Learning the Linear Dynamical System with ASOS (“**A**pproximated **S**econd-**O**rder **S**tatistics”)

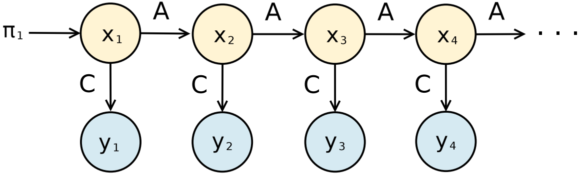
James Martens

University of Toronto

June 24, 2010

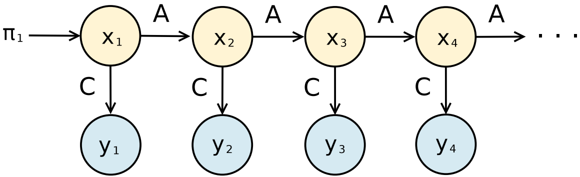
Computer Science
UNIVERSITY OF TORONTO

The Linear Dynamical System model



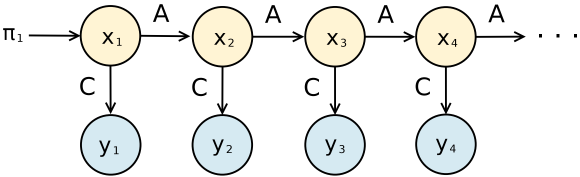
- model of vector-valued time-series $\{y_t \in \mathbb{R}^{N_y}\}_{t=1}^{T}$

The Linear Dynamical System model



- model of vector-valued time-series $\{y_t \in \mathbb{R}^{N_y}\}_{t=1}^{T}$
- widely-applied due to predictable behavior, easy inference, etc

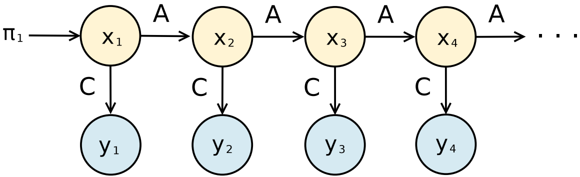
The Linear Dynamical System model



- model of vector-valued time-series $\{y_t \in \mathbb{R}^{N_y}\}_{t=1}^{T}$
- widely-applied due to predictable behavior, easy inference, etc
- vector-valued hidden states $(\{x_t \in \mathbb{R}^{N_x}\}_{t=1}^{T})$ evolve via linear dynamics,

$$x_{t+1} = Ax_t + \epsilon_t \qquad A \in \mathbb{R}^{N_x \times N_x} \qquad \epsilon_t \sim N(0, Q)$$

The Linear Dynamical System model



- model of vector-valued time-series $\{y_t \in \mathbb{R}^{N_y}\}_{t=1}^{T}$
- widely-applied due to predictable behavior, easy inference, etc
- vector-valued hidden states $(\{x_t \in \mathbb{R}^{N_x}\}_{t=1}^{T})$ evolve via linear dynamics,

$$x_{t+1} = Ax_t + \epsilon_t \qquad A \in \mathbb{R}^{N_x \times N_x} \qquad \epsilon_t \sim N(0, Q)$$

- linearly generated observations:

$$y_t = Cx_t + \delta_t \qquad C \in \mathbb{R}^{N_y \times N_x} \qquad \delta_t \sim N(0, R)$$

Learning the LDS

Expectation Maximization (EM)

- finds local optimum of log-likelihood
- pretty slow - convergence requires lots of iterations and E-step is expensive

Learning the LDS

Expectation Maximization (EM)

- finds local optimum of log-likelihood
- pretty slow - convergence requires lots of iterations and E-step is expensive

Subspace identification

- hidden states estimated directly from the data, and the parameters from these
- asymptotically unbiased / consistent
- non-iterative algorithm, but solution not optimal in any objective
- good way to initialize EM or other iterative optimizers

Our contribution

- accelerate the EM algorithm by reducing its per-iteration cost to be constant time w.r.t. T (length of the time-series)

Our contribution

- accelerate the EM algorithm by reducing its per-iteration cost to be constant time w.r.t. T (length of the time-series)
- key idea: approximate the inference done in the E-step

