Appendix: Switching Autoregressive Low-rank Tensor Models

Anonymous Author(s)
Affiliation
Address
`email`

A SALT Optimization via Tensor Regression

Let $\mathbf{y}_t \in \mathbb{R}^{N_1}$ be the t-th outputs and $\mathbf{X}_t \in \mathbb{R}^{N_2 \times N_3}$ be the t-th inputs. The regression weights are a tensor $\mathcal{A} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$, which we model via a Tucker decomposition,

$$\mathcal{A} = \sum_{i=1}^{D_1} \sum_{j=1}^{D_2} \sum_{k=1}^{D_3} g_{ijk}\, \mathbf{u}_{:i} \circ \mathbf{v}_{:j} \circ \mathbf{w}_{:k}, \tag{1}$$

where $\mathbf{u}_i$, $\mathbf{v}_j$, and $\mathbf{w}_k$ are columns of the factor matrices $\mathbf{U} \in \mathbb{R}^{N_1 \times D_1}$, $\mathbf{V} \in \mathbb{R}^{N_2 \times D_2}$, and $\mathbf{W} \in \mathbb{R}^{N_3 \times D_3}$, respectively, and g_{ijk} are entries in the core tensor $\mathcal{G} \in \mathbb{R}^{D_1 \times D_2 \times D_3}$.

Consider the linear model, $\mathbf{y}_t \sim \mathcal{N}(\mathcal{A} \times_{2,3} \mathbf{X}_t, \mathbf{Q})$ where $\mathcal{A} \times_{2,3} \mathbf{X}_t$ is defined using the Tucker decomposition of $\mathcal{A}$ as,

$$\begin{aligned} \mathcal{A} \times_{2,3} \mathbf{X}_t &= \mathcal{A}_{(1)} \mathrm{vec}(\mathbf{X}_t) && (2)\\ &= \mathbf{U}\mathcal{G}_{(1)}(\mathbf{V}^\top \otimes \mathbf{W}^\top)\mathrm{vec}(\mathbf{X}_t) && (3)\\ &= \mathbf{U}\mathcal{G}_{(1)}\mathrm{vec}(\mathbf{V}^\top \mathbf{X}_t \mathbf{W}) && (4) \end{aligned}$$

where $\mathcal{A}_{(1)} \in \mathbb{R}^{N_1 \times N_2 N_3}$ and $\mathcal{G}_{(1)} \in \mathbb{R}^{D_1 \times D_2 D_3}$ are mode-1 matricizations of the corresponding tensors. *Note that these equations assume that matricization and vectorization are performed in row-major order, as in Python but opposite to what is typically used in Wikipedia articles.*

Equation (4) can be written in multiple ways, and these equivalent forms will be useful for deriving the updates below. We have,

$$\begin{aligned} \mathcal{A} \times_{2,3} \mathbf{X}_t &= \mathbf{U}\mathcal{G}_{(1)}(\mathbf{I}_{D_2} \otimes \mathbf{W}^\top \mathbf{X}_t^\top)\mathrm{vec}(\mathbf{V}^\top) && (5)\\ &= \mathbf{U}\mathcal{G}_{(1)}(\mathbf{V}^\top \mathbf{X}_t \otimes \mathbf{I}_{D_3})\mathrm{vec}(\mathbf{W}) && (6)\\ &= \left[\mathbf{U} \otimes \mathrm{vec}(\mathbf{V}^\top \mathbf{X}_t \mathbf{W})\right] \mathrm{vec}(\mathcal{G}). && (7) \end{aligned}$$

We minimize the negative log likelihood by coordinate descent.

Optimizing the output factors Let

$$\widetilde{\mathbf{x}}_t = \mathcal{G}_{(1)}\mathrm{vec}(\mathbf{V}^\top \mathbf{X}_t \mathbf{W}) \tag{8}$$

for fixed $\mathbf{V}$, $\mathbf{W}$, and $\mathcal{G}$. The NLL as a function of $\mathbf{U}$ is,

$$\mathcal{L}(\mathbf{U}) = \frac{1}{2} \sum_t (\mathbf{y}_t - \mathbf{U}\widetilde{\mathbf{x}}_t)^\top \mathbf{Q}^{-1} (\mathbf{y}_t - \mathbf{U}\widetilde{\mathbf{x}}_t). \tag{9}$$

