Chapter 3

Complementary Material

Lemma 3C.1 [1] If a signal $\phi:[0,\infty)\to\mathcal{R}^n$ is PE and satisfies $\phi\in\mathcal{L}_\infty$, then it has the following properties:

(i) $M\phi$, where $M\in\mathcal{R}^{m\times n}$ is a constant matrix and $m\le n$ is PE if and only if $\text{rank}(M)=m$.

(ii) Given any $e:[0,\infty)\to\mathcal{R}^n$ satisfying $e\in\mathcal{L}_2\cap\mathcal{L}_\infty$ and $\lim_{t\to\infty}e(t)=0$, $\phi+e$ is PE.

(iii) Given a stable, minimum-phase, proper rational transfer function $H(s)$, if ϕ satisfies $\dot{\phi}\in\mathcal{L}_\infty$ as well, then $H(s)\phi$ is PE.

3.6.1 Gradient Algorithm with Instantaneous Cost Function

***Proof of Theorem* 3.6.1 (ii), (iii).** (ii) From (3.36) we have

$$V(t+T)=V(t)-\int_t^{t+T}\varepsilon^2 m_s^2 d\tau=V(t)-\int_t^{t+T}\frac{\left(\tilde{\theta}^T(\tau)\phi(\tau)\right)^2}{m_s^2(\tau)}d\tau \tag{3C.1}$$

for any $t,\ T>0$. Let $\bar{\phi}(t)=\frac{\phi(t)}{m_s(t)}$ and consider the identity

$$\tilde{\theta}^T(\tau)\bar{\phi}(\tau)=\tilde{\theta}^T(t)\bar{\phi}(\tau)+\left(\tilde{\theta}(\tau)-\tilde{\theta}(t)\right)^T\bar{\phi}(\tau).$$

Using the inequality $(x+y)^2\ge\frac{1}{2}x^2-y^2,$ we have

$$\begin{aligned}\int_t^{t+T}\left(\tilde{\theta}^T(\tau)\bar{\phi}(\tau)\right)^2 d\tau&=\int_t^{t+T}\left(\tilde{\theta}^T(t)\bar{\phi}(\tau)+\left(\tilde{\theta}(\tau)-\tilde{\theta}(t)\right)^T\bar{\phi}(\tau)\right)^2 d(\tau)\\&\ge\frac{1}{2}\int_t^{t+T}\left(\tilde{\theta}^T(t)\bar{\phi}(\tau)\right)^2 d\tau-\int_t^{t+T}\left(\left(\tilde{\theta}(\tau)-\tilde{\theta}(t)\right)^T\bar{\phi}(\tau)\right)^2 d\tau.\end{aligned} \tag{3C.2}$$

We have that $\frac{\phi}{m_s}=\bar{\phi}$ is PE, which means

$$\int_t^{t+T_0}\bar{\phi}(\tau)\bar{\phi}^T(\tau)d\tau\ge\alpha_0 T_0 I$$

for some $\alpha_0, T_0 > 0$ and $\forall t \geq 0$. Let us take $T = T_0$ in (3C.1), (3C.2) and consider each term in (3C.2) separately. The first term satisfies

$$\begin{aligned} \frac{1}{2}\int_t^{t+T_0} \left(\tilde{\theta}^T(t)\bar{\phi}(\tau)\right)^2 d\tau &= \frac{1}{2}\tilde{\theta}^T(t)\int_t^{t+T_0} \bar{\phi}(\tau)\bar{\phi}^T(\tau)d\tau\ \tilde{\theta}(t) \\ &\geq \frac{\alpha_0}{2}T_0\tilde{\theta}^T(t)\tilde{\theta}(t) \geq \alpha_0 T_0 \lambda_{\min}(\Gamma)V(t). \end{aligned} \tag{3C.3}$$

For the second term on the right-hand side of (3C.2), we use the following expressions:

$$\tilde{\theta}(\tau) - \tilde{\theta}(t) = \int_t^{\tau} \dot{\tilde{\theta}}(\sigma)d\sigma = \int_t^{\tau} \Gamma\varepsilon\phi\ d\sigma = -\int_t^{\tau} \Gamma\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}(\sigma)\ d\sigma.$$

Then

$$\left[\tilde{\theta}(\tau) - \tilde{\theta}(t)\right]^T \bar{\phi}(\tau) = -\int_t^{\tau} \tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}^T(\tau)\Gamma\bar{\phi}(\sigma)\ d\sigma.$$

The second term in (3C.2) with $T = T_0$ satisfies

$$\begin{aligned} \int_t^{t+T_0} \left(\left[\tilde{\theta}(\tau) - \tilde{\theta}(t)\right]^T \bar{\phi}(\tau)\right)^2 d\tau &= \int_t^{t+T_o} \left(\int_t^{\tau} \tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}^T(\tau)\Gamma\bar{\phi}(\sigma)\ d\sigma\right)^2 d\tau \\ &\leq \int_t^{t+T_0} \left(\int_t^{\tau} \left(\bar{\phi}^T(\tau)\Gamma\bar{\phi}(\sigma)\right)^2 d\sigma \int_t^{\tau} \left(\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma\right) d\tau, \end{aligned}$$

where the inequality is obtained using the Schwarz inequality (see Chapter 2).

Since $\bar{\phi} \in \mathcal{L}_\infty$, i.e., $\beta = \sup_{\tau \geq 0} |\bar{\phi}(\tau)|$ is a finite constant, we have

$$\begin{aligned} \int_t^{t+T_0} \left(\left(\tilde{\theta}(\tau) - \tilde{\theta}(t)\right)^T \bar{\phi}(\tau)\right)^2 d\tau &\leq \beta^4 \lambda_{\max}^2(\Gamma) \int_t^{t+T_0} (\tau - t) \int_t^{\tau} \left(\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma d\tau \\ &= \beta^4 \lambda_{\max}^2(\Gamma) \int_t^{t+T_0} \left(\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\right)^2 \int_{\sigma}^{t+T_0} (\tau - t) d\tau d\sigma \\ &= \beta^4 \lambda_{\max}^2(\Gamma) \int_t^{t+T_0} \left(\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\right)^2 \left(\frac{T_0^2 - (\sigma - t)^2}{2}\right) d\sigma \\ &\leq \beta^4 \lambda_{\max}^2(\Gamma) \frac{T_0^2}{2} \int_t^{t+T_0} \left(\tilde{\theta}^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma. \end{aligned} \tag{3C.4}$$

