Nonlinear Filtering
Unscented Kalman filtering with SVD

Tohru Katayama
Emeritus Professor of Kyoto University
Visiting Professor of Doshisha University
Chair Professor of Ritsumeikan University

Joint Paper

Michiaki Takeno and Tohru Katayama, On the selection of σ points and the improvement of performance in Uncesenced Kalman filter based on singular value decomposition (in preparation).

In the Unscented Transformation (UT) used for Unscented Kalman Filter (UKF), we need to compute square roots of covariance matrices to select sigma points, which approximate the covariance information of conditional probabilities. We show by using a 2^{nd}-order model that SVD-based matrix square roots better capture the covariance information than Cholesky decomposition. Simulation results for several discrete and continuous nonlinear systems are also included to show the applicability of the SVD-based UKF algorithm.

Nonlinear Filtering

- **Nonlinear filtering has a long history, i.e. in 1960s, there published many papers, including**

  Cox (IEEE 1964)
  Kushner (SIAM 1964)
  Bucy (IEEE 1965)
  Jazwinski (1970)

- **Extended Kalman filter (EKF) has been tested and successfully used for Apollo project at NASA during 1960s (Grewal & Andrews, CS Magazine 2010).**

Nonlinear Filtering Techniques

Local methods	**EKF**	**Extended Kalman Filter**
	SLKF	**Statistical Linearization Kalman Filter**
	UKF	**Unscended Kalman Filter**
	EnKF	**Ensemble Kalman Filter**
Global methods	**GSF**	**Gaussian Sum Filter**
	PMF	**Point Mass Filter**
	PF	**Particle Filter**

- **S. Lakshmivarahan and D. J. Stensrud, *CS Magazine*, June 2009**
- **M. Šimandl and J. Duník, *Automatica*, July 2009**

Outline

- **Nonlinear Filtering**
- **Statistical Linearization**
- **Unscented Transformation**
- **A Motivating Example**
- **Cholesky decomposition vs. SVD**
- **Several Simulation Results**
- **Conclusions**

Nonlinear Filtering

Consider a nonlinear stochastic system

$$x_{t+1} = f_t(x_t) + w_t$$
$$y_t = h_t(x_t) + v_t$$

$x_t \in \mathbb{R}^n$ **: the state vector,** $y_t \in \mathbb{R}^p$ **: the output vector**

$f_t : \mathbb{R}^n \to \mathbb{R}^n$, $h_t : \mathbb{R}^n \to \mathbb{R}^p$**: nonlinear transforms**

w_t, v_t**: Gaussian white noises**

$$E[w_t^{\mathrm{T}} \quad v_t^{\mathrm{T}}] \begin{bmatrix} w_s \\ v_s \end{bmatrix} = \begin{bmatrix} Q_t & 0 \\ 0 & R_t \end{bmatrix} \delta_{ts}$$

Nonlinear Filtering

The problem is to compute the conditional expectation, or the conditional mean estimate

$$\hat{x}_{t+m|t} = E\{x_{t+m} \mid Y^t\} = \int_{\mathbb{R}^n} x_{t+m} p(x_{t+m} \mid Y^t) dx_{t+m}$$

$$m = 0, 1$$

$$Y^t = \{y_0, y_1, \cdots, y_t\}$$

which minimizes the conditional Bayes risk

$$J = E\{\|x_{t+m} - \hat{x}_{t+m|t}\|^2 \mid Y^t\}$$

Conditional Probabilities

- **Observation update**

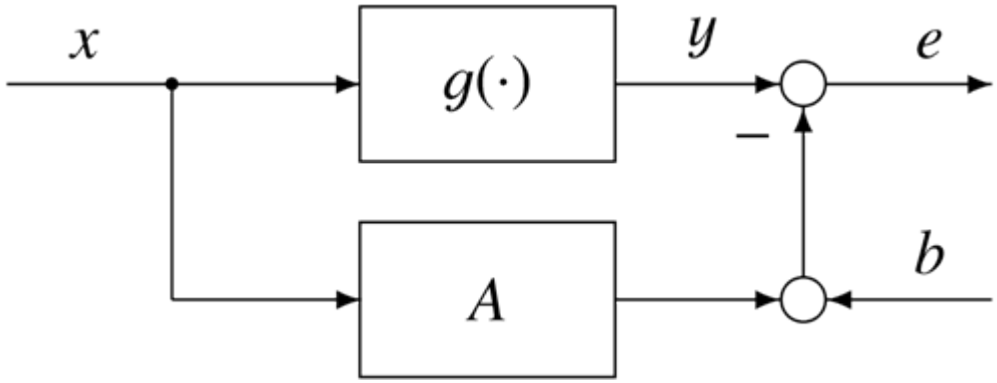
$$p(x_t \mid Y^t) = \frac{p(y_t \mid x_t)p(x_t \mid Y^{t-1})}{p(y_t \mid Y^{t-1})}$$

- **Time update**

$$p(x_{t+1} \mid Y^t) = \int_{\mathbb{R}^n} p(x_{t+1} \mid x_t)p(x_t \mid Y^t)dx_t$$

Except for the Linear Gaussian case, it is impossible to find the optimal solution to filtering problems, though there are some results for obtaining exact solutions (Beneš 1981, Daum 1986).

Statistical Linearization



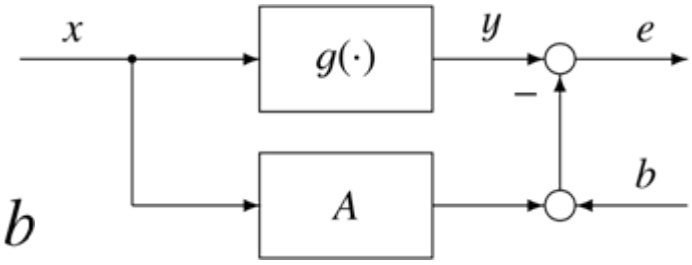
$g : \mathbb{R}^n \to \mathbb{R}^p$: **a nonlinear transformation**

$x \in \mathbb{R}^n, y \in \mathbb{R}^p$: **random variables**

The problem is to find the unbiased linear minimum variance estimate of $g(\cdot)$, i.e.

$$J = E\|e\|^2 = E\|y - Ax - b\|^2 \to \min_{A,b}, \; A \in \mathbb{R}^{p\times n}, \; b \in \mathbb{R}^p$$

Statistical Linearization



- **Unbiased estimate**

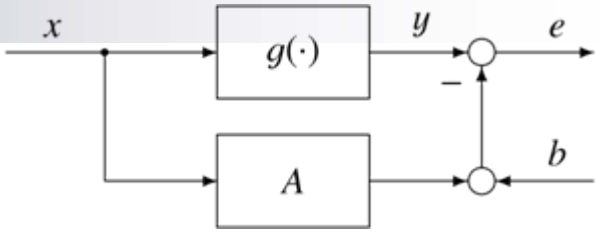
$$E(y - Ax - b) = 0 \;\rightarrow\; \mu_y = A\mu_x + b$$

$$\begin{aligned}
\min J &= \text{trace} E[(y - Ax - b)(y - Ax - b)^{\mathrm{T}}] \\
&= \text{trace} E[y - \mu_y - A(x - \mu_x)][y - \mu_y - A(x - \mu_x)]^{\mathrm{T}} \\
&= \text{trace}\left[\Sigma_{yy} - A\Sigma_{xy} - \Sigma_{yx}A^{\mathrm{T}} + A\Sigma_{xx}A^{\mathrm{T}}\right]
\end{aligned}$$

$$A = \Sigma_{yx}\Sigma_{xx}^{-1}, \;\; b = \mu_y - \Sigma_{yx}\Sigma_{xx}^{-1}\mu_x \;\rightarrow\; \hat{y} = \mu_y + \Sigma_{yx}\Sigma_{xx}^{-1}(x - \mu_x)$$

- **However, computation of μ_y, Σ_{yx} is almost impossible except for the case where $x \sim N(\mu, \Sigma)$ and the nonlinearity $g(\cdot)$ is a polynomial.**

LS Problem



Given the data (x_i, y_i), $i = 1, \cdots, N$, we have a LS problem:

$$J = \frac{1}{N}\sum_{i=1}^{N} e_i^{\mathrm{T}} e_i = \frac{1}{N}\sum_{i=1}^{N} (y_i - Ax_i - b)^{\mathrm{T}} (y_i - Ax_i - b) \;\rightarrow\; \min_{A,b}$$