This is a standard least squares problem with solution

$$\mathbf{U}^\star = \left(\sum_t \mathbf{y}_t \widetilde{\mathbf{x}}_t^\top\right)\left(\sum_t \widetilde{\mathbf{x}}_t \widetilde{\mathbf{x}}_t^\top\right)^{-1}. \tag{10}$$

Optimizing the core tensors Let $\widetilde{\mathbf{X}}_t = \mathbf{U} \otimes \mathrm{vec}(\mathbf{V}^\top \mathbf{X}_t \mathbf{W}) \in \mathbb{R}^{N_1 \times D_1 D_2 D_3}$ denote the coefficient on $\mathrm{vec}(\mathcal{G})$ in eq. (7). The NLL as a function of $\mathbf{g} = \mathrm{vec}(\mathcal{G})$ is,

$$\mathcal{L}(\mathbf{g}) = \frac{1}{2} \sum_t (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{g})^\top \mathbf{Q}^{-1} (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{g}). \tag{11}$$

The minimizer of this quadratic form is,

$$\mathbf{g}^\star = \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \widetilde{\mathbf{X}}_t \right)^{-1} \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \mathbf{y}_t \right) \tag{12}$$

Optimizing the input factors Let

$$\widetilde{\mathbf{X}}_t = \mathbf{U} \mathcal{G}_{(1)} (\mathbf{I}_{D_2} \otimes \mathbf{W}^\top \mathbf{X}_t^\top) \tag{13}$$

for fixed $\mathbf{U}$, $\mathbf{W}$, and $\mathcal{G}$. The NLL as a function of $\mathbf{v} = \mathrm{vec}(\mathbf{V}^\top)$ is,

$$\mathcal{L}(\mathbf{v}) = \frac{1}{2} \sum_t (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{v})^\top \mathbf{Q}^{-1} (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{v}). \tag{14}$$

The minimizer of this quadratic form is,

$$\mathbf{v}^\star = \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \widetilde{\mathbf{X}}_t \right)^{-1} \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \mathbf{y}_t \right) \tag{15}$$

Optimizing the lag factors Let

$$\widetilde{\mathbf{X}}_t = \mathbf{U} \mathcal{G}_{(1)} (\mathbf{V}^\top \mathbf{X}_t \otimes \mathbf{I}_{D_3}) \tag{16}$$

for fixed $\mathbf{U}$, $\mathbf{V}$, and $\mathcal{G}$. The NLL as a function of $\mathbf{w} = \mathrm{vec}(\mathbf{W})$ is,

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_t (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{w})^\top \mathbf{Q}^{-1} (\mathbf{y}_t - \widetilde{\mathbf{X}}_t \mathbf{w}). \tag{17}$$

The minimizer of this quadratic form is,

$$\mathbf{w}^\star = \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \widetilde{\mathbf{X}}_t \right)^{-1} \left(\sum_t \widetilde{\mathbf{X}}_t^\top \mathbf{Q}^{-1} \mathbf{y}_t \right) \tag{18}$$

Multiple discrete states If we have discrete states $z_t \in \{1, \ldots, H\}$ and each state has its own parameters $(\mathcal{G}^{(h)}, \mathbf{U}^{(h)}, \mathbf{V}^{(h)}, \mathbf{W}^{(h)}, \mathbf{Q}^{(h)})$, then letting $\omega_t^{(h)} = \mathbb{E}[z_t = h]$ denote the weights from the E-step, the summations in coordinate updates are weighted by $\omega_t^{(h)}$. For example, the coordinate update for the core tensors becomes,

$$\mathbf{g}^{(h)\star} = \left(\sum_t \omega_t^{(h)} \widetilde{\mathbf{X}}_t^{(h)\top} \mathbf{Q}^{(h)-1} \widetilde{\mathbf{X}}_t^{(h)} \right)^{-1} \left(\sum_t \omega_t^{(h)} \widetilde{\mathbf{X}}_t^{(h)\top} \mathbf{Q}^{(h)-1} \mathbf{y}_t \right) \tag{19}$$

B SALT approximates a Linear Dynamical System

A stationary linear dynamical system (LDS) is defined as follows:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + \mathbf{b} + \boldsymbol{\epsilon}_t \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{d} + \boldsymbol{\delta}_t \end{aligned}$$

where $\mathbf{y}_t \in \mathbb{R}^{N_1}$ is the t-th observation, $\mathbf{x}_t \in \mathbb{R}^{D_1}$ is the t-th hidden state, $\boldsymbol{\epsilon}_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{Q})$, $\boldsymbol{\delta}_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{R})$, and $\boldsymbol{\theta} = (\mathbf{A}, \mathbf{b}, \mathbf{Q}, \mathbf{C}, \mathbf{d}, \mathbf{R})$ are the parameters of the LDS.