Our contribution

- accelerate the EM algorithm by reducing its per-iteration cost to be constant time w.r.t. T (length of the time-series)
- key idea: approximate the inference done in the E-step
- E-step approximation is unbiased and asymptotically consistent
- also convergences exponentially with L, where L is a meta-parameter that trades off approximation quality with speed
  - (notation change: L is "k_{lim}" from the paper)

Learning via E.M. the Algorithm

E.M. Objective Function

At each iteration we maximize the following objective where θ_n is the current parameter estimate:

$$\mathcal{Q}_n(\theta) = E_{\theta_n}[\log p(x,y)|y] = \int_x p(x|y,\theta_n) \log p(x,y|\theta)$$

Learning via E.M. the Algorithm

E.M. Objective Function

At each iteration we maximize the following objective where θ_n is the current parameter estimate:

$$\mathcal{Q}_n(\theta) = E_{\theta_n}[\log p(x, y)|y] = \int_x p(x|y, \theta_n) \log p(x, y|\theta)$$

E-Step

- E-Step computes expectation of $\log p(x, y|\theta)$ under $p(x|y, \theta_n)$
- uses the classical Kalman filtering/smoothing algorithm

Learning via E.M. the Algorithm (cont.)

M-Step

- maximize objective $\mathcal{Q}_n(\theta)$ w.r.t. to θ, producing a new estimate θ_{n+1}

$$\theta_{n+1} = \arg\max_{\theta} \mathcal{Q}_n(\theta)$$

- very easy - similar to linear-regression

Learning via E.M. the Algorithm (cont.)

M-Step

- maximize objective $\mathcal{Q}_n(\theta)$ w.r.t. to θ, producing a new estimate θ_{n+1}

$$\theta_{n+1} = \arg\max_{\theta} \mathcal{Q}_n(\theta)$$

- very easy - similar to linear-regression

Problem

- EM can get very slow for when we have lots of data
- mainly due to call to expensive Kalman filter/smoother in the E-step
  - $O(N_x^3 T)$ where T = length of the training time-series, N_x = hidden state dim.

- the Kalman filter/smoother estimates hidden-state means and covariances:

$$x_t^k \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq k} \,]$$
$$V_{t,s}^k \equiv \mathrm{Cov}_{\theta_n}[\, x_t, x_s \mid y_{\leq k} \,]$$

for each $t = \{1, ..., T\}$ and $s = t, t+1$.

- the Kalman filter/smoother estimates hidden-state means and covariances:

$$x_t^k \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq k} \,]$$
$$V_{t,s}^k \equiv \mathrm{Cov}_{\theta_n}[\, x_t, x_s \mid y_{\leq k} \,]$$

for each $t = \{1, ..., T\}$ and $s = t, t+1$.

- these are summed over time to obtain the statistics required for M-step, e.g.:

$$\mathrm{E}_{\theta_n}[x_{t+1}x_t' \mid y_{\leq k} \,] = (x^T, x^T)_1 + \sum_{t=1}^{T-1} V_{t+1,t}^T$$

where $\boxed{(a, b)_k \equiv \sum_{t=1}^{T-k} a_{t+k} b_t'}$

- the Kalman filter/smoother estimates hidden-state means and covariances:

$$x_t^k \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq k} \,]$$
$$V_{t,s}^k \equiv \mathrm{Cov}_{\theta_n}[\, x_t, x_s \mid y_{\leq k} \,]$$

for each $t = \{1, ..., T\}$ and $s = t, t+1$.

- these are summed over time to obtain the statistics required for M-step, e.g.:

$$\mathrm{E}_{\theta_n}[x_{t+1}x_t' \mid y_{\leq k} \,] = (x^T, x^T)_1 + \sum_{t=1}^{T-1} V_{t+1,t}^T$$

where $\boxed{(a, b)_k \equiv \sum_{t=1}^{T-k} a_{t+k} b_t'}$

- but we only care about the M-statistics, not the individual inferences for each time-step $\longrightarrow$ so let's estimate these directly!