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i, \quad \bar{y} = \frac{1}{N}\sum_{i=1}^{N} g(x_i), \quad S_{yx} = \frac{1}{N}\sum_{i=0}^{N} [g(x_i) - \bar{y}][x_i - \bar{x}]^{\mathrm{T}}$$

$$A = S_{yx} S_{xx}^{-1}, \quad b = \bar{y} - S_{yx} S_{xx}^{-1} \bar{x}$$

$$\rightarrow \quad \hat{y} = \bar{y} + S_{yx} S_{xx}^{-1} (x - \bar{x})$$

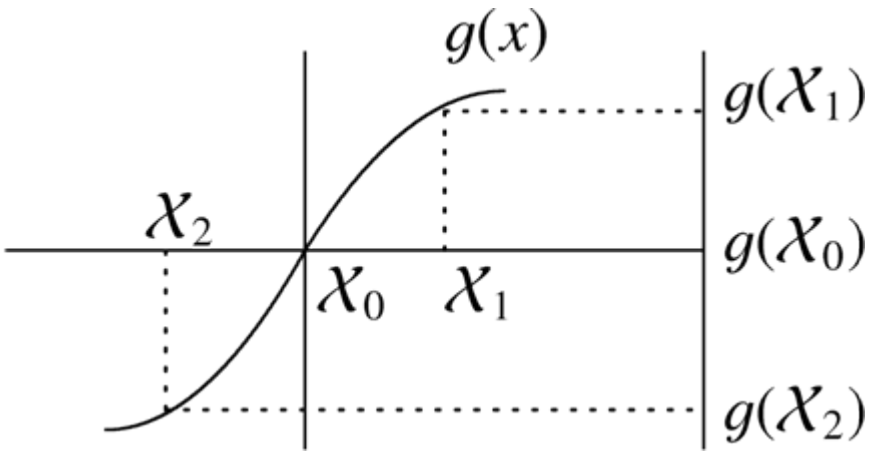
Unscented Transformation

- **Since $y = g(x)$, it suffices to specify the sample points $x_i, i = 1, \cdots, N$ to get data $(x_i, y_i), i = 1, \cdots, N$.**
- **Let $x \in \mathbb{R}^n$ be a random variable with mean μ and covariance matrix Σ.**
- **Define $2n + 1$ points and weighting coefficients: $(\mathcal{X}_i, W_i),\ i = 0, 1, \cdots, 2n$ satisfying**

$$\bar{x} = \sum_{i=0}^{2n} W_i \mathcal{X}_i = \mu, \qquad S_{xx} = \sum_{i=0}^{2n} W_i [\mathcal{X}_i - \mu][\mathcal{X}_i - \mu]^{\mathrm{T}} = \Sigma$$

$(\mathcal{X}_i, W_i),\ i = 0, 1, \cdots, 2n$ are called σ points.

Unscented Transformation



→ Mean Covariances

$$\textbf{mean} \quad \bar{y} = \sum_{i=0}^{2n} W_i g(\mathcal{X}_i)$$

$$\textbf{cov} \quad S_{yx} = \sum_{i=0}^{2n} W_i [g(\mathcal{X}_i) - \bar{y}][\mathcal{X}_i - \bar{x}]^{\mathrm{T}}$$

$$S_{yy} = \sum_{i=0}^{2n} W_i [g(\mathcal{X}_i) - \bar{y}][g(\mathcal{X}_i) - \bar{y}]^{\mathrm{T}}$$

Unscented Transformation

Usually, σ points are determined as

$$\mathcal{X}_0 = \mu, \qquad W_0 = \frac{\lambda}{n+\lambda}$$

$$\mathcal{X}_i = \mu + \left[\sqrt{(n+\lambda)\Sigma}\right]_i, \qquad W_i = \frac{1}{2(n+\lambda)}, \quad i = 1, \cdots, n$$

$$\mathcal{X}_{i+n} = \mu - \left[\sqrt{(n+\lambda)\Sigma}\right]_i, \qquad W_{i+n} = \frac{1}{2(n+\lambda)}, \quad i = 1, \cdots, n$$

$[\sqrt{\cdot}\,]_i$: the ith column vector of matrix square root

- **S. J. Julier, J. K. Uhlmann and H. F. Durrant-Whyte: A new approach for filtering nonlinear system. *Proc. American Control Conference*, Washington, DC., pp. 1628–1632, 1995.**

UKF algorithm

1. Let $\hat{x}_{0|-1} = \bar{x}_0$ **and** $P_{0|-1} = \Sigma_0$**, and generate the initial** σ **points** $(i = 1, \cdots, n)$

$$\hat{x}^{(0)}_{0|-1} = \hat{x}_{0|-1}, \quad \hat{x}^{(i)}_{0|-1} = \hat{x}_{0|-1} + \left[\sqrt{(n+\lambda)P_{0|-1}}\right]_i$$

$$\hat{x}^{(i+n)}_{0|-1} = \hat{x}_{0|-1} - \left[\sqrt{(n+\lambda)P_{0|-1}}\right]_i$$

Set $t = 0$**.**

2. Observation update

Input: $[\hat{x}^{(i)}_{t|t-1}, P_{t|t-1}, y_t] \rightarrow$ **Output:** $[\hat{x}_{t|t}, P_{t|t}]$

(a) Predicted estimate of output

$$\hat{y}_{t|t-1} = \sum_{i=0}^{2n} W_h^{(i)} h_t(\hat{x}^{(i)}_{t|t-1})$$

(b) Condtional covariances

$$V_{t|t-1} = \sum_{i=0}^{2n} W_h^{(i)} [h_t(\hat{x}_{t|t-1}^{(i)}) - \hat{y}_{t|t-1}][h_t(\hat{x}_{t|t-1}^{(i)}) - \hat{y}_{t|t-1}]^{\mathrm{T}} + R_t$$

$$U_{t|t-1} = \sum_{i=0}^{2n} W_h^{(i)} [\hat{x}_{t|t-1}^{(i)} - \hat{x}_{t|t-1}][h_t(\hat{x}_{t|t-1}^{(i)}) - \hat{y}_{t|t-1}]^{\mathrm{T}}$$

(c) UKF gain $K_t = U_{t|t-1} V_{t|t-1}^{-1}$

(d) Filtered estimate

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t[y_t - \hat{y}_{t|t-1}]$$

(e) Filtered covariance matrix

$$P_{t|t} = P_{t|t-1} - U_{t|t-1} V_{t|t-1}^{-1} U_{t|t-1}^{\mathrm{T}}$$

3. Time update

Input: $[\hat{x}_{t|t},\ P_{t|t}]$ $\rightarrow$ **Output:** $[\hat{x}_{t+1|t}^{(i)},\ P_{t+1|t}]$

(a) Generation of σ points $(i = 1, \cdots, n)$

$$\hat{x}_{t|t}^{(0)} = \hat{x}_{t|t}, \quad \hat{x}_{t|t}^{(i)} = \hat{x}_{t|t} + \left[\sqrt{(n+\lambda)P_{t|t}}\right]_i, \quad \hat{x}_{t|t}^{(i+n)} = \hat{x}_{t|t} - \left[\sqrt{(n+\lambda)P_{t|t}}\right]_i$$

(b) Transformation of σ points

$$\hat{x}_{t+1|t}^{(i)} = f_t(\hat{x}_{t|t}^{(i)}), \quad i = 0, 1, \cdots, 2n$$

(c) Predicted estimate

$$\hat{x}_{t+1|t} = \sum_{i=0}^{2n} W_f^{(i)} f_t(\hat{x}_{t|t}^{(i)})$$

(d) Predicted covariance matrix

$$P_{t+1|t} = \sum_{i=0}^{2n} W_f^{(i)} [f_t(\hat{x}_{t|t}^{(i)}) - \hat{x}_{t+1|t}][f_t(\hat{x}_{t|t}^{(i)}) - \hat{x}_{t+1|t}]^{\mathrm{T}} + Q_t$$

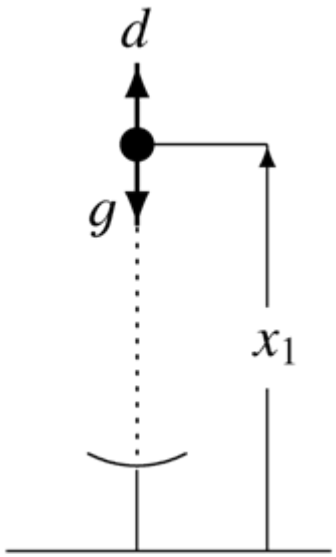
4. Put $t := t + 1$, and go to Step 2.

Classical Ballistic Missile Model

Consider the problem of tracking a body falling freely through the atmosphere (Gelb 1974).

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = d - g, \quad \dot{x}_3 = 0, \quad y = x_1 + v$$