Following the notation of Murphy [1], the one-step-ahead posterior predictive distribution for the observations of the LDS defined above can be expressed as:

$$p(\mathbf{y}_t | \mathbf{y}_{1:t-1}) = \mathcal{N}(\mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d}, \mathbf{C}\boldsymbol{\Sigma}_{t|t-1}\mathbf{C}^T + \mathbf{R})$$

where

$$\begin{aligned}
\boldsymbol{\mu}_{t|t-1} &= \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{b} \\
\boldsymbol{\mu}_{t} &= \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t \mathbf{r}_t \\
\boldsymbol{\Sigma}_{t|t-1} &= \mathbf{A}\boldsymbol{\Sigma}_{t-1}\mathbf{A}^T + \mathbf{Q} \\
\boldsymbol{\Sigma}_{t} &= (\mathbf{I}_{D_1} - \mathbf{K}_t\mathbf{C})\boldsymbol{\Sigma}_{t|t-1} \\
p(\mathbf{x}_1) &= \mathcal{N}(\boldsymbol{\mu}_{1|0}, \boldsymbol{\Sigma}_{1|0}) \\
\mathbf{K}_t &= (\boldsymbol{\Sigma}_{t|t-1}^{-1} + \mathbf{C}^T\mathbf{R}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{R}^{-1} \\
\mathbf{r}_t &= \mathbf{y}_t - \mathbf{C}\boldsymbol{\mu}_{t|t-1} - \mathbf{d}.
\end{aligned}$$

We can then expand the mean $\mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d}$ as follows:

$$\mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d} = \mathbf{C}\sum_{l=1}^{t-1} \boldsymbol{\Gamma}_l \mathbf{A}\mathbf{K}_{t-l}\mathbf{y}_{t-l} + \mathbf{C}\sum_{l=1}^{t-1} \boldsymbol{\Gamma}_l(\mathbf{b} - \mathbf{A}\mathbf{K}_{t-l}\mathbf{d}) + \mathbf{d}$$

where

$$\boldsymbol{\Gamma}_l = \prod_{i=1}^{l-1} \mathbf{A}(\mathbf{I}_{D_1} - \mathbf{K}_{t-i}\mathbf{C}).$$

If $(\mathbf{A}, \mathbf{Q})$ is stabilizable and $(\mathbf{C}, \mathbf{A})$ is detectable, $\lim_{t\to\infty}\boldsymbol{\Sigma}_{t|t-1} = \boldsymbol{\Sigma}$, where $\boldsymbol{\Sigma}$ is the unique solution of the discrete algebraic Riccati equation $\boldsymbol{\Sigma} = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T - \mathbf{A}\boldsymbol{\Sigma}\mathbf{C}^T(\mathbf{C}\boldsymbol{\Sigma}\mathbf{C}^T + \mathbf{R})^{-1}\mathbf{C}\boldsymbol{\Sigma}\mathbf{A}^T + \mathbf{Q}$ (Kumar and Varaiya [2], Katayama et al. [3]). Consequently, the Kalman gain matrix converges to $\mathbf{K} = (\boldsymbol{\Sigma}^{-1} + \mathbf{C}^T\mathbf{R}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{R}^{-1}$ for large t.