Steady-state

- first we need a basic tool from linear systems/control theory: "steady-state"

Steady-state

- first we need a basic tool from linear systems/control theory: "steady-state"
- the covariance terms, and the "filtering and smoothing matrices" (denoted K_t and J_t) do not depend on the data y - only the current parameters

Steady-state

- first we need a basic tool from linear systems/control theory: "steady-state"
- the covariance terms, and the "filtering and smoothing matrices" (denoted K_t and J_t) do not depend on the data y - only the current parameters
- and they rapidly converge to "steady-state" values:

$$V_{t,t}^T, \; V_{t,t-1}^T, \; J_t, \; K_t \longrightarrow \Lambda_0, \; \Lambda_1, \; J, \; K \quad \text{as} \quad \min(t, T-t) \rightarrow \infty$$

- we can approximate the Kalman filter/smoother equations using the steady-state matrices

- we can approximate the Kalman filter/smoother equations using the steady-state matrices
- this gives the highly simplified recurrences

$$x_t^* = Hx_{t-1}^* + Ky_t \qquad x_t^T = Jx_{t+1}^T + Px_t^*$$

where $\boxed{x_t^* \equiv x_t^t \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq t} \,]}$, $H \equiv A - KCA$ and $P \equiv I - JA$

- we can approximate the Kalman filter/smoother equations using the steady-state matrices

- this gives the highly simplified recurrences

$$x_t^* = Hx_{t-1}^* + Ky_t \qquad x_t^T = Jx_{t+1}^T + Px_t^*$$

where $\boxed{x_t^* \equiv x_t^t \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq t} \,]}$, $H \equiv A - KCA$ and $P \equiv I - JA$

- these don't require any matrix multiplications or inversions

- we can approximate the Kalman filter/smoother equations using the steady-state matrices
- this gives the highly simplified recurrences

$$x_t^* = Hx_{t-1}^* + Ky_t \qquad x_t^T = Jx_{t+1}^T + Px_t^*$$

where $\boxed{x_t^* \equiv x_t^t \equiv \mathrm{E}_{\theta_n}[\, x_t \mid y_{\leq t} \,]}$, $H \equiv A - KCA$ and $P \equiv I - JA$

- these don't require any matrix multiplications or inversions
- we apply the approximate filter/smoother everywhere except first and last i time-steps
  - yields a run-time of $O(N_x^2 T + N_x^3 i)$.

ASOS: Approximated Second-Order Statistics

- steady-state makes covariance terms easy to estimate in time independent of T

ASOS: Approximated Second-Order Statistics

- steady-state makes covariance terms easy to estimate in time independent of T
- we want something similar for sum-of-products of means terms like $(x^T, x^T)_0 \equiv \sum_t x_t^T (x_t^T)'$

ASOS: Approximated Second-Order Statistics

- steady-state makes covariance terms easy to estimate in time independent of T
- we want something similar for sum-of-products of means terms like $(x^T, x^T)_0 \equiv \sum_t x_t^T (x_t^T)'$
- such sums we will call "2^{nd}-order statistics". The ones-needed for the M-step are the "M-statistics"

ASOS: Approximated Second-Order Statistics

- steady-state makes covariance terms easy to estimate in time independent of T
- we want something similar for sum-of-products of means terms like $(x^T, x^T)_0 \equiv \sum_t x_t^T (x_t^T)'$
- such sums we will call "2^{nd}-order statistics". The ones-needed for the M-step are the "M-statistics"
- idea #1: derive recursions and equations that relate the 2^{nd}-order statistics of different "time-lags"
  - "time-lag" refers to the value of k in $(a, b)_k \equiv \sum_{t=1}^{T-k} a_{t+k} b_t'$

ASOS: Approximated Second-Order Statistics

- steady-state makes covariance terms easy to estimate in time independent of T
- we want something similar for sum-of-products of means terms like $(x^T, x^T)_0 \equiv \sum_t x_t^T (x_t^T)'$
- such sums we will call "2^{nd}-order statistics". The ones-needed for the M-step are the "M-statistics"
- idea #1: derive recursions and equations that relate the 2^{nd}-order statistics of different "time-lags"
  - "time-lag" refers to the value of k in $(a, b)_k \equiv \sum_{t=1}^{T-k} a_{t+k} b_t'$
- idea #2: evaluate these efficiently using approximations

