsymbol{C}' Diag(\tilde{\boldsymbol{\lambda}}_1)\boldsymbol{C} - \boldsymbol{Q}_0^{-1} - \boldsymbol{A}'\boldsymbol{Q}^{-1}\boldsymbol{A} \qquad \frac{\partial^2 f}{\partial x_1 \partial x_2'} = \boldsymbol{A}'\boldsymbol{Q}^{-1}$$

$$\frac{\partial^2 f}{\partial x_t \partial x_t'} = -\boldsymbol{C}' Diag(\tilde{\boldsymbol{\lambda}}_t)\boldsymbol{C} - \boldsymbol{Q}^{-1} - \boldsymbol{A}'\boldsymbol{Q}^{-1}\boldsymbol{A} \qquad \frac{\partial^2 f}{\partial x_t \partial x_{t-1}'} = \boldsymbol{Q}^{-1}\boldsymbol{A} \qquad \frac{\partial^2 f}{\partial x_t \partial x_{t+1}'} = \boldsymbol{A}'\boldsymbol{Q}^{-1}$$

$$\frac{\partial^2 f}{\partial x_T \partial x_T'} = -\boldsymbol{C}' Diag(\tilde{\boldsymbol{\lambda}}_T)\boldsymbol{C} - \boldsymbol{Q}^{-1}$$

So, the gradient is:

$$\nabla = \frac{\partial f}{\partial \boldsymbol{x}} = \left(\left(\frac{\partial f}{\partial \boldsymbol{x}_1}\right)', \dots, \left(\frac{\partial f}{\partial \boldsymbol{x}_T}\right)'\right)'$$

And the block tri-diagonal Hessian:

$$H = \frac{\partial^2 f}{\partial \boldsymbol{x} \partial \boldsymbol{x}'} = \begin{pmatrix} \frac{\partial^2 f}{\partial \boldsymbol{x}_1 \partial \boldsymbol{x}_1'} & \boldsymbol{A}'\boldsymbol{Q}^{-1} & 0 & \cdots & 0 \\ \boldsymbol{Q}^{-1}\boldsymbol{A} & \frac{\partial^2 f}{\partial \boldsymbol{x}_2 \partial \boldsymbol{x}_2'} & \boldsymbol{A}'\boldsymbol{Q}^{-1} & \cdots & \vdots \\ 0 & \boldsymbol{Q}^{-1}\boldsymbol{A} & \frac{\partial^2 f}{\partial \boldsymbol{x}_3 \partial \boldsymbol{x}_3'} & \cdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \frac{\partial^2 f}{\partial \boldsymbol{x}_T \partial \boldsymbol{x}_T'} \end{pmatrix}$$

Use Newton-Raphson to find $\boldsymbol{\mu}_{\boldsymbol{x}} = argmax_{\boldsymbol{x}}\big(f(\boldsymbol{x})\big)$ and $\boldsymbol{\Sigma}_{\boldsymbol{x}} = -\left[\frac{\partial^2 f}{\partial x \partial x'}|_{X=\mu_X}\right]^{-\mathbf{1}}$

When using Newton-Raphson, $H\backslash\nabla$ in MATLAB will make use of block tri-diagonal structure automatically.

This can be much slower than smoother in some case, and my implementation is not robust (sometimes). See two examples in newtonTest.m.

To sample efficiently, use Cholesky decomposition of $\boldsymbol{\Sigma}_{\boldsymbol{x}}^{-1} = \boldsymbol{R}\boldsymbol{R}'$: sample $\boldsymbol{Z} \sim N(\boldsymbol{R}'\boldsymbol{\mu}_x, \boldsymbol{I})$, then $\boldsymbol{x} = (\boldsymbol{R}')^{-1}\boldsymbol{Z} \sim N(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$.