$$d = \frac{\rho_0 e^{-x_1/k_\rho} x_2^2}{2\beta}, \qquad \beta = x_3$$



x_1 = **height of the missle**

x_2 = **velocity**

g = **acceleration of gravity**

β = **ballistic coefficient**

d = **drag deceleration**

k_ρ = **decay constant**

ρ_0 = **atomsheric density at sea level**

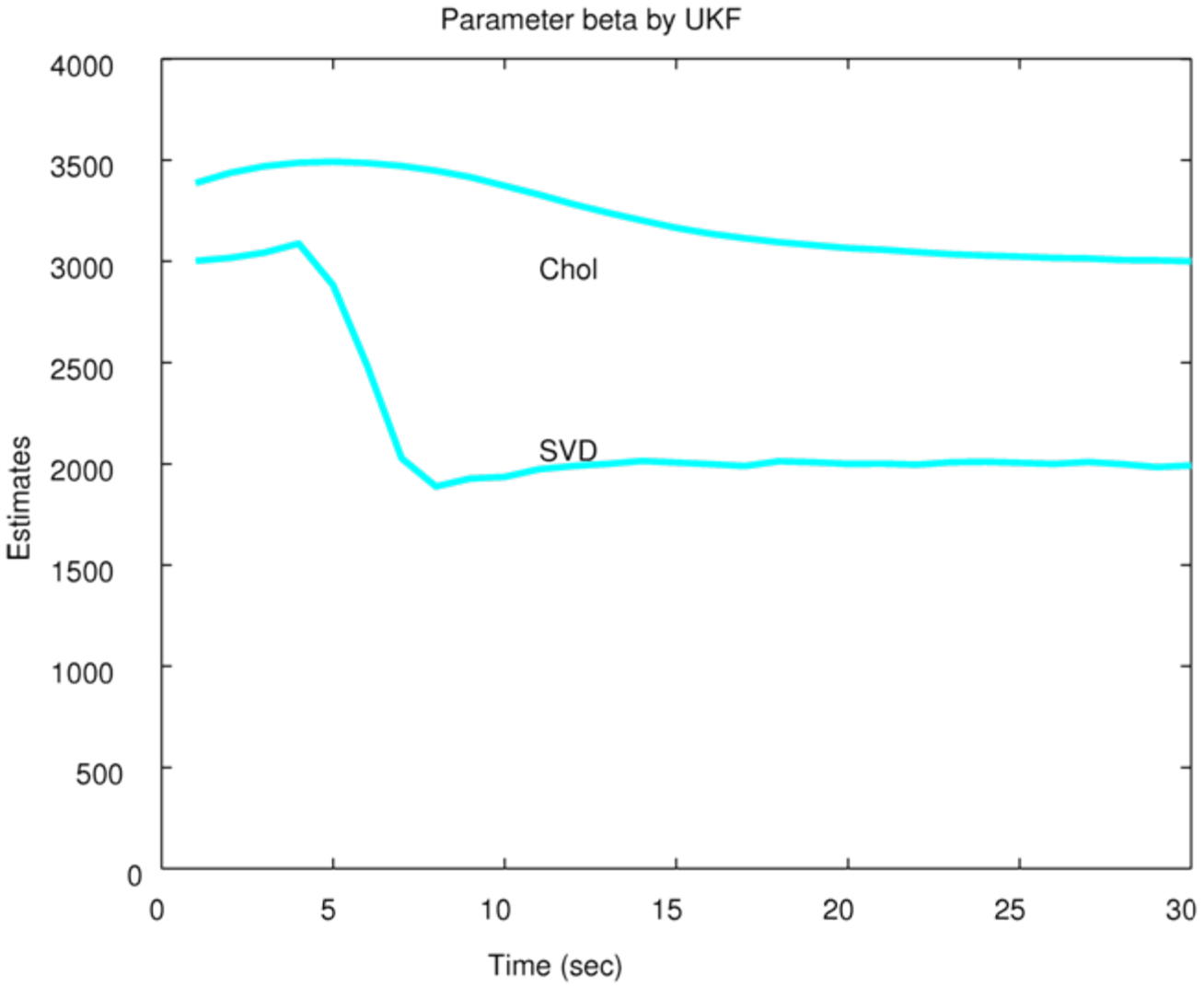
- **Data for simulation**

$$g = 9.8,\ \rho_0 = 2.202,\ k_\rho = 1000/0.1558,\ \beta = 2000$$
$$\Delta = 0.1,\ N = 300$$

$$\begin{bmatrix} x_1(0) \\ x_2(0) \\ x_3(0) \end{bmatrix} = \begin{bmatrix} 40000 \\ -3000 \\ 2000 \end{bmatrix},\ Q = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 10 \end{bmatrix},\ R = 100$$

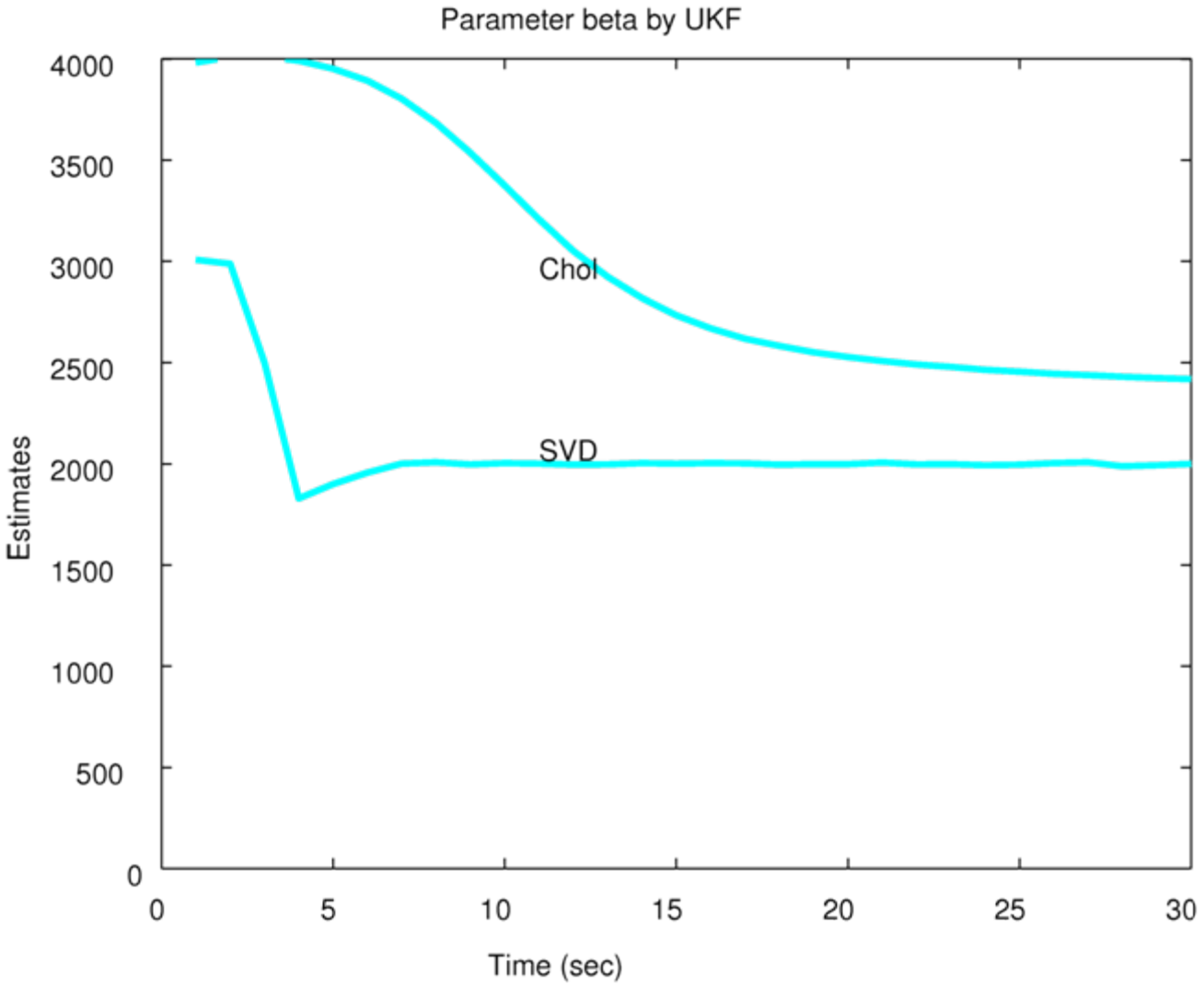
$$x(0|0) = \begin{bmatrix} 40100 \\ -3100 \\ 3000 \end{bmatrix},\ P(0|0) = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 10000 & 0 \\ 0 & 0 & 10000 \end{bmatrix}$$

- **4th-order Runge-Kutta is used for simulation.**



Estimation of parameter β (Chol - SVD)

R=100
UKF
Runge-Kutta



Estimation of parameter β (Chol - SVD)

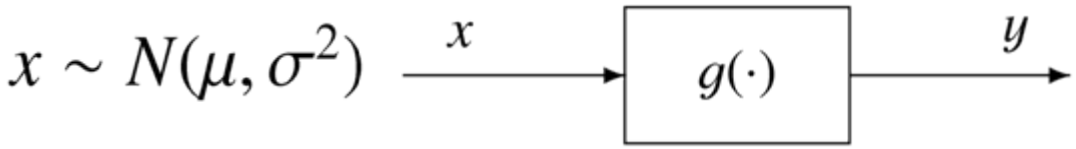
R=1
UKF
Runge-Kutta

Comments

- **Motivated by the encouraging results for a ballistic model, we have started a study of UT through examples for static nonlinearities and then simulated several dynamical models.**

- **We have used SVD and QR decomposition rather than Cholesky decomposition for matrix factorizations. Reading many papers for UKF, however, we noticed that they are based on mostly Cholesky factorization.**