Using (3C.3), (3C.4) in (3C.2) with $T = T_0$ we have

$$\int_t^{t+T_0} \left(\tilde{\theta}(\tau)^T \bar{\phi}(\tau)\right)^2 d\tau \geq \alpha_0 T_0 \lambda_{\min}(\Gamma)V(t) - \beta^4 \lambda_{\max}^2(\Gamma)\frac{T_0^2}{2} \int_t^{t+T_0} \left(\tilde{\theta}(\tau)^T \bar{\phi}(\tau)\right)^2 d\tau,$$

which implies that

$$\int_t^{t+T_0} \left(\tilde{\theta}(\tau)^T \bar{\phi}(\tau)\right)^2 d\tau \geq \frac{2\alpha_0 T_0 \lambda_{\min}(\Gamma)}{2+\beta^4 \lambda_{\max}^2(\Gamma) T_0^2} V(t) = \gamma_1 V(t), \tag{3C.5}$$

where $\gamma_1 = \frac{2\alpha_0 T_0 \lambda_{\min}(\Gamma)}{2+\beta^4 \lambda_{\max}^2(\Gamma) T_0^2}$.

Using (3C.5) in (3C.1) with $T = T_0$, we have

$$V(t+T_0) \leq V(t) - \gamma_1 V(t) = (1-\gamma_1) V(t). \tag{3C.6}$$

Since $\gamma_1 > 0$ and $V(t+T_0) \geq 0$ it follows that $0 < \gamma_1 < 1$. (Note that γ_1 cannot be equal to 1 since that would imply $V(t+T_0) = 0 \; \forall t \geq 0$, which is possible only when $\tilde{\theta}(t+T_0) = 0$, i.e., $\theta(t+T_0) = \theta^* \;\; \forall \geq 0$, which in turn implies that no adaptation is necessary.) Since (3C.6) holds for all $t \geq 0$ we can take $t = (n-1)$, where $n = 0,1,2,3,\ldots,$ to obtain

$$V(t) \leq V(nT_0) \leq (1-\gamma_1) V((n-1)T_0) \leq \cdots \leq (1-\gamma_1)^n V(0)$$

$\forall t \geq nT_0$. Hence $V(t) \to 0$ as $t \to \infty$ exponentially fast, which implies that $\theta(t) \to \theta^*$ exponentially fast.

(iii) The regressor vector ϕ in (3.27) is of the form

$$\phi = H(s)u,$$

where $H(s) = \left[\frac{s^m}{\Lambda(s)}, \ldots, \frac{1}{\Lambda(s)}, \; -\frac{s^{n-1}G(s)}{\Lambda(s)}, \ldots, -\frac{G(s)}{\Lambda(s)}\right]^T$. All elements of $H(s)$ are strictly proper with stable poles (due to stable poles of $G(s)$ and $\frac{1}{\Lambda(s)}$). It can be shown (see [**1**, p. 263]) that if $G(s)$ has no zero-pole cancellations, then the vectors $H(j\omega_1), \ldots, H(j\omega_{n+m+1})$ are linearly independent in $\mathcal{C}^{n+m+1}$ for any $\omega_1, \omega_2, \ldots, \omega_{n+m+1}$ with $\omega_i \neq \omega_j$ for $i \neq j$. Applying Theorem 3.43, we have that ϕ is PE if and only if u is sufficiently rich of order $n+m+1$. Since u is bounded and the plant is stable, ϕ, m_s are bounded and

$$\int_t^{t+T_0} \frac{\phi\phi^T}{m_s^2} d\tau \geq \frac{1}{m_0} \int_t^{t+T_0} \phi\phi^T d\tau \geq \frac{\alpha_0}{m_0} T_0 I,$$

where $m_0 = \sup_\tau m_s^2(\tau)$, which implies that $\frac{\phi}{m_s}$ is PE too. Since ϕ is PE it follows that $|\theta(t) - \theta^*| \to 0$ exponentially fast. From (3.31) we have $\varepsilon m_s = -\frac{\tilde{\theta}^T \phi}{m_s}$, where $\tilde{\theta} = \theta - \theta^*$. Since $\frac{\phi}{m_s} \in \mathcal{L}_\infty$ and $\tilde{\theta}(t) \to 0$ exponentially fast, it follows that εm_s and therefore ε converges to zero exponentially fast. □

3.6.2 Gradient Algorithm with Integral Cost Function

***Proof of Theorem* 3.6.7 (i), (ii).** Since $\frac{\phi}{m_s} \in \mathcal{L}_\infty$, we immediately have $R, Q \in \mathcal{L}_\infty$. Therefore, the system behaves as an LTV system with bounded input. Substituting for $z = \phi^T \theta^*$ in the adaptive law equations, we get

$$\begin{aligned} R(t) &= \int_0^t e^{-\beta(t-\tau)} \frac{\phi(\tau)\phi^T(\tau)}{m_s^2(\tau)} d\tau, \\ Q(t) &= -R(t)\theta^* \end{aligned} \tag{3C.7}$$

and, therefore,

$$\dot{\tilde{\theta}}(t) = \dot{\theta}(t) = -\Gamma R(t)\tilde{\theta}(t). \tag{3C.8}$$

Consider the Lyapunov-like function

$$V\left(\tilde{\theta}\right) = \frac{\tilde{\theta}^T \Gamma^{-1} \tilde{\theta}}{2}.$$

The time derivative of V satisfies

$$\dot{V}(t) = -\tilde{\theta}^T(t) R(t) \tilde{\theta}(t) \le 0 \quad \forall t \ge 0. \tag{3C.9}$$

Since $V > 0$ and $\dot{V} \le 0$, we have $V, \tilde{\theta}, \theta \in \mathcal{L}_\infty$. Integrating both sides of (3C.9), we establish that $\left(\tilde{\theta}^T R \tilde{\theta}\right)^{\frac{1}{2}} = \left| R^{\frac{1}{2}} \tilde{\theta} \right| \in \mathcal{L}_2$. From $\varepsilon = -\frac{\tilde{\theta}^T \phi}{m_s^2}$ and $\tilde{\theta}, \frac{\phi}{m_s} \in \mathcal{L}_\infty$ we conclude that $\varepsilon, \varepsilon m_s \in \mathcal{L}_\infty$. On the other hand, from (3C.8) we also have

$$\left|\dot{\theta}\right| \le \left\| \Gamma R^{\frac{1}{2}} \right\| \left| R^{\frac{1}{2}} \tilde{\theta} \right|,$$

which together with $R \in \mathcal{L}_\infty$ and $\left| R^{\frac{1}{2}} \tilde{\theta} \right| \in \mathcal{L}_\infty \cap \mathcal{L}_2$ imply that $\dot{\theta} \in \mathcal{L}_\infty \cap \mathcal{L}_2$. Since $\dot{\tilde{\theta}}, \dot{R} \in \mathcal{L}_\infty$, it follows from (3C.8) that $\ddot{\tilde{\theta}} \in \mathcal{L}_\infty$, which together with $\dot{\tilde{\theta}} \in \mathcal{L}_2$ implies $\lim_{t\to\infty} \left|\dot{\theta}(t)\right| = 0$.