(2) Update $\boldsymbol{x}_0$:

$$P(\boldsymbol{x}_0|\boldsymbol{x}_1, \boldsymbol{Q}_0 \dots) \propto N(\boldsymbol{x}_1|\boldsymbol{x}_0, \boldsymbol{Q}_0)N(\boldsymbol{x}_0|\boldsymbol{\mu}_{x_{00}}, \boldsymbol{\Sigma}_{x_{00}})$$

Because of independence, we can update element by element. To write things easily, just write in matrix form. By conjugacy, $\boldsymbol{x}_0|\boldsymbol{x}_1, \boldsymbol{Q}_0 \dots \sim N(\boldsymbol{\mu}_{x_0}, \boldsymbol{\Sigma}_{x_0})$

$$\boldsymbol{\Sigma}_{x_0} = \left[\boldsymbol{\Sigma}_{x_{00}}^{-1} + \boldsymbol{Q}_0^{-1}\right]^{-1}$$
$$\boldsymbol{\mu}_{x_0} = \boldsymbol{\Sigma}_{x_0}\left(\boldsymbol{\Sigma}_{x_{00}}^{-1}\boldsymbol{\mu}_{x_{00}} + \boldsymbol{Q}_0^{-1}\boldsymbol{x}_1\right)$$

(3) Update $\{d_i\}_{i=1}^N$ and $\{\boldsymbol{c}_i\}_{i=1}^N$:

$$P\left(d_i, \boldsymbol{c}_i \middle| \boldsymbol{y}_i, \left\{\boldsymbol{x}_t^{(z_i)}\right\}_{t=1}^T, \dots\right) \propto \left[\prod_{t=1}^T P\left(y_{it}\middle|\boldsymbol{x}_t^{(z_i)}, d_i, \boldsymbol{c}_i\right)\right] P((d_i, \boldsymbol{c}_i')')$$

Notice $d_i + \boldsymbol{c}_i'\boldsymbol{x}_t^{(z_i)} = \left(1, \boldsymbol{x}_t^{(z_i)'}\right)(d_i, \boldsymbol{c}_i')'$. Then the problem is reduced to Bayesian Poisson regression. Denote $(d_i, \boldsymbol{c}_i')' = \boldsymbol{\zeta}_i \in R^{p+1}$ and $\left(1, \boldsymbol{x}_t^{(z_i)'}\right) = \widetilde{\boldsymbol{x}}_t^{(z_i)'}$. To get an efficient update, use the Laplace approximation: $P\left(\boldsymbol{\zeta}_i \middle| \boldsymbol{y}_i, \left\{\boldsymbol{x}_t^{(z_i)}\right\}_{t=1}^T, \dots\right) = \exp f(\boldsymbol{\zeta}_i) \approx N(\boldsymbol{\zeta}_i|\boldsymbol{\mu}_{\zeta_i}, \boldsymbol{\Sigma}_{\zeta_i})$

$$\frac{\partial f}{\partial \boldsymbol{\zeta}_i} = \frac{\partial l}{\partial \boldsymbol{\zeta}_i} - \boldsymbol{\Sigma}_{dc0}^{-1}(\boldsymbol{\zeta}_i - \boldsymbol{\mu}_{dc0}) = \left[\sum_{t=1}^T \widetilde{\boldsymbol{x}}_t^{(z_i)}(y_{it} - \lambda_{it})\right] - \boldsymbol{\Sigma}_{dC_0}^{-1}(\boldsymbol{\zeta}_i - \boldsymbol{\mu}_{dC_0})$$

$$\frac{\partial^2 f}{\partial \boldsymbol{\zeta}_i \partial \boldsymbol{\zeta}_i'} = \frac{\partial^2 l}{\partial \boldsymbol{\zeta}_i \partial \boldsymbol{\zeta}_i'} - \boldsymbol{\Sigma}_{dc0}^{-1} = -\left[\sum_{t=1}^{T} \lambda_{it} \widetilde{\boldsymbol{x}}_t^{(z_i)} \widetilde{\boldsymbol{x}}_t^{(z_i)\prime}\right] - \boldsymbol{\Sigma}_{dc0}^{-1}$$

Where l is Poisson log-likelihood.