UT: Example n=1



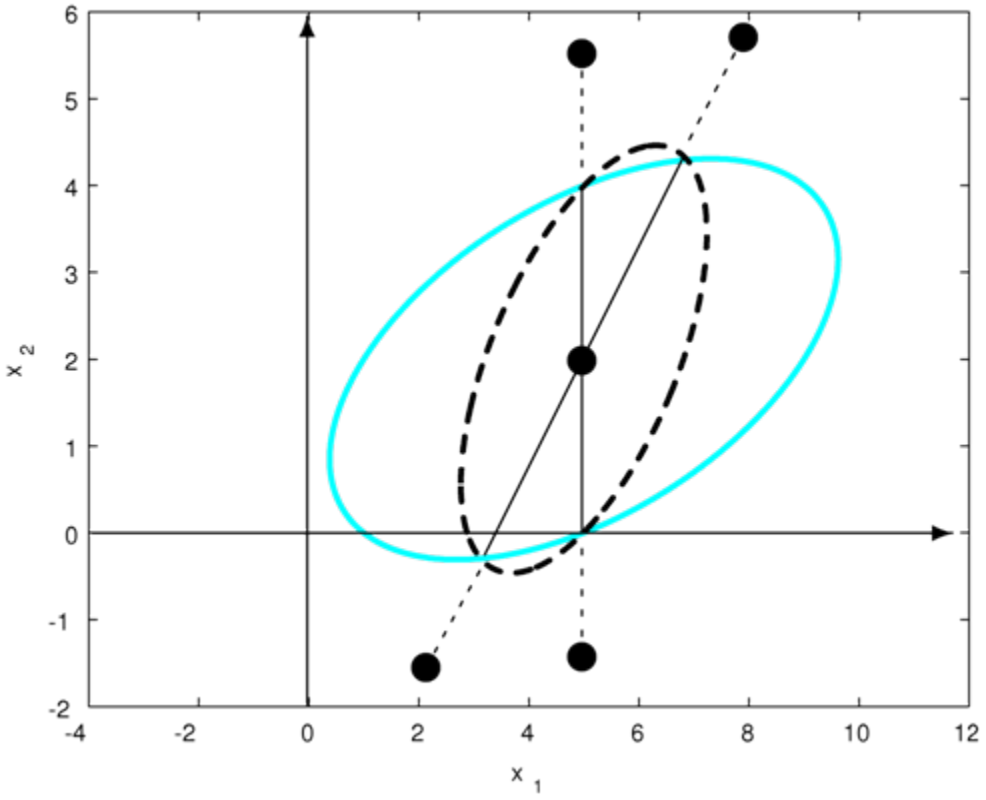
$y = x^2$	μ_y	σ_{yx}	σ_y^2
True	$\mu^2 + \sigma^2$	$2\mu\sigma^2$	$4\mu^2\sigma^2 + 2\sigma^4$
Linear	μ^2	$2\mu\sigma^2$	$4\mu^2\sigma^2$
UT	$\mu^2 + \sigma^2$	$2\mu\sigma^2$	$4\mu^2\sigma^2 + \lambda\sigma^4$

$y = x^3$	μ_y	σ_{yx}	σ_y^2
True	$\mu^3 + 3\mu\sigma^2$	$3\mu^2\sigma^2 + 3\sigma^4$	$9\mu^4\sigma^2 + 36\mu^2\sigma^4 + 15\sigma^6$
Linear	μ^3	$3\mu^2\sigma^2$	$9\mu^4\sigma^2$
UT	$\mu^3 + 3\mu\sigma^2$	$3\mu^2\sigma^2 + (1 + \lambda)\sigma^4$	$9\mu^4\sigma^2 + (6 + 15\lambda)\mu^2\sigma^4 + (1 + \lambda)^2\sigma^6$

UT: Cholesky n=2

$$\Sigma = LL^{\mathrm{T}} = \begin{bmatrix} * & 0 \\ * & * \end{bmatrix} \begin{bmatrix} * & * \\ 0 & * \end{bmatrix}$$

$$L = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} = [l_1 \quad l_2]$$



$$\chi_0 = \mu_x$$

$$\chi_i = \mu_x \pm (n+\lambda)^{1/2} l_i$$

$$(x-\mu)^{\mathrm{T}} \Sigma^{-1} (x-\mu) = C$$

$$(C = 1,\ \mu_1 = 5,\ \mu_2 = 2)$$

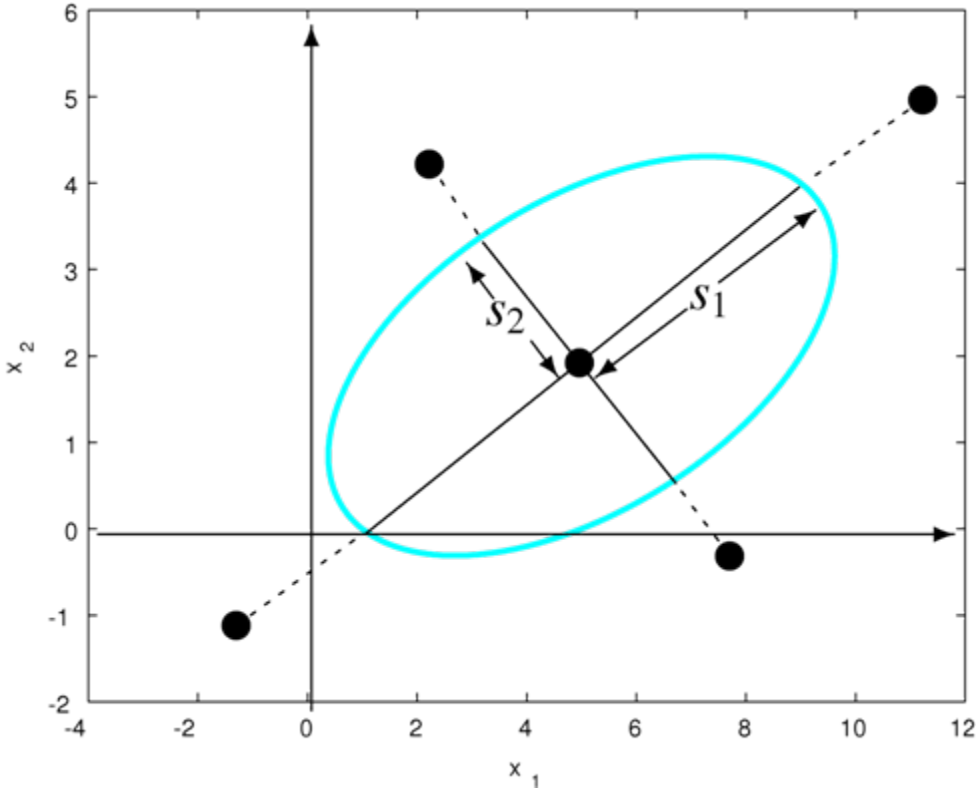
Elipsoid of constant density

UT: SVD n=2

$$\Sigma = USU^{\mathrm{T}} = U \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} U^{\mathrm{T}}$$

$$U^s = U \begin{bmatrix} \sqrt{s_1} & 0 \\ 0 & \sqrt{s_2} \end{bmatrix}$$

$$= [\sqrt{s_1}u_1 \quad \sqrt{s_2}u_2]$$



$$\chi_0 = \mu_x$$

$$\chi_i = \mu_x \pm (n+\lambda)^{1/2} \sqrt{s_i} u_i$$

$$(x-\mu)^{\mathrm{T}} \Sigma^{-1} (x-\mu) = C$$

$$(C = 1,\ \mu_1 = 5,\ \mu_2 = 2)$$

Elipsoid of constant density

UT: Example n=2

$$y_1 = x_1 - x_1 x_2$$

(Lotka-Volterra)

$$y_2 = -0.8x_2 + 1.2x_1 x_2$$

$$\mu_x = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \quad \Sigma_x = \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix}$$

$$\sigma_1^2 = \sigma_2^2 = 1, \ \sigma_{12} = 0.5$$

We compute the 1st- and 2nd-order moments of the output y, where the true, the linear approximation and the UT transformation are shown.

UT: Example n=2

Means of y_1 and y_2

	$\mu_1 = 0$ $\mu_2 = 0$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 2$ $\mu_2 = 2$
True μ_{y_1}	−0.5	−0.5	−2.5
Linear $\overline{y}_{1L}$	0.0	0.0	−2.0
Chol $\overline{y}_{1c}$	−0.4330	−0.4330	−2.4330
SVD $\overline{y}_{1s}$	−0.5000	−0.5000	−2.5000
True μ_{y_2}	0.6	1.0	3.8
Linear $\overline{y}_{2L}$	0.0	0.4	3.2
Chol $\overline{y}_{2c}$	0.5196	0.9196	3.7196
SVD $\overline{y}_{2s}$	0.6000	1.0000	3.8000

$\mathrm{Var}(y_1)$

	$\mu_1 = 0$ $\mu_2 = 0$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 2$ $\mu_2 = 2$
True $\sigma^2_{y_1}$	2.25	2.25	8.25
Linear σ^2_{y1L}	1.0	1.0	7.0
Chol $\lambda_c = 4.5$	2.2812	1.7812	7.0133
Chol $\lambda_c = 6.0$	2.5625	2.0625	7.2946
Chol $\lambda_c = 7.0$	2.7500	2.2500	7.4821
Chol $\lambda_c = 8.0$	2.9375	2.4375	7.6696
Chol $\lambda_c = 11.0$	3.5000	3.0000	8.2321
SVD $\lambda_s = 0.30$	2.1875	2.1875	8.1875
SVD $\lambda_s = 0.35$	2.2188	2.2188	8.2188
SVD $\lambda_s = 0.40$	2.2500	2.2500	8.2500
SVD $\lambda_s = 0.45$	2.2812	2.2812	8.2812

$\mathrm{Var}(y_2)$

	$\mu_1 = 0$ $\mu_2 = 0$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 2$ $\mu_2 = 2$
True σ^2_{y2}	2.44	3.88	13.96
Linear σ^2_{y2L}	0.64	2.08	12.16
Chol $\lambda_c = 3.5$	1.6950	3.5507	13.6605
Chol $\lambda_c = 4.5$	1.9650	3.8207	13.9305
Chol $\lambda_c = 5.5$	2.2350	4.0907	14.2005
Chol $\lambda_c = 6.5$	2.5050	4.3607	14.4705
SVD $\lambda_s = 0.30$	2.3500	3.7900	13.8700
SVD $\lambda_s = 0.35$	2.3950	3.8350	13.9150
SVD $\lambda_s = 0.40$	2.4400	3.8800	13.9600
SVD $\lambda_s = 0.45$	2.4850	3.9250	14.0050