On the other hand, observing that

$$\frac{d}{dt} \tilde{\theta}^T R \tilde{\theta} = \varepsilon^2 m_s^2 - 2\tilde{\theta}^T R \Gamma R \tilde{\theta} - \beta \tilde{\theta}^T R \tilde{\theta},$$

we have

$$\int_0^t \varepsilon^2(\tau) m_s^2(\tau) d\tau = \tilde{\theta}^T(t) R(t) \tilde{\theta}(t) + 2\int_0^t \tilde{\theta}^T(\tau) R(\tau) \Gamma R(\tau) \tilde{\theta}(\tau) d\tau + \beta \int_0^t \tilde{\theta}^T(\tau) R(\tau) \tilde{\theta}(\tau) d\tau.$$

Since $\lim_{t\to\infty}\tilde{\theta}^T(t)R(t)\tilde{\theta}(t)=0$ and $\left|R^{\frac{1}{2}}\tilde{\theta}\right|\in\mathcal{L}_2$ it follows that $\int_0^\infty \varepsilon^2 m_s^2 d\tau<\infty$, i.e., $\varepsilon m_s\in\mathcal{L}_2$. Hence, the proof of (i),(ii) is complete.

(iii) Since ϕ is PE and m_s is bounded, using (3C.7) we have

$$R(t)\ \geq\ \int_{t-T_0}^{t} e^{-\beta(t-\tau)}\frac{\phi(\tau)\phi^T(\tau)}{m_s^2(\tau)}d\tau\ \geq\ \bar{\alpha}_0 e^{-\beta T_0}\int_{t-T_0}^{t}\phi(\tau)\phi^T(\tau)d\tau\ \geq\ \beta_1 e^{-\beta T_0}I$$

for any $t\geq T_0$, where $\beta_1=\alpha_0\bar{\alpha}_0T_0$, $\bar{\alpha}_0=\inf_{t\geq 0}m_s^{-2}(t)$, and $\alpha_0,T_0>0$ are constants given in the definition of PE. Therefore, (3C.9) implies that

$$\dot{V}\leq-\beta_1 e^{-\beta T_0}\tilde{\theta}^T\tilde{\theta}\leq-2\beta_1\lambda_{min}(\Gamma)e^{-\beta T_0}V\ \forall t\geq T_0,$$

and hence

$$V(t)\leq e^{-\alpha(t-T_0)}V(T_0)\ \forall t\geq T_0,$$

where $\alpha=2\beta_1 e^{-\beta T_0}\lambda_{min}(\Gamma)$. Using $\sqrt{2\lambda_{min}(\Gamma)V}\leq|\tilde{\theta}|\leq\sqrt{2\lambda_{max}(\Gamma)V}$, we obtain

$$|\tilde{\theta}(t)|\leq\sqrt{2\lambda_{max}(\Gamma)V(T_0)}e^{-\frac{\alpha}{2}(t-T_0)}\leq\sqrt{\frac{\lambda_{max}(\Gamma)}{\lambda_{min}(\Gamma)}}|\tilde{\theta}(T_0)|e^{-\frac{\alpha}{2}(t-T_0)}\ \forall t\geq T_0,$$

which implies that $\theta(t)\to\theta^*$ as $t\to\infty$ exponentially fast with the rate $\frac{\alpha}{2}$. For $\Gamma=\gamma I$, the rate of convergence is given by

$$\alpha/2=\beta_1\gamma e^{-\beta T_0},$$

which can be made arbitrarily large by increasing the value of γ.

The proof of (iv) is included in the proof of Theorem 3.6.1 and is thus omitted. □

3.7.1 Recursive LS Algorithm with Forgetting Factor

***Proof of Theorem* 3.7.1 [1].** Denoting $\Gamma(t)=P^{-1}(t)$, it follows from (3.45) that

$$\dot{\Gamma}=-\beta\Gamma+\frac{\phi\phi^T}{m_s^2},\ \ \Gamma(0)=P_0^{-1},$$

i.e.,

$$\Gamma(t)=e^{-\beta t}P_0^{-1}+\int_0^t e^{-\beta(t-\tau)}\frac{\phi(\tau)\phi^T(\tau)}{m_s^2(\tau)}d\tau.$$

Therefore, since $\phi(t)$ is PE and $m_s\in\mathcal{L}_\infty$, we have

$$\Gamma(t)\ \geq\ \int_{t-T_0}^{t} e^{-\beta(t-\tau)}\frac{\phi(\tau)\phi^T(\tau)}{m_s^2(\tau)}d\tau\ \geq\ \bar{\alpha}_0 e^{-\beta T_0}\int_{t-T_0}^{t}\phi(\tau)\phi^T(\tau)d\tau\ \geq\ \beta_1 e^{-\beta T_0}I \qquad (3C.10)$$

$\forall t \geq T_0$, where $\bar{\alpha}_0 = \inf_{t\geq 0} m_s^{-2}(t)$, $\beta_1 = \bar{\alpha}_0 \alpha_0 T_0$, and $\alpha_0, T_0 > 0$ are constants given in the definition of PE. For $t \leq T_0$, we have

$$\Gamma(t) \geq e^{-\beta T_0} P_0^{-1} \geq \lambda_{min}(P_0^{-1}) e^{-\beta T_0} I. \tag{3C.11}$$