Deriving the 2^{nd}-order recursions/equations: An example

- suppose we wish to find the recursion for $(x^*, y)_k$

Deriving the 2nd-order recursions/equations: An example

- suppose we wish to find the recursion for $(x^*, y)_k$
- steady-state Kalman recursion for x^*_{t+k} is: $x^*_{t+k} = Hx^*_{t+k-1} + Ky_{t+k}$
- right-multiply both sides by y_t' and sum over t

$$\boxed{(x^*, y)_k} \equiv \sum_{t=1}^{T-k} x^*_{t+k} y_t' = \sum_{t=1}^{T-k} (Hx^*_{t+k-1} y_t' + Ky_{t+k} y_t')$$

Deriving the 2nd-order recursions/equations: An example

- suppose we wish to find the recursion for $(x^*, y)_k$
- steady-state Kalman recursion for x^*_{t+k} is: $x^*_{t+k} = Hx^*_{t+k-1} + Ky_{t+k}$
- right-multiply both sides by y'_t and sum over t
- factor out matrices H and K

$$\boxed{(x^*, y)_k} \equiv \sum_{t=1}^{T-k} x^*_{t+k} y'_t = \sum_{t=1}^{T-k} (Hx^*_{t+k-1} y'_t + Ky_{t+k} y'_t)$$

$$= H \sum_{t=1}^{T-k} x^*_{t+k-1} y'_t + K \sum_{t=1}^{T-k} y_{t+k} y'_t$$

Deriving the 2nd-order recursions/equations: An example

- suppose we wish to find the recursion for $(x^*, y)_k$
- steady-state Kalman recursion for x^*_{t+k} is: $x^*_{t+k} = Hx^*_{t+k-1} + Ky_{t+k}$
- right-multiply both sides by y'_t and sum over t
- factor out matrices H and K
- finally, re-write everything using our special notation for 2nd-order statistics: $(a, b)_k \equiv \sum_{t=1}^{T-k} a_{t+k} b'_t$

$$\boxed{(x^*, y)_k} \equiv \sum_{t=1}^{T-k} x^*_{t+k} y'_t = \sum_{t=1}^{T-k} (Hx^*_{t+k-1} y'_t + Ky_{t+k} y'_t)$$

$$= H \sum_{t=1}^{T-k} x^*_{t+k-1} y'_t + K \sum_{t=1}^{T-k} y_{t+k} y'_t$$

$$= H(\boxed{(x^*, y)_{k-1}} - x^*_T y'_{T-k+1}) + K \boxed{(y, y)_k}$$

The complete list (don't bother to memorize this)

The recursions:

$$
\begin{aligned}
(y, x^*)_k &= (y, x^*)_{k+1} H' + ((y, y)_k - y_{1+k} y_1') K' + y_{1+k} x_1^{*\prime} \\
(x^*, y)_k &= H((x^*, y)_{k-1} - x_T^* y_{T-k+1}') + K(y, y)_k \\
(x^*, x^*)_k &= (x^*, x^*)_{k+1} H' + ((x^*, y)_k - x_{1+k}^* y_1') K' + x_{1+k}^* x_1^{*\prime} \\
(x^*, x^*)_k &= H((x^*, x^*)_{k-1} - x_T^* x_{T-k+1}^{*}{}') + K(y, x^*)_k \\
(x^T, y)_k &= J(x^T, y)_{k+1} + P((x^*, y)_k - x_T^* y_{T-k}{}') + x_T^T y_{T-k}{}' \\
(x^T, x^*)_k &= J(x^T, x^*)_{k+1} + P((x^*, x^*)_k - x_T^* x_{T-k}^*{}') + x_T^T x_{T-k}^*{}' \\
(x^T, x^T)_k &= ((x^T, x^T)_{k-1} - x_k^T x_1^T) J' + (x^T, x^*)_k P' \\
(x^T, x^T)_k &= J(x^T, x^T)_{k+1} + P((x^*, x^T)_k - x_T^* x_{T-k}^T{}') + x_T^T x_{T-k}^T{}'
\end{aligned}
$$