$\mathrm{Cov}(y_1, y_2)$

	$\mu_1 = 0$ $\mu_2 = 0$	$\mu_1 = 1$ $\mu_2 = 1$	$\mu_1 = 2$ $\mu_2 = 2$
True σ_{y12}	−1.9	−2.5	−10.3
Linear σ_{y12L}	−0.4	−1.0	−8.8
Chol $\lambda_c = 6.0$	−1.9214	−2.3946	−9.7463
Chol $\lambda_c = 6.5$	−2.0339	−2.5071	−9.8588
Chol $\lambda_c = 7.0$	−2.1464	−2.6196	−9.9713
Chol $\lambda_c = 8.5$	−2.4839	−2.9571	−10.3088
SVD $\lambda_s = 0.30$	−1.8250	−2.4250	−10.2250
SVD $\lambda_s = 0.35$	−1.8625	−2.4625	−10.2625
SVD $\lambda_s = 0.40$	−1.9000	−2.5000	−10.3000
SVD $\lambda_s = 0.45$	−1.9375	−2.5375	−10.3375

Comments

- **Numerical results for static nonlinearities above shows that the SVD-based UT method is better than Cholesky-based one.**

- **We show some more examples for dynamic systems. Computational load for SVD-based method is 10% higher than Cholesky decomposition-based method in MATLAB.**

Lotka-Volterra Model

Consider the state and parameter estimation problem for Lotka-Volterra model

$$\dot{x}_1 = r_1 x_1 - a_{12} x_1 x_2 + w_1$$

$$\dot{x}_2 = -r_2 x_2 + a_{21} x_1 x_2 + w_2$$

$$y_1 = x_1 + v_1$$

$$y_2 = x_2 + v_2$$

x_1 = **number of prey** x_2 = **number of predator**

r_1 = **coefficient of growth rate** r_2 = **coefficient of death rate**

a_{12}, a_{21} = **coefficients of interaction**

- **Data for simulation**

$a_{12} = 1,\ a_{21} = 1,\ r_1 = 0.8,\ r_2 = 1.2$

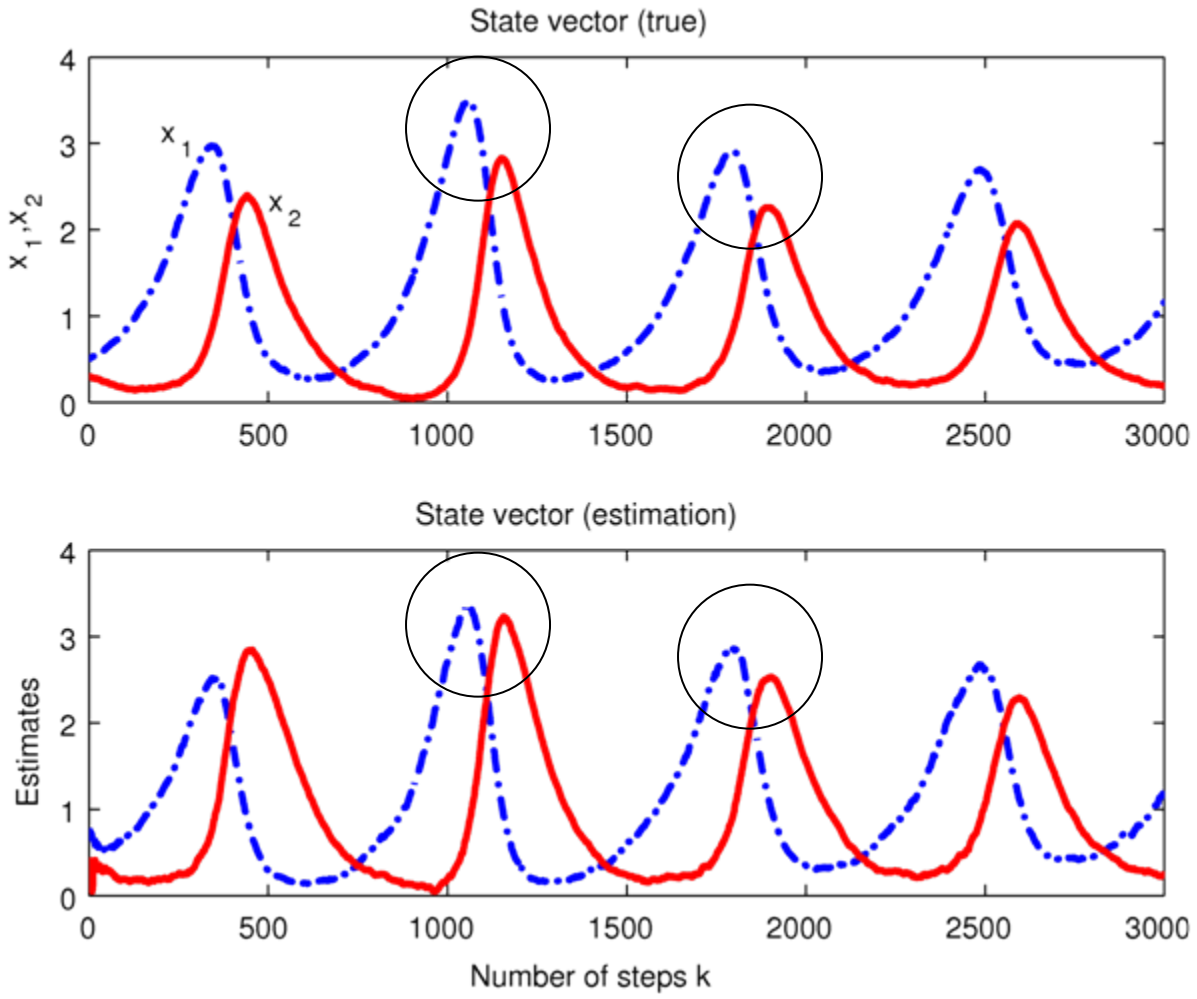
$x_3 = r_1,\ x_4 = r_2,\ \Delta = 0.01,\ N = 3000$

$Q = 10^{-5} I_4,\ R = 0.05 I_2$

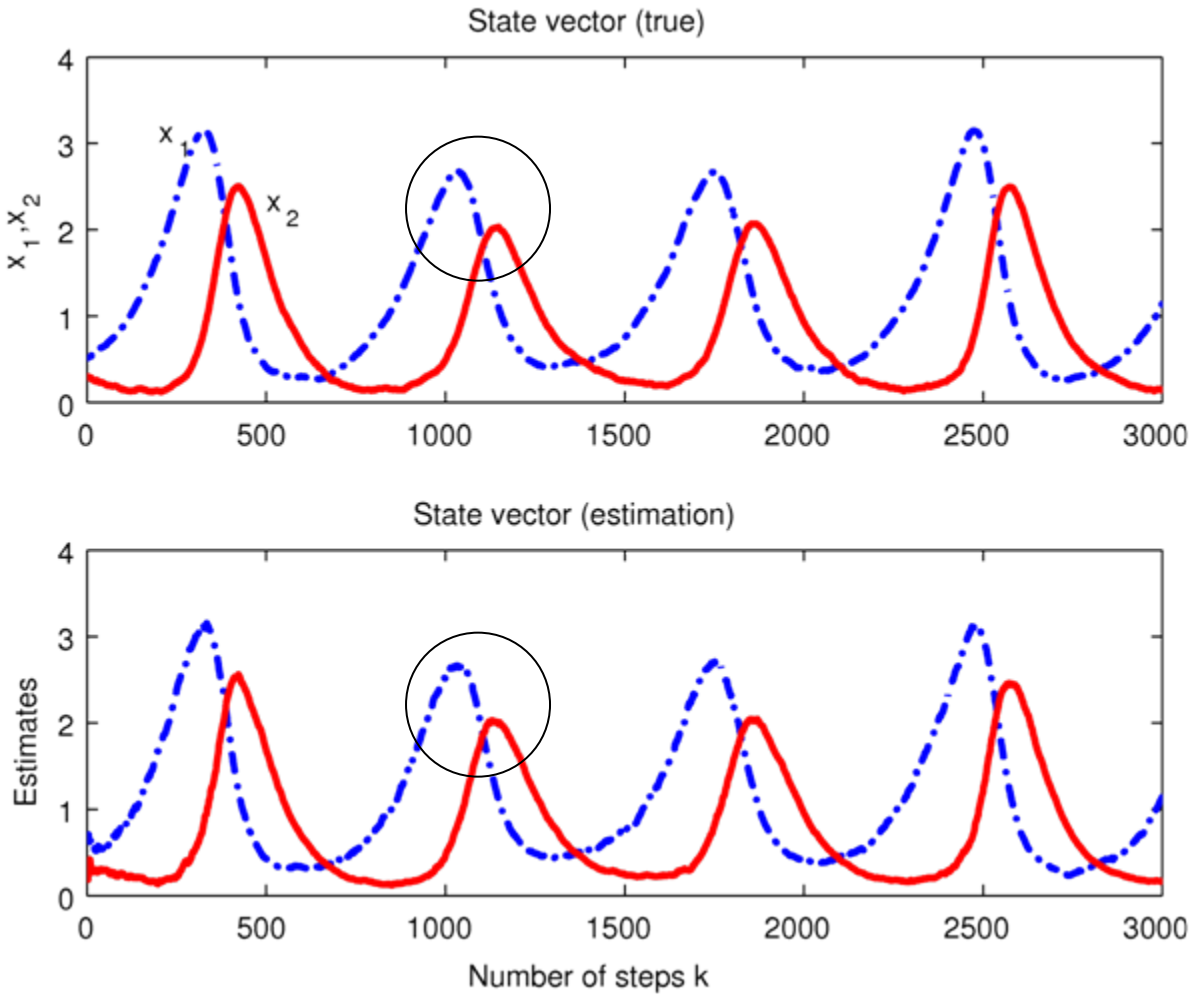
$$x(0) = \begin{bmatrix} 10^{-2} \\ 10^{-2} \\ 10^{-2} \\ 10^{-2} \end{bmatrix}, \quad P(0) = \begin{bmatrix} 1.0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}$$