Assuming that $(\mathbf{A}, \mathbf{Q})$ is stabilizable, $(\mathbf{C}, \mathbf{A})$ is detectable, and t is large enough, we can approximate the mean as

$$\begin{aligned}
\mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d} &\approx \mathbf{C}\sum_{l=1}^{t-1} \boldsymbol{\Gamma}_{l,\mathrm{stable}}\mathbf{A}\mathbf{K}\mathbf{y}_{t-l} + \mathbf{C}\sum_{l=1}^{t-1} \boldsymbol{\Gamma}_{l,\mathrm{stable}}(\mathbf{b} - \mathbf{A}\mathbf{K}\mathbf{d}) + \mathbf{d} \\
&\approx \mathbf{C}\sum_{l=1}^{N_3} \boldsymbol{\Gamma}_{l,\mathrm{stable}}\mathbf{A}\mathbf{K}\mathbf{y}_{t-l} + \mathbf{C}\sum_{l=1}^{N_3} \boldsymbol{\Gamma}_{l,\mathrm{stable}}(\mathbf{b} - \mathbf{A}\mathbf{K}\mathbf{d}) + \mathbf{d}
\end{aligned}$$

where

$$\boldsymbol{\Gamma}_{l,\mathrm{stable}} = \prod_{i=1}^{l-1} \mathbf{A}(\mathbf{I}_{D_1} - \mathbf{K}\mathbf{C}) = (\mathbf{A}(\mathbf{I}_{D_1} - \mathbf{K}\mathbf{C}))^{l-1}$$

and N_3 is sufficiently large.

Now we further assume that $\mathbf{A}(\mathbf{I} - \mathbf{K}\mathbf{C})$ has linearly independent eigenvectors and let $\mathbf{E}\boldsymbol{\Lambda}\mathbf{E}^{-1}$ be the eigendecomposition of $\mathbf{A}(\mathbf{I} - \mathbf{K}\mathbf{C})$ in real modal form. If $\mathbf{A}(\mathbf{I} - \mathbf{K}\mathbf{C})$ has n real eigenvalues and m pairs of complex eigenvalues (i.e., $n + 2m = D_1$), we can express $\mathbf{E}$, $\boldsymbol{\Lambda}$, and $\mathbf{E}^{-1}$ as:

$$\mathbf{E} = \begin{bmatrix} \mathbf{a}_1 & \dots & \mathbf{a}_n & \mathbf{b}_1 & \mathbf{c}_1 & \dots & \mathbf{b}_m & \mathbf{c}_m \end{bmatrix}$$

$$\boldsymbol{\Lambda} = \begin{bmatrix}
\lambda_1 & & & & & & & \\
& \ddots & & & & & & \\
& & \lambda_n & & & & & \\
& & & \sigma_1 & \omega_1 & & & \\
& & & -\omega_1 & \sigma_1 & & & \\
& & & & & \ddots & & \\
& & & & & & \sigma_m & \omega_m \\
& & & & & & -\omega_m & \sigma_m
\end{bmatrix}$$

$$
\mathbf{E}^{-1} = \begin{bmatrix} \mathbf{d}_1^T \\ \vdots \\ \mathbf{d}_n^T \\ \mathbf{e}_1^T \\ \boldsymbol{f}_1^T \\ \vdots \\ \mathbf{e}_m^T \\ \boldsymbol{f}_m^T \end{bmatrix}
$$

where $\mathbf{a}_1 \ldots \mathbf{a}_n$ are the right eigenvectors corresponding to n real eigenvalues $\lambda_1 \ldots \lambda_n$, and $\mathbf{b}_i$ and $\mathbf{c}_i$ are the real and imaginary parts of the eigenvector corresponding to the complex eigenvalue $\sigma_i + j\omega_i$.

Since

$$
\boldsymbol{\Gamma}_{l,\text{stable}} = (\mathbf{A}(\mathbf{I}_{D_1} - \mathbf{KC}))^{l-1} = \mathbf{E}\boldsymbol{\Lambda}^{l-1}\mathbf{E}^{-1}
$$

The l^{th} power of $\boldsymbol{\Lambda}$, $\boldsymbol{\Lambda}^l$, where $l \geq 0$, can be expressed as:

$$
\boldsymbol{\Lambda}^l = \begin{bmatrix}
\lambda_1^l & & & & & & & \\
& \ddots & & & & & & \\
& & \lambda_n^l & & & & & \\
& & & \sigma_{1,l} & \omega_{1,l} & & & \\
& & & -\omega_{1,l} & \sigma_{1,l} & & & \\
& & & & & \ddots & & \\
& & & & & & \sigma_{m,l} & \omega_{m,l} \\
& & & & & & -\omega_{m,l} & \sigma_{m,l}
\end{bmatrix}
$$

where $\sigma_{i,l} = \sigma_{i,l-1}^2 - \omega_{i,l-1}^2$, $\omega_{i,l} = 2\sigma_{i,l-1}\omega_{i,l-1}$ for $l \geq 2$, $\sigma_{i,1} = \sigma_i$, $\omega_{i,1} = \omega_i$, $\sigma_{i,0} = 1$, and $\omega_{i,0} = 0$.