Using (3C.10), (3C.11), we get

$$\Gamma(t) \geq \gamma_1 I \tag{3C.12}$$

$\forall t \geq 0,$ where $\gamma_1 = \min\left\{\frac{\alpha_0 T_0}{\beta_1}, \lambda_{min}(P_0^{-1})\right\} e^{-\beta T_0}$. Since $\phi \in \mathcal{L}_\infty$, we also have

$$\Gamma(t) \ \leq \ P_0^{-1} + \beta_2 \int_0^t e^{-\beta(t-\tau)} d\tau I \ \leq \ \gamma_2 I \tag{3C.13}$$

for some $\beta_2 > 0$, where $\gamma_2 = \lambda_{max}(P_0^{-1}) + \frac{\beta_2}{\beta} > 0$. Now (3C.12) and (3C.13) can be combined and rewritten as

$$\gamma_2^{-1} I \leq P(t) = \Gamma^{-1}(t) \leq \gamma_1^{-1} I.$$

Hence, $P(t), P^{-1}(t) \in \mathcal{L}_\infty$. Exponential convergence of $\theta(t)$ to θ^* as $t \to \infty$ can be established applying the procedure and arguments used in the proof of Theorem 3.6.1. □

3.7.3 Modified LS Algorithms

***Proof of Theorem* 3.7.4.** For the pure LS algorithm with covariance resetting (3.47), the covariance matrix $P(t)$ has elements that are discontinuous functions of time whose values between discontinuities are defined by the differential equation (3.47). At the discontinuity or resetting point t_r, $P(t_r^+) = P_0 = \rho_0 I$; therefore, $P^{-1}(t_r^+) = \rho_0^{-1} I$. Between discontinuities $\frac{d}{dt} P^{-1}(t) \geq 0$, i.e., $P^{-1}(t_2) - P^{-1}(t_1) \geq 0 \ \ \forall t_2 \geq t_1 \geq 0$ such that $t_r \notin [t_1, t_2]$, which implies that $P^{-1}(t) \geq \rho_0^{-1} I \ \forall t \geq 0$. Because of the resetting, $P(t) \geq \rho_1 I \ \forall t \geq 0$. Therefore, (3.47) guarantees that

$$\rho_0 I \geq P(t) \geq \rho_1 I \Leftrightarrow \rho_1^{-1} I \geq P^{-1}(t) \geq \rho_0^{-1} I \ \forall t \geq 0.$$

Let us now consider the function

$$V(\tilde{\theta}) = \frac{\tilde{\theta}^T P^{-1} \tilde{\theta}}{2}, \tag{3C.14}$$

where P is given by (3.47). Because P^{-1} is a bounded positive definite symmetric matrix, it follows that V is decrescent and radially unbounded in the space of $\tilde{\theta}$. Along with the solution of (3.47) we have

$$\dot{V} = \frac{1}{2} \tilde{\theta}^T \frac{d(P^{-1})}{dt} \tilde{\theta} + \tilde{\theta}^T P^{-1} \dot{\tilde{\theta}} = -\varepsilon^2 m_s^2 + \frac{1}{2} \tilde{\theta}^T \frac{d(P^{-1})}{dt} \tilde{\theta}.$$

Between resetting points we have from (3.47) that $\frac{d(P^{-1})}{dt} = \frac{\phi\phi^T}{m_s^2}$; therefore,

$$\dot{V} = -\varepsilon^2 m_s^2 + \frac{1}{2}\frac{(\tilde{\theta}^T\phi)^2}{m_s^2} = -\frac{\varepsilon^2 m_s^2}{2} \le 0 \tag{3C.15}$$

$\forall t \in [t_1, t_2]$, where $[t_1, t_2]$ is any interval in $[0, \infty)$ for which $t_r \notin [t_1, t_2]$. At the points of discontinuity of P, we have

$$V(t_r^+) - V(t_r) = \frac{1}{2}\tilde{\theta}^\top (P^{-1}(t_r^+) - P^{-1}(t_r))\tilde{\theta}.$$

Since $P^{-1}(t_r^+) = \frac{1}{\rho_0} I, P^{-1}(t_r) \ge \frac{1}{\rho_0} I$, it follows that $V(t_r^+) - V(t_r) \le 0$, which implies that $V \ge 0$ is a nonincreasing function of time for all $t \ge 0$. Hence, $V \in \mathcal{L}_\infty$ and $\lim_{t\to\infty} V(t) = V_\infty < \infty$. Since the points of discontinuities t_r form a set of measure zero, it follows from (3.47) that $\varepsilon m_s, \varepsilon, \varepsilon n_s \in \mathcal{L}_2$. From $V \in \mathcal{L}_\infty$ and $\rho_1^{-1} I \ge P^{-1}(t) \ge \rho_0^{-1} I$ we have $\tilde{\theta} \in \mathcal{L}_\infty$, which implies that $\varepsilon, \varepsilon m_s, \varepsilon n_s \in \mathcal{L}_\infty$. Using $\varepsilon m_s \in \mathcal{L}_\infty \bigcap \mathcal{L}_2$ and $\rho_0 I \ge P \ge \rho_1 I$, we have $\dot{\theta} \in \mathcal{L}_\infty \bigcap \mathcal{L}_2$, and the proof of (i) is, therefore, complete for the pure LS algorithm with covariance resetting.

For the modified LS algorithm with forgetting factor (3.48), the same results can be established by choosing the same Lyapunov-like function as in (3C.14) and using the identity $\frac{dP^{-1}}{dt} = -P^{-1}\dot{P}P^{-1}$ to establish

$$\frac{dP^{-1}}{dt} = \begin{cases} -\beta P^{-1} + \dfrac{\phi\phi^T}{m_s^2} & \text{if } \|P\| \le R_0, \\ 0 & \text{otherwise,} \end{cases}$$

where $P^{-1}(0) = P_0^{-1}$, which leads to

$$\dot{V} = \begin{cases} -\dfrac{\varepsilon^2 m_s^2}{2} - \dfrac{\beta}{2}\tilde{\theta}^T P^{-1}\tilde{\theta} & \text{if } \|P\| \le R_0, \\ -\dfrac{\varepsilon^2 m_s^2}{2} & \text{otherwise.} \end{cases}$$

Because $\dot{V} \le -\frac{\varepsilon^2 m_s^2}{2} \le 0$ and $P(t)$ is bounded and positive definite for all $t \ge 0$, the rest of the analysis for the proof (i) is exactly the same as above. (ii) and (iii) can be proven using exactly the same procedure and arguments as in the proof of Theorem 3.6.1. □