- **4th-order Runge-Kutta is used for simulation.**

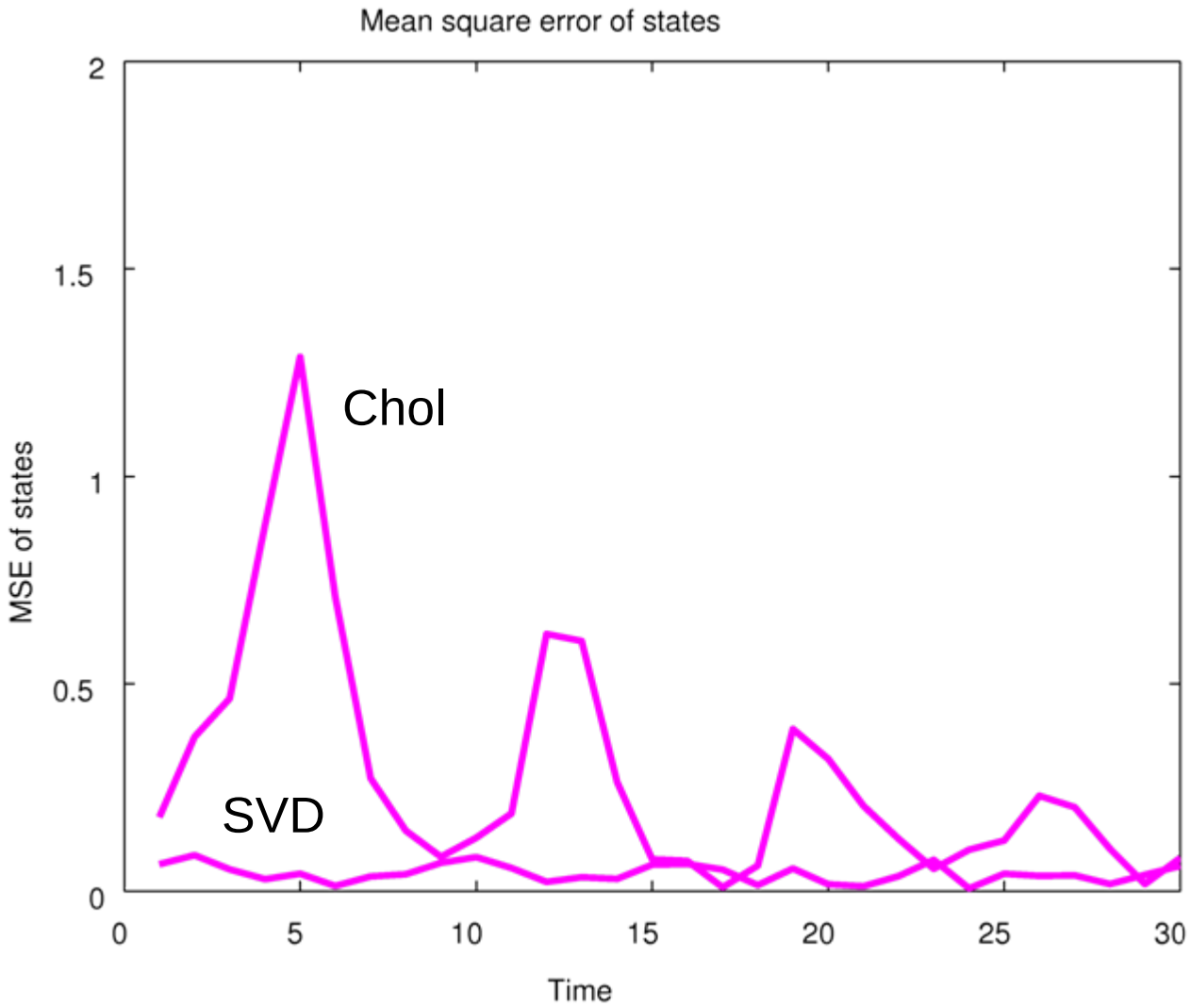
True states (top) and estimates (bottom) by Cholesky



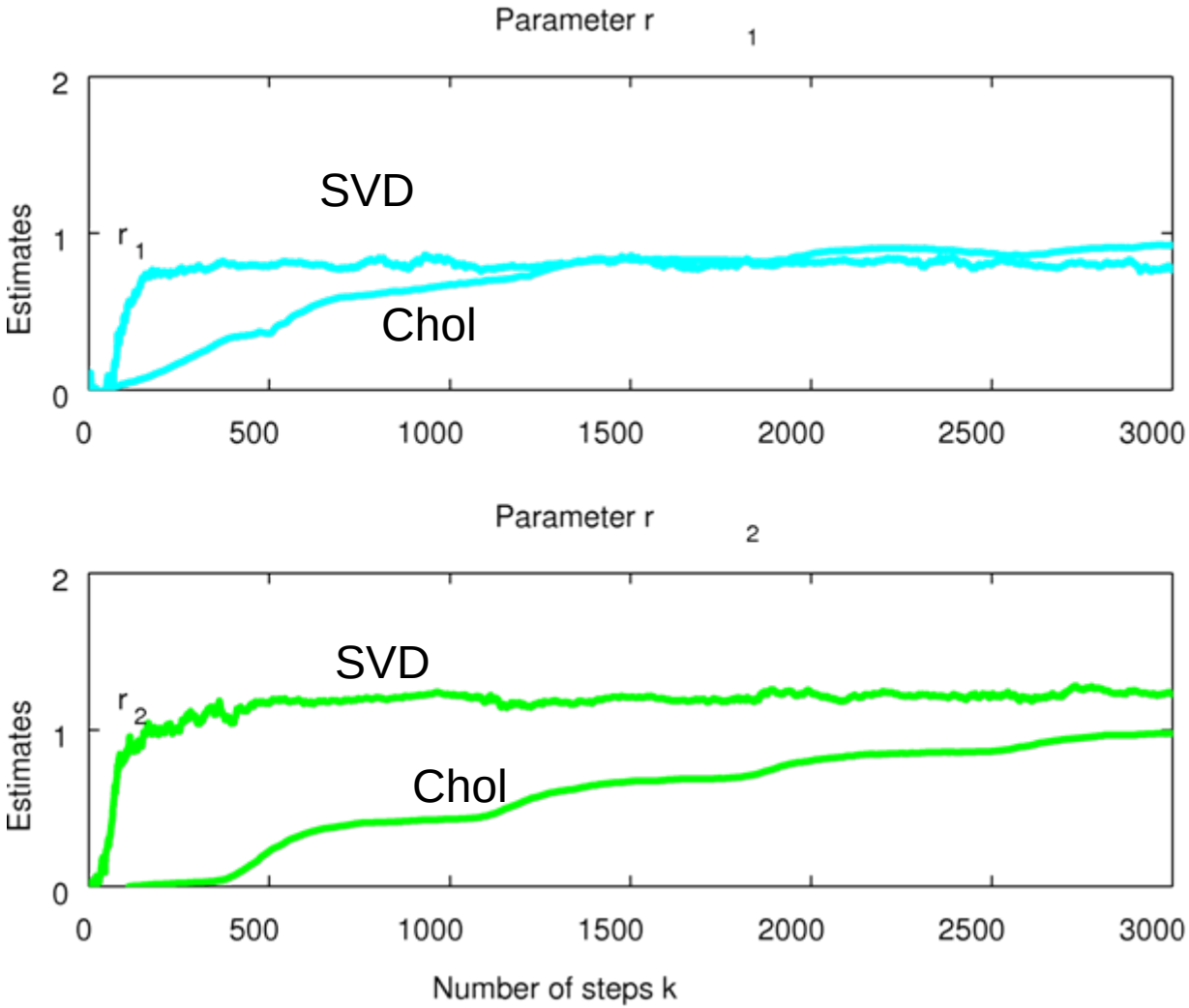
True states (top) and estimates (bottom) by SVD