Tucker Tensor Regression Let $\mathcal{H} \in \mathbb{R}^{D \times D \times L}$ be a three-way tensor, whose l^{th} frontal slice $\mathbf{H}_{::l} = \boldsymbol{\Lambda}^{l-1}$. Let $\mathcal{G} \in \mathbb{R}^{D \times D \times D}$ be a three-way tensor, whose entry $g_{ijk} = \mathbb{1}_{i=j=k}$ for $1 \leq k \leq n$, and $g_{ijk} = (-1)^{\mathbb{1}_{i+1=j=k+1}} \mathbb{1}_{(i=j=k)\vee(i-1=j-1=k)\vee(i=j+1=k+1)\vee(i+1=j=k+1)}$ for $k \in \{n+1, n+3, \ldots, n+2m-1\}$. Let $\mathbf{W} \in \mathbb{R}^{L \times D}$ be a matrix, whose entry $w_{lk} = \lambda_k^{l-1}$ for $1 \leq k \leq n$, $w_{lk} = \sigma_{k,l-1}$ for $k \in \{n+1, n+3, \ldots, n+2m-1\}$, and $w_{lk} = -\omega_{k,l-1}$ for $k \in \{n+2, n+4, \ldots, n+2m\}$. We can then decompose $\mathcal{H}$ into $\mathcal{G} \in \mathbb{R}^{D \times D \times D}$ and $\mathbf{W} \in \mathbb{R}^{L \times D}$ such that $\mathcal{H} = \mathcal{G} \times_3 \mathbf{W}$ (Figure 1).

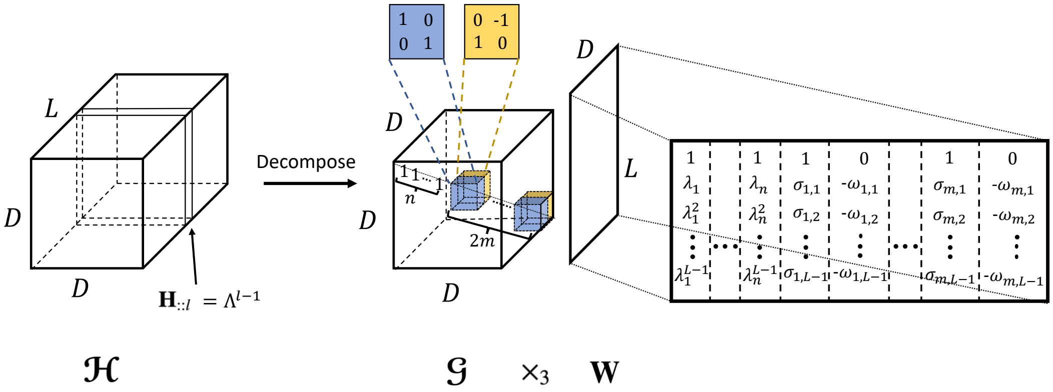

Figure 1: **Decomposition of $\mathcal{H}$ into $\mathcal{G}$ and W such that $\mathcal{H} = \mathcal{G} \times_3 \mathbf{W}$:** Given an LDS whose $\mathbf{A}(\mathbf{I} - \mathbf{KC})$ has n real eigenvalues and m pairs of complex eigenvalues, this decomposition illustrates how Tucker-SALT can approximate the LDS well with rank $n + 2m$.