The equations:

$$
(x^*, x^*)_k = H(x^*, x^*)_k H' + ((x^*, y)_k - x_{1+k}^* y_1') K' - H x_T^* x_{T-k}^*{}' H' + K(y, x^*)_{k+1} H' + x_{1+k}^* x_1^{*\prime}
$$

$$
(x^T, x^T)_k = J(x^T, x^T)_k J' + P((x^*, x^T)_k - x_T^* x_{T-k}^T{}') - J x_{k+1}^T x_1^{T\prime} J' + J(x^T, x^*)_{k+1} P' + x_T^T x_{T-k}^T{}'
$$

- noting that statistics of time-lag $T+1$ are 0 by definition we can start the 2^{nd}-order recursions at $t = T$
- but this doesn't get us anywhere - would be even more expensive than the usual Kalman recursions on the 1^{st}-order terms

- noting that statistics of time-lag $T+1$ are 0 by definition we can start the 2$^{\mathrm{nd}}$-order recursions at $t = T$
- but this doesn't get us anywhere - would be even more expensive than the usual Kalman recursions on the 1$^{\mathrm{st}}$-order terms
- instead, start the recursions at time-lag $\sim L$ with unbiased approximations ("ASOS approximations")

$$(y, x^*)_{L+1} \approx CA\left((x^*, x^*)_L - x^*_T x^{*\prime}_{T-L}\right), \quad (x^T, x^*)_L \approx (x^*, x^*)_L, \quad (x^T, y)_L \approx (x^*, y)_L$$

- noting that statistics of time-lag $T+1$ are 0 by definition we can start the 2^{nd}-order recursions at $t = T$
- but this doesn't get us anywhere - would be even more expensive than the usual Kalman recursions on the 1^{st}-order terms
- instead, start the recursions at time-lag $\sim L$ with unbiased approximations ("ASOS approximations")

$$(y, x^*)_{L+1} \approx CA\left((x^*, x^*)_L - x^*_T x^*_{T-L}{}'\right), \quad (x^T, x^*)_L \approx (x^*, x^*)_L, \quad (x^T, y)_L \approx (x^*, y)_L$$

- we also need x_t^T for $t \in \{1, 2, ..., L\} \cup \{T{-}L, T{-}L{+}1, ..., T\}$ but these can be approximated by a separate procedure (see paper)

Why might this be reasonable?

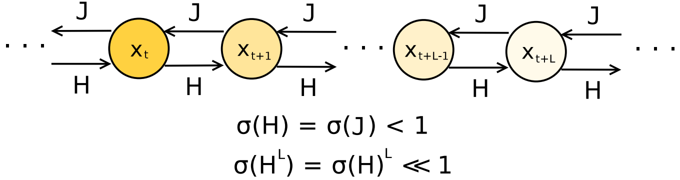
- 2^{nd}-order statistics with large time lag quantify relationships between variables that are far apart in time
  - weaker and less important than relationships between variables that are close in time

Why might this be reasonable?

- 2^{nd}-order statistics with large time lag quantify relationships between variables that are far apart in time
  - weaker and less important than relationships between variables that are close in time
- in steady-state Kalman recursions, information is propagated via multiplication by H and J:

$$x_t^* = Hx_{t-1}^* + Ky_t \qquad x_t^T = Jx_{t+1}^T + Px_t^*$$

- both of these have spectral radius (denoted $\sigma(\cdot)$) less than 1, and so they decay the signal exponentially



Procedure for estimating the M-statistics

- how do we compute an estimate of the M-statistics consistent with the 2nd-order recursions/equations and the approximations?

Procedure for estimating the M-statistics

- how do we compute an estimate of the M-statistics consistent with the 2nd-order recursions/equations and the approximations?
- essentially it is just a large linear system of dimension $O(N_x^2 L)$

Procedure for estimating the M-statistics

- how do we compute an estimate of the M-statistics consistent with the 2nd-order recursions/equations and the approximations?