State estimation errors



Parameter estimation by SVD & Cho



Models for Simulation

1. 1st-order linear system

$$x_{t+1} = ax_t + bu_t + w_t$$

$$y_t = x_t + v_t$$

2. 1st-order NL system

$$\dot{x} = \gamma(u - ax + b)$$

$$y = x + \sin x + v$$

3. Ballistic missile model

4. Lotka-Volterra model

5. Lorenz model

$$\dot{x}_1 = \sigma(x_2 - x_1) + w_1$$

$$\dot{x}_2 = \rho x_1 - x_2 - x_1 x_3 + w_2$$

$$\dot{x}_3 = x_1 x_2 - \beta x_3 + w_3$$

$$y = I_3 x + v$$

6. Two-dim tracking model

$$\dot{x}_1 = x_2 + w_1$$

$$\dot{x}_2 = -x_1 + w_2$$

$$y_1 = \sqrt{(x_1 - \alpha)^2 + x_2^2} + v_1$$

$$y_2 = \tan^{-1}\left(\frac{x_2}{x_1 - \alpha}\right) + v_2$$

Estimation Results for Six Models

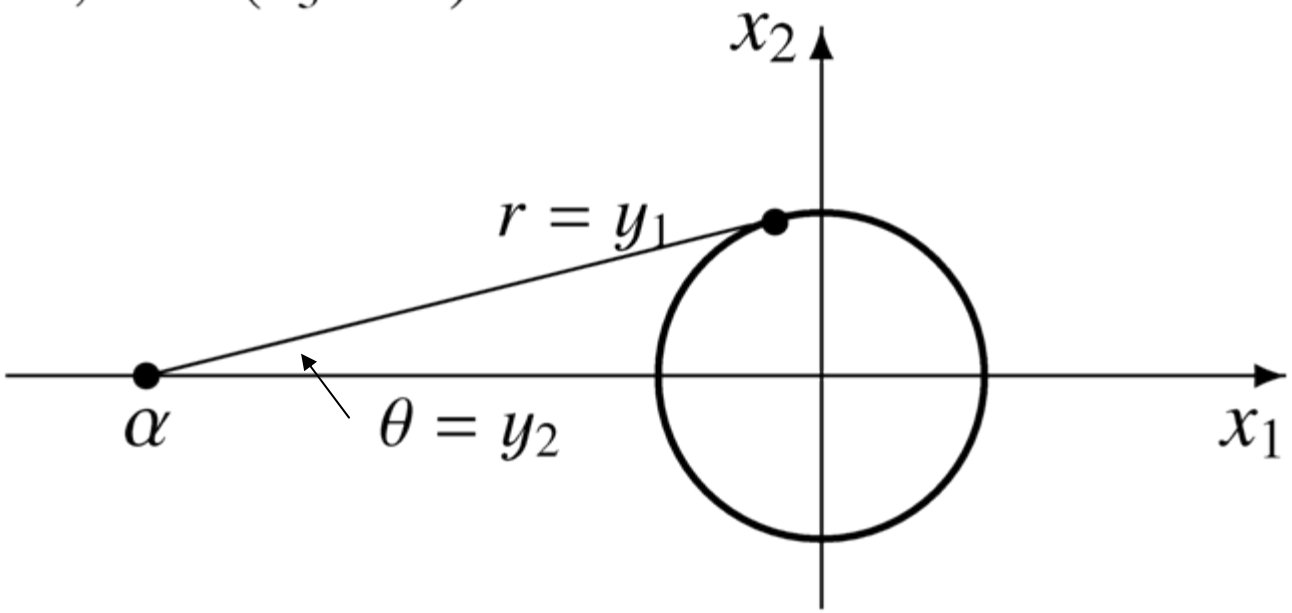
Model	Parameters	EKF	UKF_s	UKF_c
1	a	○	○	○
	b	○	○	×
2	a	×	○	○
	b	○	○	×
3	β	○	○	×
4	r_1, r_2	○	○	×
	a_{12}, a_{21}	○	○	×
5	σ, ρ, β	×	○	×
6	—	○	○	○
		Euler	R-K	R-K

Two-dim Tracking Model

$$\dot{x}_1 = x_2 + w_1, \qquad y_1 = r = \sqrt{(x_1 - \alpha)^2 + x_2^2} + v_1$$

$$\dot{x}_2 = -x_1 + w_2, \qquad y_2 = \theta = \tan^{-1}\left(\frac{x_2}{x_1 - \alpha}\right) + v_2$$

$$\dot{x}_3 = 0, \qquad (x_3 = \alpha)$$



- **Data for simulation**

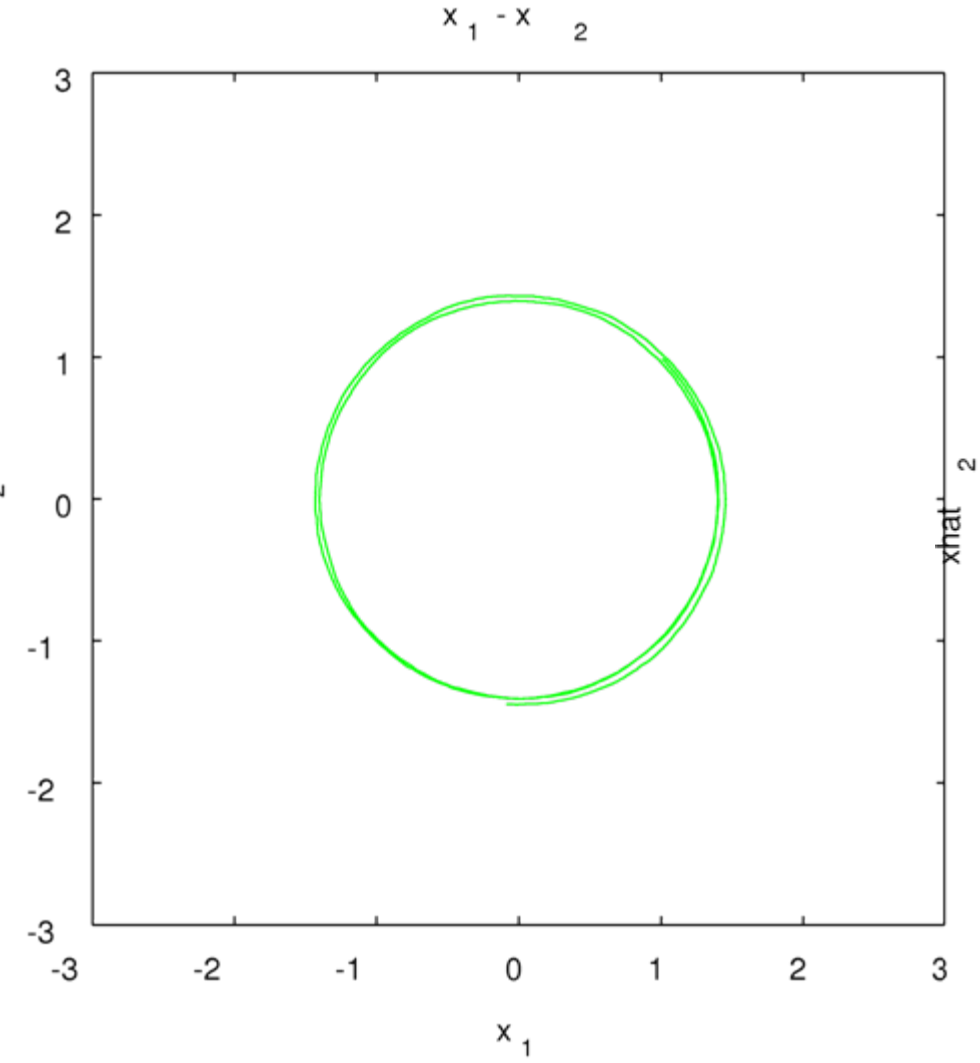
$$\begin{bmatrix} x_1(0) \\ x_2(0) \\ x_3(0) \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 4(=\alpha) \end{bmatrix}, \quad Q = \begin{bmatrix} 10^{-6} & 0 & 0 \\ 0 & 10^{-6} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$R = 10^{-2} I_2$$

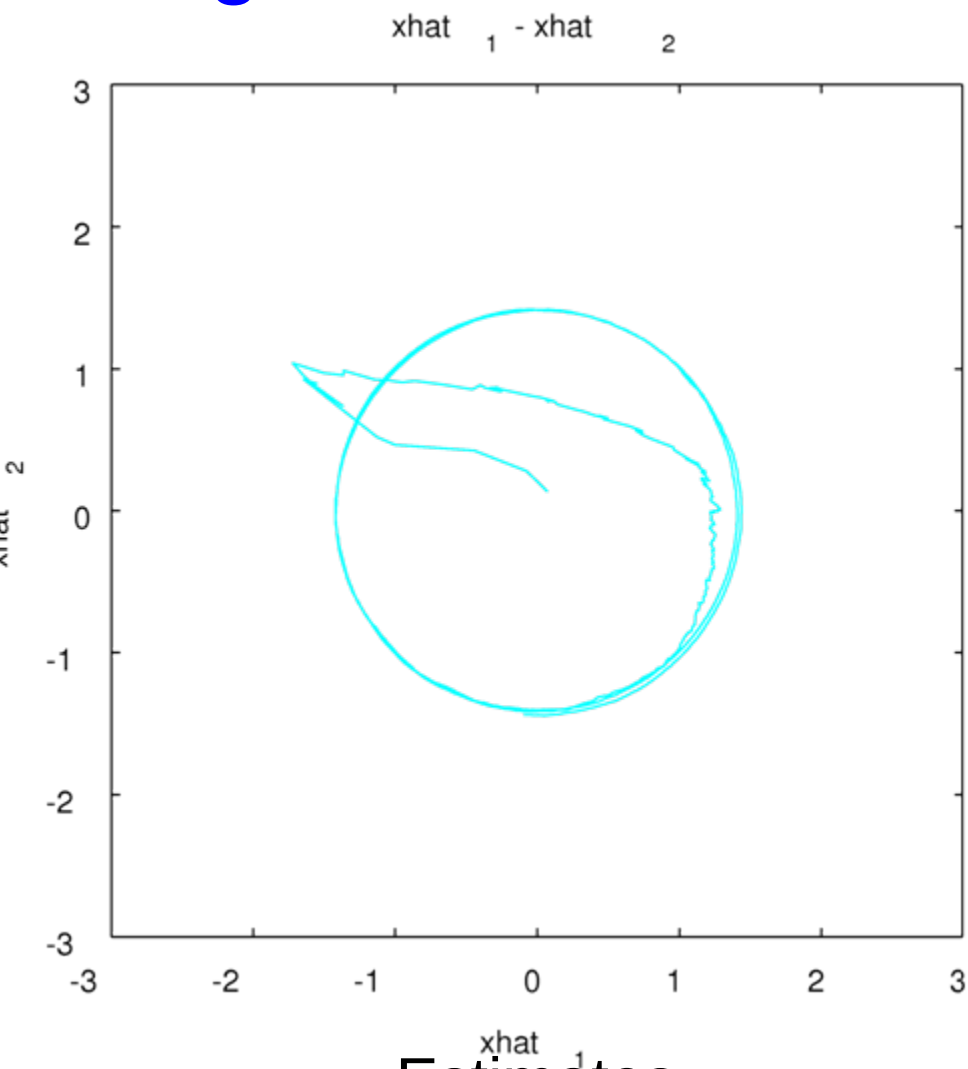
$$\hat{x}(0) = \begin{bmatrix} 10^{-1} \\ 10^{-1} \\ 10^{-1} \end{bmatrix}, \quad P(0) = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 2.0 \end{bmatrix}$$