With $\mathbf{V} = (\mathbf{E}^{-1}\mathbf{AK})^T$, $\mathbf{U} = \mathbf{CE}$, $\mathbf{m} = \mathbf{C}\sum_{l=1}^{N_3} \mathbf{\Gamma}_{l,\text{stable}}(\mathbf{b} - \mathbf{AKd}) + \mathbf{d}$, and $\mathbf{X}_t = \mathbf{y}_{t-1:t-N_3}$, we can rearrange the mean to:

$$
\begin{aligned}
\mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d} \approx & \mathbf{C}\sum_{l=1}^{N_3} \mathbf{E}\mathbf{\Lambda}^{l-1}\mathbf{E}^{-1}\mathbf{AK}\mathbf{y}_{t-l} + \mathbf{C}\sum_{l=1}^{N_3} \mathbf{\Gamma}_{l,\text{stable}}(\mathbf{b} - \mathbf{AKd}) + \mathbf{d} \\
= & \mathbf{U}\sum_{l=1}^{N_3} \mathbf{H}_{::l}\mathbf{V}^T\mathbf{y}_{t-l} + \mathbf{m} \\
= & \mathbf{U}\sum_{l=1}^{N_3} (\mathcal{G}\bar{\times}_3\mathbf{w}_l)\mathbf{V}^T\mathbf{y}_{t-l} + \mathbf{m} \\
= & \mathbf{U}\sum_{l=1}^{N_3} ((\mathcal{G} \times_2 \mathbf{V})\bar{\times}_3\mathbf{w}_l)\mathbf{y}_{t-l} + \mathbf{m} \\
= & \mathbf{U}\sum_{l=1}^{N_3}\sum_{j=1}^{D_2}\sum_{k=1}^{D_3} \mathbf{g}_{:jk} \circ \mathbf{v}_{:j}(w_{lk}\mathbf{y}_{t-l}) + \mathbf{m} \\
= & \mathbf{U}\sum_{j=1}^{D_2}\sum_{k=1}^{D_3} \mathbf{g}_{:jk}(\mathbf{v}_{:j}^{\top}\mathbf{X}_t\mathbf{w}_{:k}) + \mathbf{m} \\
= & \sum_{i=1}^{D_1}\sum_{j=1}^{D_2}\sum_{k=1}^{D_3} \mathbf{u}_{:i}g_{ijk}(\mathbf{v}_{:j}^{\top}\mathbf{X}_t\mathbf{w}_{:k}) + \mathbf{m} \\
= & \left[\sum_{i=1}^{D_1}\sum_{j=1}^{D_2}\sum_{k=1}^{D_3} g_{ijk}\mathbf{u}_{:i} \circ \mathbf{v}_{:j} \circ \mathbf{w}_{:k}\right] \times_{2,3} \mathbf{X}_t + \mathbf{m}
\end{aligned}
$$

CP Tensor Regression By rearranging $\mathbf{E}, \mathbf{\Lambda}^l$, and $\mathbf{E}^{-1}$ into $\mathbf{J}$, $\mathbf{P}_l$, and $\mathbf{S}$ respectively as follows:

$$
\mathbf{J} = \begin{bmatrix} \mathbf{a}_1 & \dots & \mathbf{a}_n & \mathbf{b}_1 + \mathbf{c}_1 & \mathbf{b}_1 & \mathbf{c}_1 & \dots & \mathbf{b}_m + \mathbf{c}_m & \mathbf{b}_m & \mathbf{c}_m \end{bmatrix}
$$

$$\mathbf{P}_l = \begin{bmatrix} \lambda_1^l & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & \lambda_n^l & & & & & & & \\ & & & \sigma_{1,l} & & & & & & \\ & & & & \alpha_{1,l} & & & & & \\ & & & & & \beta_{1,l} & & & & \\ & & & & & & \ddots & & & \\ & & & & & & & \sigma_{m,l} & & \\ & & & & & & & & \alpha_{m,l} & \\ & & & & & & & & & \beta_{m,l} \end{bmatrix}$$

$$\mathbf{S} = \begin{bmatrix} \mathbf{d}_1^T \\ \vdots \\ \mathbf{d}_n^T \\ \mathbf{e}_1^T + \boldsymbol{f}_1^T \\ \boldsymbol{f}_1^T \\ \boldsymbol{e}_1^T \\ \vdots \\ \mathbf{e}_m^T + \boldsymbol{f}_m^T \\ \boldsymbol{f}_m^T \\ \boldsymbol{e}_m^T \end{bmatrix}$$

where $\mathbf{J} \in \mathbb{R}^{D\times(n+3m)}$, $\mathbf{P}_l \in \mathbb{R}^{(n+3m)\times(n+3m)}$, $\mathbf{S} \in \mathbb{R}^{(n+3m)\times D}$, $\alpha_{i,l} = \omega_{i,l} - \sigma_{i,l}$, and $\beta_{i,l} = -\omega_{i,l} - \sigma_{i,l}$, we can diagonalize $(\mathbf{A}(\mathbf{I}_{D_1} - \mathbf{KC}))^l$ as $\mathbf{JP}_l\mathbf{S}$.