3.8 Parameter Identification based on DPM

***Proof of Theorem* 3.8.1 (ii).** In proving part (i), we have shown that the time derivative $\dot{V}$ of

$$V = \frac{e^T P_c e}{2} + \frac{\tilde{\theta}^T \Gamma^{-1} \tilde{\theta}}{2},$$

where $\Gamma = \Gamma^T > 0$ and $P_c = P_c^T > 0$, satisfies $\dot{V} \le -\overline{\nu}\varepsilon^2$ for some constant $\overline{\nu} > 0$. Defining

$$A(t) = \begin{bmatrix} A_c - b_c c_c^T n_s^2(t) & -b_c \phi^T(t) \\ \Gamma \phi(t) c_c^T & 0 \end{bmatrix}, \quad C = [c_c^T \ 0]^T, \quad P = \frac{1}{2}\begin{bmatrix} P_c & 0 \\ 0 & \Gamma^{-1} \end{bmatrix},$$

and $x = [e^T, \tilde{\theta}^T]^T$, we combine (3.52),

$$\dot{e} = A_c e + b_c \left(-\tilde{\theta}^T \phi - \varepsilon n_s^2\right),$$
$$\varepsilon = c_c^T e,$$

and (3.54),

$$\dot{\theta} = \dot{\tilde{\theta}} = \Gamma \varepsilon \phi,$$

in the compact form

$$\dot{x} = A(t)x, \quad \varepsilon = C^T x$$

and express V and $\dot{V}$ as

$$\begin{aligned} V &= x^T P x, \\ \dot{V} &= 2x^T P A x + x^T \dot{P} x \\ &= x^T (PA + A^T P) x \le -\overline{\nu} x^T C C^T x = -\overline{\nu}\varepsilon^2, \end{aligned}$$

which implies that

$$A^T(t)P + PA(t) + \overline{\nu} C C^T \le 0.$$

Using Theorem A.8.3, we establish that the equilibrium $e_e = 0, \ \tilde{\theta}_e = 0$ of (3.52), (3.54) is u.a.s., provided that (C, A) is a UCO pair. On the other hand, due to Lemma A.8.4, (C, A) is UCO if and only if $(C, \ A + KC^T)$ is UCO, where

$$K = \begin{bmatrix} b_c n_s^2 \\ -\Gamma \phi \end{bmatrix},$$

which is bounded. Hence to show that $e_e = 0, \ \tilde{\theta}_e = 0$ is e.s., we need to show only that $(C, A + KC^T)$ is UCO. The UCO property of $(C, A + KC^T)$ can be proven as follows: We use the identity

$$A + KC = \begin{bmatrix} A_c & -b_c \phi^T(t) \\ 0 & 0 \end{bmatrix}$$

and associate $(C, A+KC^T)$ with the system

$$\begin{aligned} \dot{Y}_1 &= A_c Y_1 - b_c \phi^T Y_2, \\ \dot{Y}_2 &= 0, \\ y_0 &= c_c^T Y_1. \end{aligned} \tag{3C.16}$$

Since the UCO property of the system (3C.16) implies the UCO of $(C, A+KC^T)$ because of Lemma A.8.5, the result follows. □

3.9 Parameter Identification based on B-SPM

***Proof of Theorem* 3.9.1 (iii).** We have that

$$\begin{aligned} z - \hat{z} &= \rho^* \left(\theta^{*T}\phi + z_0\right) - \rho\left(\theta^T\phi + z_0\right) \\ &= -\rho^* \tilde{\theta}^T \phi - \tilde{\rho}\xi, \end{aligned}$$

where $\tilde{\rho} = \rho - \rho^*$. Hence, from (3.56) we have

$$\begin{aligned} \dot{\tilde{\theta}} &= \Gamma \frac{z-\hat{z}}{m_s^2} \phi \operatorname{sgn}(\rho^*) \\ &= -\Gamma \frac{\rho^* \tilde{\theta}^T \phi + \tilde{\rho}\xi}{m_s^2} \phi \operatorname{sgn}(\rho^*) \\ &= -\Gamma \left|\rho^*\right| \frac{\phi\phi^T}{m_s^2} \tilde{\theta} - \Gamma \operatorname{sgn}(\rho^*) \frac{\phi}{m_s} \tilde{\rho} \frac{\xi}{m_s}. \end{aligned} \tag{3C.17}$$

Now (3C.17) can be rewritten as

$$\dot{\tilde{\theta}}(t) = \Gamma A_\theta(t)\tilde{\theta}(t) + \overline{\xi}(t), \tag{3C.18}$$

where $A_\theta(t) \triangleq -\left|\rho^*\right| \overline{\phi}(t)\overline{\phi}^T(t)$, $\overline{\phi}(t) = \frac{\phi(t)}{m_s(t)}$, and $\overline{\xi} \triangleq -\operatorname{sgn}(\rho^*)\Gamma\overline{\phi}(t)\tilde{\rho}\frac{\xi}{m_s}$. Since $\overline{\phi}(t), \tilde{\rho} \in \mathcal{L}_\infty$ and $\frac{\xi}{m_s} \in \mathcal{L}_2 \cap \mathcal{L}_\infty$, we have $\overline{\xi} \in \mathcal{L}_2 \cap \mathcal{L}_\infty$. Let us consider the homogeneous part of (3C.18)

$$\dot{x}(t) = \Gamma A_\theta(t) x(t), \qquad x(0) = \tilde{\theta}(0) \tag{3C.19}$$

and the Lyapunov function

$$V_x = \frac{x^T \Gamma^{-1} x}{2}$$

whose derivative is given by

$$\dot{V}_x = x^T A_\theta x.$$

For any $t, T > 0$,

$$\begin{aligned} V_x(t+T) &= V_x(t) + \int_t^{t+T} \left(x^T(\tau) A_\theta(\tau) x(\tau) \right) d\tau \\ &= V_x(t) - \left|\rho^*\right| \int_t^{t+T} \left(x^T(\tau) \bar{\phi}(\tau) \right)^2 d\tau. \end{aligned} \tag{3C.20}$$

Considering the identity

$$x^T(\tau)\bar{\phi}(\tau) = x^T(t)\bar{\phi}(\tau) + \left(x(\tau) - x(t)\right)^T \bar{\phi}(\tau)$$

and using the inequality $(a+b)^2 \geq \frac{1}{2}a^2 - b^2$, we have

$$\begin{aligned} \int_t^{t+T} \left(x^T(\tau)\bar{\phi}(\tau) \right)^2 d\tau &= \int_t^{t+T} \left(x^T(t)\bar{\phi}(\tau) + \left(x(\tau) - x(t)\right)^T \bar{\phi}(\tau) \right)^2 d(\tau) \\ &\geq \frac{1}{2} \int_t^{t+T} \left(x^T(t)\bar{\phi}(\tau) \right)^2 d\tau - \int_t^{t+T} \left(\left(x(\tau) - x(t)\right)^T \bar{\phi}(\tau) \right)^2 d\tau. \end{aligned} \tag{3C.21}$$