- **4th-order Runge-Kutta is used for simulation.**
- $T = 15$ **(sec),** $\Delta = 0.01$**,** $\lambda = 1$

Two-dim Tracking Model

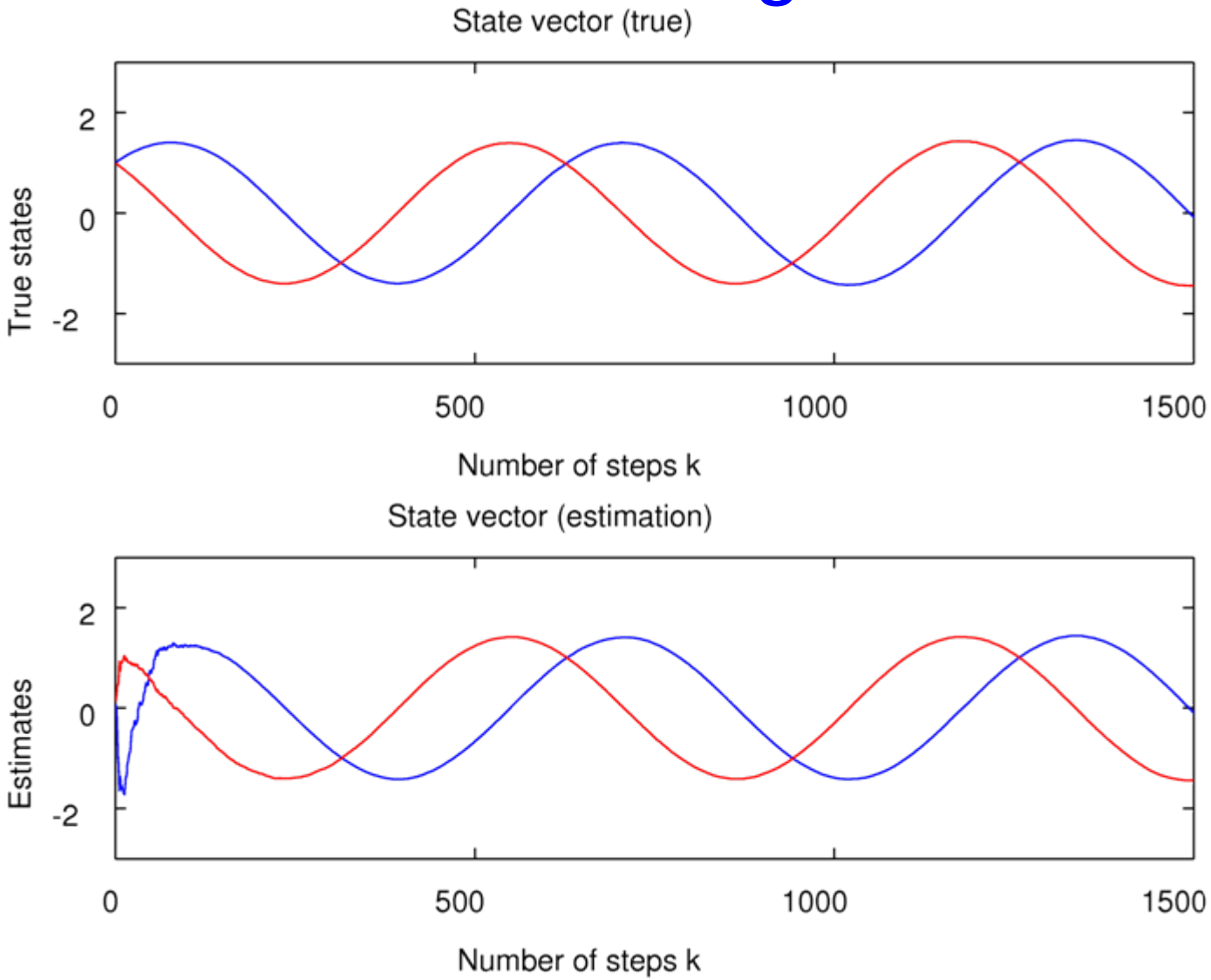


True trajectory

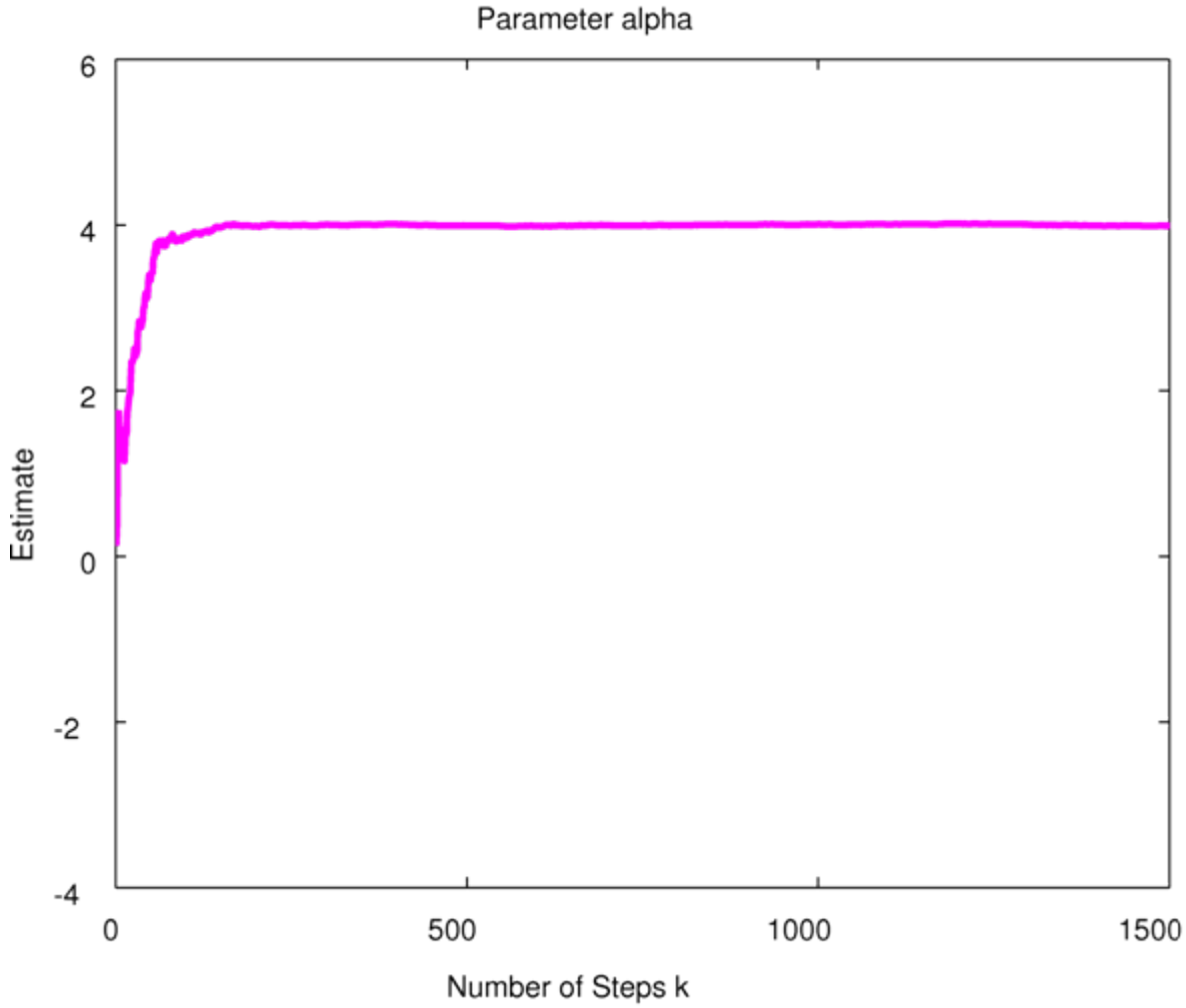


Estimates

Two-dim Tracking Model



Two-dim Tracking Model



Conclusions

- **As shown in examples of UT for static nonlinear transforatiom as well as for many nonlinear dynamical systems, the SVD-based method shows better performance than Cholesky-based method.**

- **Computational load for SVD is 10% higher than that for Cholesky decomposition in MATLAB.**

- **It is however not easy to give a recipe for selecting a suitable value for the parameter λ.**