Appendix 1

Algorithm Standard Kalman Filter Smoother for estimating the moments required in the E-step of an EM algorithm for a linear dynamical system
0. Define $\boldsymbol{x}_t^\tau = \mathrm{E}(\boldsymbol{x}_t \vert \boldsymbol{Y}_1^\tau)$,$\mathbf{V}_t^\tau = \mathrm{Var}(\boldsymbol{x}_t \vert \boldsymbol{Y}_1^\tau)$, $\hat{\boldsymbol{x}}_t \equiv \boldsymbol{x}_t^T$ and $\hat{P}_t \equiv V_t^T + \boldsymbol{x}_t^T \boldsymbol{x}_t^{T^\intercal}$
1. Forward Recursions:
$\boldsymbol{x}_t^{t-1} = A\boldsymbol{x}_{t-1}^{t-1}$
$\mathbf{V}_t^{t-1} = A\mathbf{V}_{t-1}^{t-1} + \mathbf{Q}$
$K_t = \mathbf{V}_t^{t-1} C^\intercal (C V_t^{t-1} C^\intercal + R)^{-1}$
$\boldsymbol{x}_t^t = \boldsymbol{x}_t^{t-1} + K_t(\boldsymbol{y}_t - C\boldsymbol{x}_t^{t-1})$
$V_t^t = V_t^{t-1} - K_t C V_t^{t-1}$
$\boldsymbol{x}_1^0 = \pi_0$, $V_1^0 = \mathbf{V}_0$
2. Backward Recursions:
$J_{t-1} = V_{t-1}^{t-1} A^\intercal (V_t^{t-1})^{-1}$
$\boldsymbol{x}_{t-1}^T = \boldsymbol{x}_{t-1}^{t-1} + J_{t-1}(\mathbf{x_t^T} - \mathbf{A}\boldsymbol{x}_{\mathbf{t-1}}^{\mathbf{t-1}})$
$V_{t-1}^T = V_{t-1}^{t-1} + J_{t-1}(V_t^T - V_t^{t-1}) J_{t-1}^\intercal$
$\hat{P}_{t,t-1} \equiv V_{t,t-1}^T + \boldsymbol{x}_t^T \boldsymbol{x}_t^{T^\intercal}$
$V_{T,T-1}^T = (I - K_T C) A V_{T-1}^{T-1}$

Appendix 2

In general, FISTA optimize a target function

$$\min_{x \in \mathcal{X}} \quad \mathbf{F}(\mathbf{x}; \lambda) = \mathbf{g}(\mathbf{x}) + \lambda \|\mathbf{x}\|_{\mathbf{1}} \tag{1}$$

where $\mathbf{g} : R^n \to R$ is a continuously differentiable convex function and $\lambda > 0$ is the regularization parameter. A FISTA algorithm with constant step is detailed below

Algorithm FISTA($\mathbf{g}, \lambda$).

1. Input an initial guess $\mathbf{x_0}$ and Lipschitz constant $\mathbf{L}$ for $\nabla\mathbf{g}$, set $\mathbf{y_1} = \mathbf{x_0}, t_1 = 1$
2. Choose $\tau \in (0, 1/\mathbf{L}]$; Set $k \leftarrow 0$.
3. **loop**
4. Evaluate $\nabla\mathbf{g}(\mathbf{y_k})$
5. Compute $\mathbf{x_1}$= $\mathbf{S}_{\tau\lambda}(\mathbf{y_k} - \tau\nabla\mathbf{g}(\mathbf{y_k}))$
6. Compute $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
7. $\mathbf{y_{k+1}} = \mathbf{x_k} + \Big(\frac{t_k-1}{t_{k+1}}\Big)\Big(\mathbf{x_k} - \mathbf{x_{k-1}}\Big)$
8. Set $k \leftarrow k+1$
9. **end loop**

In the above

$$\mathbf{S}_\lambda(\mathbf{y}) = (|\mathbf{y}| - \lambda)_+ \mathbf{sign}(\mathbf{y}) = \begin{cases} y - \lambda & \text{if } y > \lambda \\ y + \lambda & \text{if } y < -\lambda \\ 0 & \text{if } |y| \leq \lambda. \end{cases}$$

Appendix 3

Algorithm k-step predictions with PCA and MR. SID

1. Denote estimations with PCA and MR. SID as $A_{pca}, C_{pca}, A_{plds}$, and C_{plds} respectively.
2. PCA estimated latent states at $t = 1000$: $x_{1000,pca}$ = column 1000 of $\boldsymbol{X}_{d\times T}$ from Section 3.3
3. MR. SID estimated latent states at $t = 1000$: $x_{1000,pls}$ is from the E step in Section 3.4
4. **for i = 1 to k**
5. $x_{1000+k,pca} \;=\; A_{pca}\ x_{999+k,pca}$
6. $y_{1000+k,pca} \;=\; C_{pca}\ x_{1000+k,pca}$
7. $x_{1000+k,plds} \;=\; A_{plds}\ x_{999+k,plds}$
8. $y_{1000+k,plds} \;=\; C_{plds}\ x_{1000+k,plds}$
9. **end**

Appendix 4

Algorithm Simulation Data Generation

1. Denote the dimensions as p, d and T respectively
2. Generate a $p \times d$ matrix C_0 from a standard Gaussian distribution
3. Sort each column of C_0 in ascending order to get matrix C
4. Generate a $d \times d$ matrix A_0 from a standard Gaussian distribution
5. Add a multiple of the identity matrix to A_0
6. Replace entries in A_0 with small absolute values with 0
7. Scale A_0 to make sure its eigen values are between -1 and 1; use A_0 as the A matrix
8. Let R be a diagonal matrix with positive diagonal entries and Q be the identity matrix
9. Generate simulation data with A, C, Q and R
10. **end**