The PE property of $\bar{\phi}$ implies that

$$\int_t^{t+T_0} \bar{\phi}(\tau)\bar{\phi}^T(\tau) d\tau \geq \alpha_0 T_0 I$$

for some α_0 , $T_0 > 0$ and $\forall t \geq 0$. Let us take $T = T_0$ in (3C.20), (3C.21). The first term on the right-hand side of (3C.21) satisfies

$$\begin{aligned} \frac{1}{2} \int_t^{t+T_0} \left(x^T(t)\bar{\phi}(\tau) \right)^2 d\tau &= \frac{1}{2} x^T(t) \int_t^{t+T_0} \bar{\phi}(\tau)\bar{\phi}^T(\tau) d\tau \; x(t) \\ &\geq \frac{\alpha_0}{2} T_0 x^T(t) x(t) \geq \alpha_0 T_0 \lambda_{\min}(\Gamma) V_x(t). \end{aligned} \tag{3C.22}$$

Let us consider the second term on the right-hand side of (3C.21) . We start with

$$x(\tau) - x(t) = \int_t^\tau \dot{x}(\sigma) d\sigma = \int_t^\tau \Gamma A_\theta(\sigma) x(\sigma) \, d\sigma = -\left|\rho^*\right| \Gamma \int_t^\tau x^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}(\sigma) d\sigma$$

and obtain

$$\left[x(\tau) - x(t)\right]^T \bar{\phi}(\tau) = -\left|\rho^*\right| \int_t^\tau x^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}^T(\tau)\Gamma\bar{\phi}(\sigma) d\sigma.$$

Therefore, for $T = T_0$ the second term on the right-hand side of (3C.21) satisfies

$$\int_t^{t+T_0}\left(\left[x(\tau)-x(t)\right]^T\bar{\phi}(\tau)\right)^2 d\tau = \int_t^{t+T_o}\left(\left|\rho^*\right|\int_t^{\tau} x^T(\sigma)\bar{\phi}(\sigma)\bar{\phi}^T(\tau)\Gamma x(\sigma)d\sigma\right)^2 d\tau$$
$$\le \left|\rho^*\right|^2\int_t^{t+T_0}\left(\int_t^{\tau}\left(\bar{\phi}^T(\tau)\Gamma\bar{\phi}(\sigma)\ \right)^2 d\sigma\int_t^{\tau}\left(x^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma\right)d\tau.$$

Since $\bar{\phi}\in\mathcal{L}_\infty$, i.e., $\beta_\phi = \sup_{\tau\ge 0}\left|\bar{\phi}(\tau)\right|$ is a finite constant, we have

$$\begin{aligned}\int_t^{t+T_0}\left(\left(x(\tau)-x(t)\right)^T\bar{\phi}(\tau)\right)^2 d\tau &\le \left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)\int_t^{t+T_0}(\tau-t)\int_t^{\tau}\left(x^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma d\tau\\ &= \left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)\int_t^{t+T_0}\left(x^T(\sigma)\bar{\phi}(\sigma)\right)^2\int_\sigma^{t+T_0}(\tau-t)d\tau d\sigma\\ &= \left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)\int_t^{t+T_0}\left(x^T(\sigma)\bar{\phi}(\sigma)\right)^2\left(\frac{T_0^2-(\sigma-t)^2}{2}\right)d\sigma\\ &\le \frac{\left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)T_0^{\,2}}{2}\int_t^{t+T_0}\left(x^T(\sigma)\bar{\phi}(\sigma)\right)^2 d\sigma.\end{aligned} \tag{3C.23}$$

Using (3C.22), (3C.23) in (3C.21) with $T=T_0$, we have

$$\int_t^{t+T_0}\left(x(\tau)^T\bar{\phi}(\tau)\right)^2 d\tau \ge \alpha_0 T_0\lambda_{\min}(\Gamma)V_x(t) - \frac{\left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)T_0^{\,2}}{2}\int_t^{t+T_0}\left(x^T(\tau)\bar{\phi}(\tau)\right)^2 d\tau,$$

which implies that

$$\int_t^{t+T_0}\left(x(\tau)^T\bar{\phi}(\tau)\right)^2 d\tau \ge \frac{2\alpha_0 T_0\lambda_{\min}(\Gamma)}{2+\left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)T_0^2}V_x(t) = \gamma_1 V_x(t), \tag{3C.24}$$

where $\gamma_1 = \frac{2\alpha_0 T_0\lambda_{\min}(\Gamma)}{2+\left|\rho^*\right|^2\beta_\phi^4\lambda_{\max}^2(\Gamma)T_0^2}$. Using (3C.24) in (3C.20) with $T=T_0$, we have

$$V_x(t+T_0)\le(1-\gamma_1)V_x(t)\ \ \forall t\ge 0;$$

i.e.,

$$V_x(t)\le V_x(nT_0)\le(1-\gamma_1)V_x\left((n-1)T_0\right)\le\cdots\le(1-\gamma_1)^n V_x(0)\ \ \forall t\ge nT_0,\ n=1,2,\ldots.$$

Since $\gamma_1>0$ and $V_x(t+T_0)>0$ for $x\ne 0$ it follows that $0<\gamma_1<1$, and hence $V(t)\to 0$ as $t\to\infty$ exponentially fast, which implies that $x(t)\to 0$ as $t\to\infty$ exponentially fast; i.e., the transition matrix $\Phi(t,t_0)$ of (3C.18) satisfies

$$\left\|\Phi(t,t_0)\right\|\le\bar{\alpha}e^{-\bar{\gamma}(t-t_0)}$$

for some constants $\overline{\alpha},\overline{\gamma}>0$. Since the homogeneous part of (3C.21) is e.s. and its input $\overline{\xi}\in\mathcal{L}_2\cap\mathcal{L}_\infty$, it follows from the results of the Appendix that $\tilde{\theta}(t)\to 0$ as $t\to\infty$. □

3.12.4 Dead Zone

***Proof of Theorem* 3.12.4 [1].** (i) As before, we consider the Lyapunov-like function

$$V\left(\tilde{\theta}\right)=\frac{\tilde{\theta}^T\Gamma^{-1}\tilde{\theta}}{2}.$$