Use Newton-Raphson to find $\boldsymbol{\mu}_{\zeta_j} = argmax_{\zeta_j}\left(f(\boldsymbol{\zeta}_j)\right)$ and $\boldsymbol{\Sigma}_{\zeta_j} = -\left[\frac{\partial^2 f}{\partial \zeta_j \partial \zeta_j'}|_{\zeta_j = \boldsymbol{\mu}_{\zeta_j}}\right]^{-1}$

(4) Update $\boldsymbol{A}$ and $\boldsymbol{b}$

a. Assume $\boldsymbol{Q}$ be diagonal:

$$P((b_k, \boldsymbol{a}_k')'|\{x_{kt}\}_{t=1}^T, q_k, \dots) \propto \left[\prod_{t=2}^{T} N(x_{kt}|\boldsymbol{a}_k'\boldsymbol{x}_{t-1} + b_k, q_k)\right] P((b_k, \boldsymbol{a}_k')')$$

Again, rewrite $\boldsymbol{a}_k'\boldsymbol{x}_{t-1} + b_k = (1, \boldsymbol{x}_{t-1}')(b_k, \boldsymbol{a}_k')'$. Then the problem is reduced to Bayesian linear regression. Denote $(b_k, \boldsymbol{a}_k')' = \boldsymbol{\gamma}_k \in R^{Jp+1}$ and $\widetilde{\boldsymbol{X}} = ((1, \boldsymbol{x}_1')', \dots, (1, \boldsymbol{x}_{T-1}')')'$. Then by conjugacy,

$$P(\boldsymbol{\gamma}_k|\{x_{kt}\}_{t=1}^T, q_k, \dots) = N(\boldsymbol{\gamma}_k|\boldsymbol{\mu}_{\gamma_k}, \boldsymbol{\Sigma}_{\gamma_k})$$

$$\boldsymbol{\Sigma}_{\gamma_k} = \left[\boldsymbol{\Sigma}_{ba0}^{-1} + \frac{\widetilde{\boldsymbol{X}}'\widetilde{\boldsymbol{X}}}{q_k}\right]^{-1}$$

$$\boldsymbol{\mu}_{\gamma_k} = \boldsymbol{\Sigma}_{\gamma_k}\left(\boldsymbol{\Sigma}_{ba0}^{-1}\boldsymbol{\mu}_{ba0} + \widetilde{\boldsymbol{X}}'(x_{k2}, \dots, x_{kT})'/q_k\right)$$

b. Assume $\boldsymbol{Q}$ be block-diagonal:

$$P\left((\boldsymbol{b}_j', \widetilde{\boldsymbol{A}}_j')' \middle| \left\{\boldsymbol{x}_t^{(j)}\right\}_{t=1}^T, \boldsymbol{Q}^{(j)}, \dots\right) \propto \left[\prod_{t=2}^{T} N\left(\boldsymbol{x}_t^{(j)}|\boldsymbol{A}_j\boldsymbol{x}_{t-1} + \boldsymbol{b}_j, \boldsymbol{Q}^{(j)}\right)\right] P\left((\boldsymbol{b}_j', \widetilde{\boldsymbol{A}}_j')'\right)$$

Again, notice $\boldsymbol{A}_j\boldsymbol{x}_{t-1} + \boldsymbol{b}_j = \left((1, \boldsymbol{x}_{t-1}{}')\otimes\boldsymbol{I}_p\right)\left(\boldsymbol{b}_j', \widetilde{\boldsymbol{A}}_j'\right)'$, then the problem is reduced to Bayesian linear regression. Denote $(1, \boldsymbol{x}_{t-1}{}')\otimes\boldsymbol{I}_p = \widetilde{\boldsymbol{X}}_{t-1} \in R^{p\times p(1+pJ)}$, $\widetilde{\boldsymbol{X}} = \left(\widetilde{\boldsymbol{X}}_1', \dots, \widetilde{\boldsymbol{X}}_{T-1}'\right)'$ and $\left(\boldsymbol{b}_j', \widetilde{\boldsymbol{A}}_j'\right)' = \boldsymbol{\gamma}_j \in R^{p(1+pJ)}$. Then