- essentially it is just a large linear system of dimension $O(N_x^2 L)$

- but using a general solver would be far too expensive: $O(N_x^6 L^3)$

Procedure for estimating the M-statistics

- how do we compute an estimate of the M-statistics consistent with the 2nd-order recursions/equations and the approximations?
- essentially it is just a large linear system of dimension $O(N_x^2 L)$
- but using a general solver would be far too expensive: $O(N_x^6 L^3)$
- fortunately, using the special structure of this system, we have developed a (non-trivial) algorithm which is much more efficient
  - equations can be solved using an efficient iterative algorithm we developed for a generalization of the Sylvester equation
  - evaluating recursions is then straightforward

Procedure for estimating the M-statistics

- how do we compute an estimate of the M-statistics consistent with the 2nd-order recursions/equations and the approximations?
- essentially it is just a large linear system of dimension $O(N_x^2 L)$
- but using a general solver would be far too expensive: $O(N_x^6 L^3)$
- fortunately, using the special structure of this system, we have developed a (non-trivial) algorithm which is much more efficient
  - equations can be solved using an efficient iterative algorithm we developed for a generalization of the Sylvester equation
  - evaluating recursions is then straightforward
- the cost is then just $O(N_x^3 L)$ after $(y, y)_k \equiv \sum_t y_{t+k} y_t'$ has been pre-computed for $k = 0, ..., L$

First convergence result: our intuition confirmed

- **First result:** For a fixed θ the ℓ_2-error in the M-statistics is bounded by a quantity proportional to $L^2\lambda^{L-1}$, where $\lambda = \sigma(H) = \sigma(J) < 1$
    - ($\sigma(\cdot)$ denotes the spectral radius)

First convergence result: our intuition confirmed

- **First result:** For a fixed θ the ℓ_2-error in the M-statistics is bounded by a quantity proportional to $L^2\lambda^{L-1}$, where $\lambda = \sigma(H) = \sigma(J) < 1$
  - ($\sigma(\cdot)$ denotes the spectral radius)
- so as L grows, the estimation error for the M-statistics will decay exponentially

First convergence result: our intuition confirmed

- **First result:** For a fixed θ the ℓ_2-error in the M-statistics is bounded by a quantity proportional to $L^2\lambda^{L-1}$, where $\lambda = \sigma(H) = \sigma(J) < 1$
  - ($\sigma(\cdot)$ denotes the spectral radius)

- so as L grows, the estimation error for the M-statistics will decay exponentially

- *but*, λ might be close enough to 1 so that we need to make L too big

First convergence result: our intuition confirmed

- **First result:** For a fixed θ the ℓ_2-error in the M-statistics is bounded by a quantity proportional to $L^2\lambda^{L-1}$, where $\lambda = \sigma(H) = \sigma(J) < 1$
  - ($\sigma(\cdot)$ denotes the spectral radius)
- so as L grows, the estimation error for the M-statistics will decay exponentially
- *but*, λ might be close enough to 1 so that we need to make L too big
- fortunately we have a 2nd result which provides a very different type of guarantee

Second convergence result

- **Second result:** updates produced by ASOS procedure are asymptotically consistent with the usual EM updates in the limit as $T \to \infty$
  - assumes data is generated from the model

Second convergence result

- **Second result:** updates produced by ASOS procedure are asymptotically consistent with the usual EM updates in the limit as $T \to \infty$
  - assumes data is generated from the model
- first result didn't make strong use any property of the approx.
  - (could use 0 for each and result would still hold)

Second convergence result

- **Second result:** updates produced by ASOS procedure are asymptotically consistent with the usual EM updates in the limit as $T \to \infty$
  - assumes data is generated from the model
- first result didn't make strong use any property of the approx.
  - (could use 0 for each and result would still hold)
- this second one is justified in the opposite way
  - strong use of the approximation
  - follows from convergence of $\frac{1}{T}$-scaled expected ℓ_2 error of approx. towards zero
  - result holds for any value of L value

Experiments

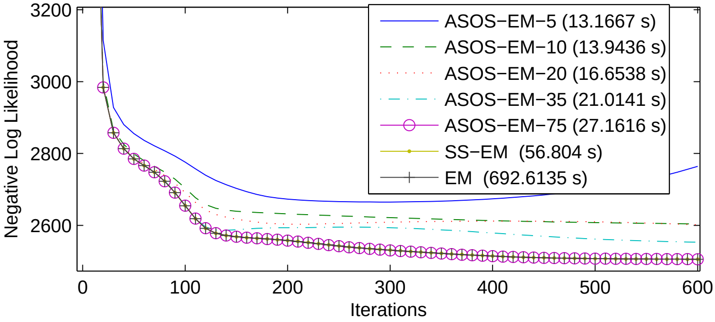
- we considered 3 real datasets of varying sizes and dimensionality
- each algorithm initialized from same random parameters
- latent dimension N_x determined by trial-and-error