Using $\varepsilon m_s^2=-\tilde{\theta}^T\phi+\eta$, the time derivative of V along the solution of (3.103) is derived as

$$\dot{V}=\tilde{\theta}^T\phi\left(\varepsilon+g\right)=-\left(\varepsilon m_s^2-\eta\right)\left(\varepsilon+g\right). \tag{3C.25}$$

Since

$$\left(\varepsilon m_s^2-\eta\right)\left(\varepsilon+g\right)=\begin{cases}\left(\varepsilon m_s+g_0\right)^2-\left(g_0+\dfrac{\eta}{m_s}\right)\left(\varepsilon m_s+g_0\right)>0 & \text{if } \varepsilon m_s<-g_0,\\ \left(\varepsilon m_s-g_0\right)^2+\left(g_0-\dfrac{\eta}{m_s}\right)\left(\varepsilon m_s-g_0\right)>0 & \text{if } \varepsilon m_s>g_0,\\ 0 & \text{otherwise,}\end{cases} \tag{3C.26}$$

we have $(\varepsilon m_s^2-\eta)(\varepsilon+g)\ge 0$ and hence $\dot{V}\le 0$, which implies that $V,\theta,\tilde{\theta},\varepsilon,\varepsilon m_s$, $\dot{\theta}\in\mathcal{L}_\infty$ and $\sqrt{(\varepsilon m_s^2-\eta)(\varepsilon+g)}\in\mathcal{L}_2$.

(ii), (iii) From (3.103) we have

$$\dot{\theta}^T\dot{\theta}=\frac{\phi^T\Gamma\Gamma\phi}{m_s^2}\left(\varepsilon+g\right)^2m_s^2, \tag{3C.27}$$

where

$$\left(\varepsilon+g\right)^2m_s^2=\left(\varepsilon m_s+gm_s\right)^2=\begin{cases}\left(\varepsilon m_s+g_0\right)^2 & \text{if } \varepsilon m_s<-g_0,\\ \left(\varepsilon m_s-g_0\right)^2 & \text{if } \varepsilon m_s>g_0,\\ 0 & \text{if } -g_0\le\varepsilon m_s\le g_0.\end{cases}$$

(3C.26) and (3C.27) imply that $0\le(\varepsilon+g)^2m_s^2\le(\varepsilon m_s^2-\eta)(\varepsilon+g)$, i.e., $(\varepsilon+g)m_s\in\mathcal{L}_2$ and hence $\dot{\theta}\in\mathcal{L}_2$, since $\frac{\phi}{m_s}\in\mathcal{L}_\infty$. Now (3C.25) can be rewritten as

$$\dot{V}\le-\varepsilon^2m_s^2+\left|\varepsilon m_s\right|\frac{\left|\eta\right|}{m_s}+\left|\varepsilon m_s\right|g_0+\frac{\left|\eta\right|}{m_s}g_0$$

using $\left|g\right|\le\frac{g_0}{m_s}$. Completing the squares, we obtain

$$\begin{aligned}\dot{V} &\le -\frac{\varepsilon^2 m_s^2}{2} + \frac{|\eta|}{m_s^2} + g_0^2 + \frac{|\eta|}{m} g_0 \\ &\le -\frac{\varepsilon^2 m_s^2}{2} + \frac{3|\eta|}{2m_s^2} + \frac{3}{2} g_0^2,\end{aligned}$$

which together with $V \in \mathcal{L}_\infty$ implies that $\varepsilon m_s \in \mathcal{S}\left(g_0^2 + \frac{\eta^2}{m_s^2}\right)$. Since $|\dot{\theta}| \le (|\varepsilon m_s| + g_0)$ because of the inequality $|g| \le \frac{g_0}{m_s} \le c_1 g_0$, we can conclude that $\dot{\theta} \in \mathcal{S}\left(g_0^2 + \frac{\eta^2}{m_s^2}\right)$. Since g_0 is a constant, we equivalently have $\varepsilon m_s, \dot{\theta} \in \mathcal{S}\left(g_0 + \frac{\eta^2}{m_s^2}\right)$.

Using (3C.25) and the fact that $(\varepsilon m_s^2 - \eta)(\varepsilon + g) \ge 0$, we obtain

$$\begin{aligned}\dot{V} &\le -\left(\varepsilon m_s^2 - \eta\right)\left(\varepsilon + g\right) = -\left|\varepsilon m_s - \frac{\eta}{m_s}\right| |\varepsilon + g| m_s \\ &\le -\left|\left|\varepsilon m_s\right| - \frac{|\eta|}{m_s}\right| |\varepsilon + g| m_s \\ &\le -\left|g_0 - \frac{|\eta|}{m_s}\right| |\varepsilon + g| m_s \qquad \left(\text{because } |\varepsilon + g| = 0 \text{ if } |\varepsilon m_s| \le g_0\right).\end{aligned}$$

Since $\frac{\eta}{m_s} \in \mathcal{L}_\infty$ and $g_0 > \frac{|\eta|}{m_s}$, integrating both sides of the inequality and using the fact that $V \in \mathcal{L}_\infty$, we get $(\varepsilon + g) m_s \in \mathcal{L}_1$, which together with (3.103) and the fact that $\frac{\phi}{m} \in \mathcal{L}_\infty$ implies $\dot{\theta} = \dot{\tilde{\theta}} \in \mathcal{L}_1$.

(iv) $\dot{\theta} \in \mathcal{L}_1$ implies that the $\lim_{t\to\infty} \int_0^t \dot{\theta} d\tau$ exists and therefore θ converges to some constant vector, $\overline{\theta}$.

(v) We express (3.103) in terms of the parameter error as

$$\dot{\tilde{\theta}} = -\Gamma \frac{\phi \phi^T}{m_s^2} \tilde{\theta} + \Gamma \frac{\phi \eta}{m_s^2} + \Gamma \phi g, \tag{3C.28}$$

where g satisfies $|g| \le \frac{g_0}{m_s} \le c_1 g_0$ for some constant $c_1 \ge 0$. The homogeneous part of (3C.28) is e.s. when ϕ / m_s is PE, as established in section 3.4. The exponential convergence of $\tilde{\theta}$ to the given residual set D_d can be established using the same arguments as in the proof of Theorem 3.12.2 (iv). $\square$

3.16 Examples Using the Toolbox

3.16.1 Gradient Algorithm

The Adaptive Control Toolbox uses several MATLAB functions to implement the gradient algorithms introduced in the previous subsections. The MATLAB function **`ucgrad`**[*] can be used to simulate the gradient algorithm with instantaneous cost function, (3.34), as shown in Examples 3.16.1 and 3.16.2.