$$\prod_{t=2}^{T} N\left(\boldsymbol{x}_t^{(j)}|\boldsymbol{A}_j\boldsymbol{x}_{t-1} + \boldsymbol{b}_j, \boldsymbol{Q}^{(j)}\right) = \prod_{t=2}^{T} N\left(\boldsymbol{x}_t^{(j)}|\widetilde{\boldsymbol{X}}_{t-1}\boldsymbol{\gamma}_j, \boldsymbol{Q}^{(j)}\right)$$

$$= N\left(\left(\boldsymbol{x}_2^{(j)\prime}, \dots, \boldsymbol{x}_T^{(j)\prime}\right)' |\widetilde{\boldsymbol{X}}\boldsymbol{\gamma}_j, \boldsymbol{I}_{T-1}\otimes\boldsymbol{Q}^{(j)}\right)$$

By conjugacy:

$$P\left(\boldsymbol{\gamma}_j \middle| \left\{\boldsymbol{x}_t^{(j)}\right\}_{t=1}^T, \boldsymbol{Q}^{(j)}, \dots\right) = N\left(\boldsymbol{\gamma}_j|\boldsymbol{\mu}_{\gamma_j}, \boldsymbol{\Sigma}_{\gamma_j}\right)$$

$$\boldsymbol{\Sigma}_{\gamma_j} = \left[\boldsymbol{\Sigma}_{bA_0}^{-1} + \widetilde{\boldsymbol{X}}'\boldsymbol{I}_{T-1}\otimes\left(\boldsymbol{Q}^{(j)}\right)^{-1}\widetilde{\boldsymbol{X}}\right]^{-1}$$

$$\boldsymbol{\mu}_{\gamma_j} = \boldsymbol{\Sigma}_{\gamma_j}\left(\boldsymbol{\Sigma}_{bA_0}^{-1}\boldsymbol{\mu}_{bA_0} + \widetilde{\boldsymbol{X}}'\boldsymbol{I}_{T-1}\otimes\left(\boldsymbol{Q}^{(j)}\right)^{-1}\left(\boldsymbol{x}_2^{(j)\prime}, \dots, \boldsymbol{x}_T^{(j)\prime}\right)\right)$$

c. No constraints on $\boldsymbol{Q}$:

$$P\left((\boldsymbol{b}', \widetilde{\boldsymbol{A}}')' \middle| \{\boldsymbol{x}_t\}_{t=1}^T, \boldsymbol{Q}, \dots\right) \propto \left[\prod_{t=2}^{T} N(\boldsymbol{x}_t|\boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{b}, \boldsymbol{Q})\right] P\left((\boldsymbol{b}', \widetilde{\boldsymbol{A}}')'\right)$$

Let $\left(\boldsymbol{b}', \widetilde{\boldsymbol{A}}'\right)' = \boldsymbol{\gamma} \in R^{Jp(1+Jp)}$, $(1, \boldsymbol{x}_{t-1}{}')\otimes\boldsymbol{I}_{Jp} = \widetilde{\boldsymbol{G}}_{t-1} \in R^{Jp\times Jp(1+pJ)}$ and $\widetilde{\boldsymbol{G}} = \left(\widetilde{\boldsymbol{G}}_1', \dots, \widetilde{\boldsymbol{G}}_{T-1}'\right)'$. By conjugacy, $\boldsymbol{\gamma}|\{\boldsymbol{x}_t\}_{t=1}^T, \boldsymbol{Q}, \dots \sim N(\boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma)$.

$$\boldsymbol{\Sigma}_{\boldsymbol{\gamma}} = \left[\boldsymbol{\Sigma}_{BA_0}^{-1} + \widetilde{\boldsymbol{G}}'\boldsymbol{I}_{T-1}\otimes(\boldsymbol{Q})^{-1}\widetilde{\boldsymbol{G}}\right]^{-1}$$

$$\boldsymbol{\mu}_{\boldsymbol{\gamma}} = \boldsymbol{\Sigma}_{\boldsymbol{\gamma}}\left(\boldsymbol{\Sigma}_{BA_0}^{-1}\boldsymbol{\mu}_{BA_0} + \widetilde{\boldsymbol{G}}'\boldsymbol{I}_{T-1}\otimes(\boldsymbol{Q})^{-1}(\boldsymbol{x}_2', \dots, \boldsymbol{x}_T')\right)$$