Let $\mathbf{V} = (\mathbf{SAK})^T$, $\mathbf{U} = \mathbf{CJ}$, $\mathbf{m} = \mathbf{C}\sum_{l=1}^{N_3} \boldsymbol{\Gamma}_{l,\text{stable}}(\mathbf{b} - \mathbf{AKd}) + \mathbf{d}$, and $\mathbf{X}_t = \mathbf{y}_{t-1:t-N_3}$. Let $\mathbf{W} \in \mathbb{R}^{N_3\times(n+3m)}$ be a matrix, whose element in the l^{th} row and k^{th} column is $p_{l-1,kk}$ (i.e., the element in the k^{th} row and k^{th} column of $\mathbf{P}_{l-1}$), and $\mathcal{G} \in \mathbb{R}^{D_1\times D_1\times D_1}$ be a superdiagonal 3-way tensor, where $g_{ijk} = \mathbb{1}_{i=j=k}$. We can then rearrange the mean to:

$$\begin{aligned} \mathbf{C}\boldsymbol{\mu}_{t|t-1} + \mathbf{d} \approx& \mathbf{C}\sum_{l=1}^{N_3} \mathbf{E}\boldsymbol{\Lambda}^{l-1}\mathbf{E}^{-1}\mathbf{AKy}_{t-l} + \mathbf{C}\sum_{l=1}^{N_3} \boldsymbol{\Gamma}_{l,\text{stable}}(\mathbf{b} - \mathbf{AKd}) + \mathbf{d} \\ =& \mathbf{C}\sum_{l=1}^{N_3} \mathbf{JP}_{l-1}\mathbf{SAKy}_{t-l} + \mathbf{m} \\ =& \mathbf{U}\sum_{l=1}^{N_3} \mathbf{P}_{l-1}\mathbf{V}^\top\mathbf{y}_{t-l} + \mathbf{m} \\ =& \sum_{l=1}^{N_3}\sum_{i}^{n+3m}\sum_{j}^{n+3m}\sum_{k}^{n+3m} g_{ijk}\, \mathbf{u}_{:i} \circ \mathbf{v}_{:j}(p_{l-1,kk}\mathbf{y}_{t-l}) + \mathbf{m} \\ =& \sum_{i}^{n+3m}\sum_{j}^{n+3m}\sum_{k}^{n+3m} g_{ijk}\, \mathbf{u}_{:i} \circ \mathbf{v}_{:j}(\mathbf{X}_t\mathbf{w}_{:k}) + \mathbf{m} \\ =& \left[\sum_{i=1}^{n+3m}\sum_{j=1}^{n+3m}\sum_{k=1}^{n+3m} g_{ijk}\, \mathbf{u}_{:i} \circ \mathbf{v}_{:j} \circ \mathbf{w}_{:k}\right] \times_{2,3} \mathbf{X}_t + \mathbf{m} \end{aligned}$$

Correspondence between SALT and Switching Linear Dynamical Systems Above, we demonstrated how SALT can approximate a linear dynamical system. For a switching linear dynamical system, this correspondence will hold within each discrete state. It is important to note, however, that this correspondence will be less exact near the boundary between discrete states.