Experimental parameters

Name	*length* (T)	N_y	N_x
evaporator	6305	3	15
motion capture	15300	10	40
warship sounds	750000	1	20

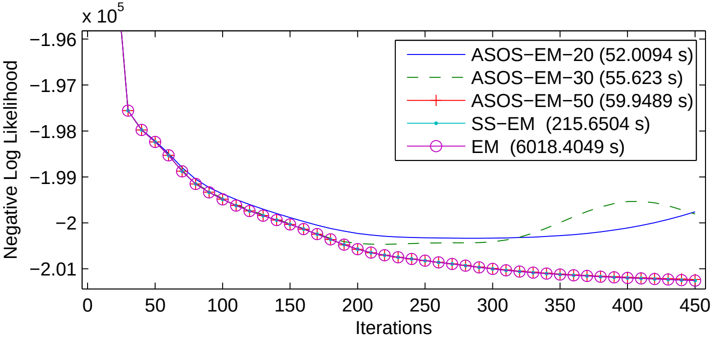
Experimental results (cont.)

Name	*length (T)*	N_y	N_x
evaporator	6305	3	15
motion capture	15300	10	40
warship sounds	750000	1	20



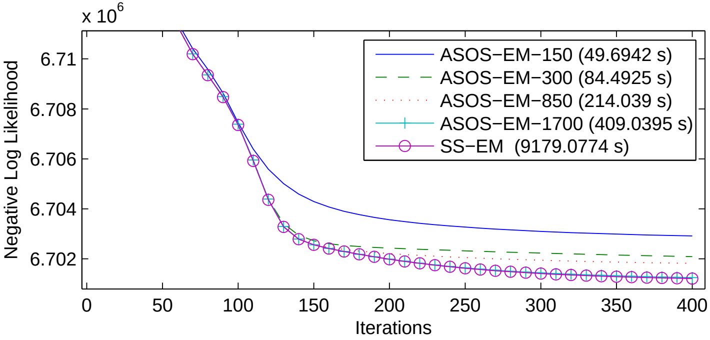
Experimental results (cont.)

Name	*length (T)*	N_y	N_x
evaporator	6305	3	15
motion capture	15300	10	40
warship sounds	750000	1	20



Experimental results (cont.)

Name	*length (T)*	N_y	N_x
evaporator	6305	3	15
motion capture	15300	10	40
warship sounds	750000	1	20



Conclusion

- we applied steady-state approximations to derive a set of "2nd-order recursions and equations"

Conclusion

- we applied steady-state approximations to derive a set of "2nd-order recursions and equations"
- approximated statistics of time-lag L

Conclusion

- we applied steady-state approximations to derive a set of "2nd-order recursions and equations"
- approximated statistics of time-lag L
- produced an efficient algorithm for solving the resulting system

Per-iteration run-times:

EM	SS-EM	ASOS-EM
$O(N_x^3 T)$	$O(N_x^2 T + N_x^3 i)$	$O(N_x^3 k_{lim})$

Conclusion

- we applied steady-state approximations to derive a set of "2nd-order recursions and equations"
- approximated statistics of time-lag L
- produced an efficient algorithm for solving the resulting system

Per-iteration run-times:

EM	SS-EM	ASOS-EM
$O(N_x^3 T)$	$O(N_x^2 T + N_x^3 i)$	$O(N_x^3 k_{lim})$

- gave 2 formal convergence results

Conclusion

- we applied steady-state approximations to derive a set of "2nd-order recursions and equations"
- approximated statistics of time-lag L
- produced an efficient algorithm for solving the resulting system

Per-iteration run-times:

EM	SS-EM	ASOS-EM
$O(N_x^3 T)$	$O(N_x^2 T + N_x^3 i)$	$O(N_x^3 k_{lim})$

- gave 2 formal convergence results
- demonstrated significant performance benefits for learning with long time-series

Thank you for your attention