(5) Update $\boldsymbol{Q}$:

a. Assume $\boldsymbol{Q}$ be diagonal:

Let $\boldsymbol{a}_k'\boldsymbol{x}_{t-1} + b_k = \mu_{x_{kt}}$

$$P(q_k|\{x_{kt}\}_{t=1}^T, b_k, \boldsymbol{a}_k', \dots) \propto \left[\prod_{t=2}^{T} N(x_{kt}|\mu_{x_{kt}}, q_k)\right] IG\left(q_k \middle| \frac{\nu_0}{2}, \frac{\nu_0\sigma_0^2}{2}\right)$$

By conjugacy,

$$P(q_k|\{x_{kt}\}_{t=1}^T, b_k, \boldsymbol{a}_k', \dots) = IG\left(q_k| \frac{\nu_0 + T - 1}{2}, \frac{\nu_0\sigma_0^2 + \sum_{t=2}^T \left(x_{kt} - \mu_{x_{kt}}\right)^2}{2}\right)$$

b. Assume $\boldsymbol{Q}$ be block-diagonal:

Let $\boldsymbol{A}_j\boldsymbol{x}_{t-1} + \boldsymbol{b}_j = \boldsymbol{\mu}_{\boldsymbol{x}_t^{(j)}}$

$$P\left(\boldsymbol{Q}^{(j)} \middle| \left\{\boldsymbol{x}_t^{(j)}\right\}_{t=1}^T, \boldsymbol{A}_j, \boldsymbol{b}_j, \dots\right) \propto \left[\prod_{t=2}^{T} N\left(\boldsymbol{x}_t^{(j)}|\boldsymbol{\mu}_{\boldsymbol{x}_t^{(j)}}, \boldsymbol{Q}^{(j)}\right)\right] W^{-1}(\boldsymbol{Q}^{(j)}|\Psi_0, \nu_0)$$

By conjugacy,

$$P\left(\boldsymbol{Q}^{(j)} \middle| \left\{\boldsymbol{x}_t^{(j)}\right\}_{t=1}^T, \boldsymbol{A}_j, \boldsymbol{b}_j, \dots\right)$$
$$= W^{-1}\left(\boldsymbol{Q}^{(j)}|\Psi_0 + \sum_{t=2}^{T}\left(\boldsymbol{x}_t^{(j)} - \boldsymbol{\mu}_{\boldsymbol{x}_t^{(j)}}\right)\left(\boldsymbol{x}_t^{(j)} - \boldsymbol{\mu}_{\boldsymbol{x}_t^{(j)}}\right)', T - 1 + \nu_0\right)$$

c. No constraints on $\boldsymbol{Q}$:

Let $\boldsymbol{A}\boldsymbol{x}_{t-1} + \boldsymbol{b} = \boldsymbol{\mu}_{x_t}$

$$P(\boldsymbol{Q}|\{\boldsymbol{x}_t\}_{t=1}^T, \boldsymbol{A}, \boldsymbol{b}, \dots) \propto \left[\prod_{t=2}^{T} N(\boldsymbol{x}_t|\boldsymbol{\mu}_{x_t}, \boldsymbol{Q})\right] W^{-1}(\boldsymbol{Q}|\Psi_{\boldsymbol{Q}_0}, \nu_{\boldsymbol{Q}_0})$$

By conjugacy,

$$\boldsymbol{Q}|\{\boldsymbol{x}_t\}_{t=1}^T, \boldsymbol{A}, \boldsymbol{b}, \dots \sim W^{-1}\left(\Psi_0 + \sum_{t=2}^{T}(\boldsymbol{x}_t - \boldsymbol{\mu}_{x_t})(\boldsymbol{x}_t - \boldsymbol{\mu}_{x_t})', T - 1 + \nu_0\right)$$