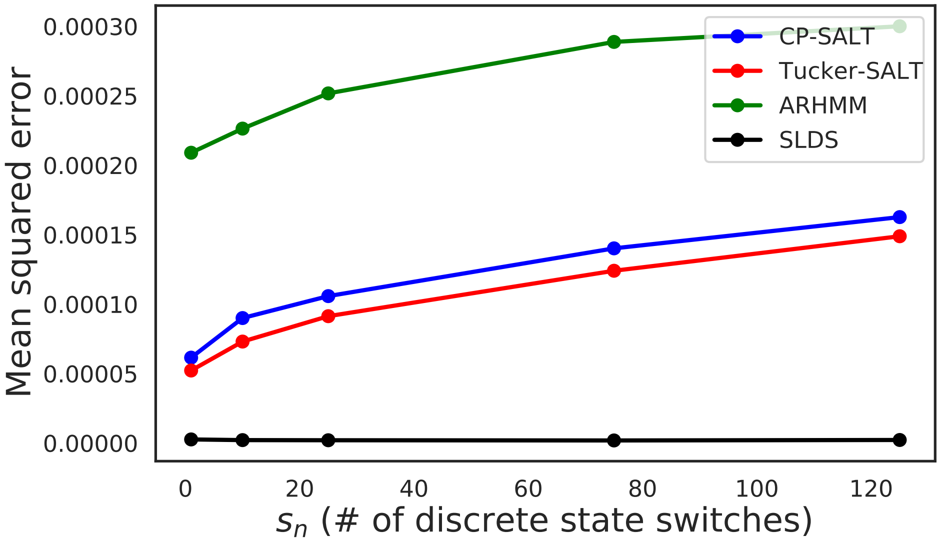

Figure 2: **The quality of SALT approximation of SLDSs decreases as the number of discrete state switches increases:** The data comes from an SLDS with $H = 2$, $N = 20$, and $D = 7$. 15,000 timesteps were generated, with varying numbers of evenly spaced out discrete state switches (x-axis). The mean squared error of reconstructing the autoregressive tensors increased with the number of discrete state switches.

This is because in SALT, the autoregressive dynamics only depend on the current discrete state. However, in an SLDS, when we marginalize out the continuous latent states, the autoregressive dynamics in the observation space will depend on previous discrete states as well.

C Modeling Mouse Behavior: Videos

Here we describe how the mouse behavioral videos were generated. We first determined the CP-SALT hyperparameters as those which led to the highest log-likelihood on the validation dataset. Then, using that CP-SALT model, we computed the most likely discrete states on the train and test data. Given a discrete state h, we extracted slices of the data whose most likely discrete state was h. We padded the data by 30 frames (i.e. 1 second) both at the beginning and the end of each slice for the movie. A red dot appears on each mouse for the duration of discrete state h. We generated such videos for all 50 discrete states (as long as there existed at least one slice for each discrete state) on the train and test data. For a given discrete state, the mice in each video behaved very similarly (e.g., the mice in the video for state 18 “pause" when the red dots appear, and those in the video for state 32 “walk" forward), suggesting that CP-SALT is capable of segmenting the data into useful behavioral syllables.

D Additional Synthetic Data Experiments

D.1 The effect of the number of switches on the recovery of the parameters of the autoregressive dynamic tensors

We asked how the number of discrete state switches affected SALT’s ability to recover the autoregressive tensors. We trained CP-SALT, Tucker-SALT, the ARHMM, all with $L = 5$ lags, and the SLDS on data sampled from an SLDS with varying number of discrete state switches. The ground-truth SLDS model had $H = 2$ discrete states, $N = 20$ observations and $D = 7$ dimensional continuous latent states. The matrix $\mathbf{A}^{(h)}(\mathbf{I} - \mathbf{K}^{(h)}\mathbf{C}^{(h)})$ of each discrete state of the ground-truth SLDS had 1 real eigenvalue and 3 pairs of complex eigenvalues. We sampled 5 batches of $T = 15,000$ timesteps of data from the ground-truth SLDS, with $s_n \in \{1, 10, 25, 75, 125\}$ number of discrete state switches that were evenly spaced out across the data. We then computed the mean squared error (MSE) between the SLDS tensors and the tensors reconstructed by SALT, the ARHMM, and the SLDS. (Figure 2). More precisely, we combined the 3rd order autoregressive tensors from each discrete into a 4th order tensor, and calculated the MSE based on these 4th order tensors. As expected, the MSE increased with the number of switches in the data, indicating that the quality of SALT approximation of SLDSs decreases as the number of discrete state switches increases.

E Code

As part of Supplementary materials, we include the source code for SALT (in Python) and an example Jupyter Notebook for sampling data from SALT and fitting SALT to the data with an EM algorithm. SALT code requires the ssm-jax package.

References

[1] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[2] Panqanamala Ramana Kumar and Pravin Varaiya. *Stochastic systems: Estimation, identification, and adaptive control*. SIAM, 2015.

[3] Tohru Katayama et al. *Subspace methods for system identification*, volume 1. Springer, 2005.