Example 3.16.1 Consider the signal

$$y = A\cos(\omega t + \varphi), \tag{3C.29}$$

which is broadcast with a known frequency ω but an unknown phase φ and an unknown amplitude A. Assuming that the signal y is available for measurement, our objective is to estimate A and φ using the knowledge of ω and measurements of y. First, we need to transform (3C.29) into the linear parametric form (3.27). Using the identity

$$A\cos(\omega t + \varphi) = A\cos\varphi\cos\omega t - A\sin\varphi\sin\omega t,$$

we obtain

$$y = \theta^{*T}\phi$$

with $\theta^* = [A_1,\ A_2]^T$ and $\phi(t) = [\cos\omega t,\ -\sin\omega t]^T$, where $A_1 = A\cos\varphi$ and $A_2 = A\sin\varphi$. Now, θ^* can be estimated as $\theta = [\hat{A}_1,\ \hat{A}_2]^T$ using (3.34), i.e.,

$$\dot{\theta} = \Gamma\varepsilon\phi\ ,$$
$$\varepsilon = \frac{y - \hat{y}}{m_s^{\,2}} = \frac{y - \theta^T\phi}{m_s^{\,2}}.$$

Since $\phi \in \mathcal{L}_\infty$, we can take the normalizing signal $m_s = 1$. Once θ^* is estimated as $\theta = [\hat{A}_1,\ \hat{A}_2]^T$, we can obtain the estimates of A and φ as

$$\hat{A} = \sqrt{\hat{A}_1^2 + \hat{A}_2^2} \quad \text{and} \quad \hat{\varphi} = \cos^{-1}\left(\frac{\hat{A}_1}{\hat{A}}\right).$$

Now, assume that $A = 3$, $\varphi = 25^\circ = 0.436$ rad for $0 \le t \le 10$ sec, $A = 5$, $\varphi = 35^\circ = 0.611$ rad for $t > 10$ sec, and $\omega = 5$ rad/sec. Let us choose the adaptive gain Γ as the identity matrix. We can use **`ucgrad`** to obtain the estimates of the parameters online as follows:

[*] For details about this command, refer to the toolbox manual—B. Fidan and P. A. Ioannou, *Adaptive Control Toolbox for Use with MATLAB and Simulink: User's Guide* (available from the authors), 2006.

```
t_final = 20;
dt = 0.01;
t = 0:dt:t_final;
N_10 = 10/dt+1;
N_final = t_final/dt+1;
A = [3*ones(1,N_10), 5*ones(1,N_final-N_10)];
phi = [0.436*ones(1,N_10), 0.611*ones(1,N_final-N_10)];
omega = 5;
theta = [0, pi/2];
[nx, x] = ucgrad('init', theta, [], [], []);
for n_step = 1:N_final,
   omegat = omega*t(n_step);
   y(n_step)= A(n_step)*cos(omegat+phi(n_step));
   yt = y(n_step);
   phit = [cos(omegat) -sin(omegat)]';
   dx = ucgrad('state',x,yt,phit,[],0,1,[],0,[],[],[]);
   x = x + dx*dt;
   theta = ucgrad('parameter',x,2,0);
   Ahat(n_step) = sqrt(theta(1)^2+theta(2)^2);
   phihat(n_step) = acos(theta(1)/Ahat(n_step));
   yhat(n_step)= Ahat(n_step)*cos(omegat+phihat(n_step));
end
```

Here, the initial values of the estimates are taken to be $\hat{A}_1(0) = 0$ and $\hat{A}_2(0) = \frac{\pi}{2}$, and the Euler approximation is used for solving the differential equations. The results are shown in Figure 3C.1. More accurate results can be obtained using the Runge–Kutta or the Dormand–Prince method. This can be performed using the Simulink block **`Parameter Estimator`** in the scheme drawn in Figure 3C.2. In this Simulink scheme, the two sine-signal sources at the top and the switch are for the simulation of the broadcasting signal, and the bottom two are for generation of the regressor signal ϕ. The parameters of the **`Parameter Estimator`** can be entered by clicking on it and choosing the continuous-time linear gradient scheme with the model order $n = 2$, the initial parameter $[\theta_1(0),\ \theta_2(0)] = [0,\ 1.571]$, and the adaptive gain $\Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. ■

Example 3.16.2 Consider the plant

$$y = \frac{b_1 s + b_0}{s^2 + 2s + 1} u,$$

where the parameters b_1 and b_0 are unknown. The plant equation can be rewritten in the linear parametric model as

$$y = \theta^{*T} \phi,$$

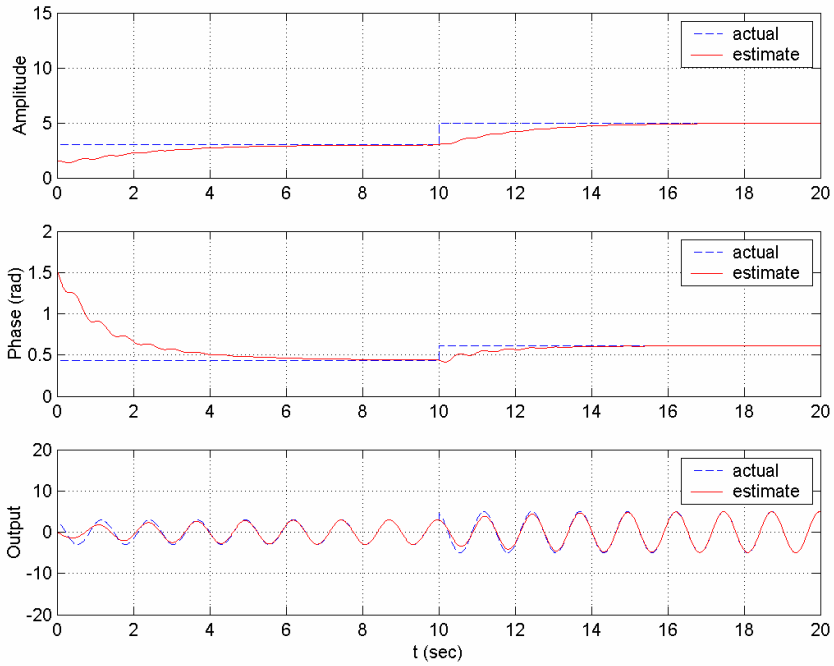

Figure 3C.1 *Actual and estimate values of the amplitude* A, *the phase* φ, *and the output signal* y *in Example* 3.16.1.

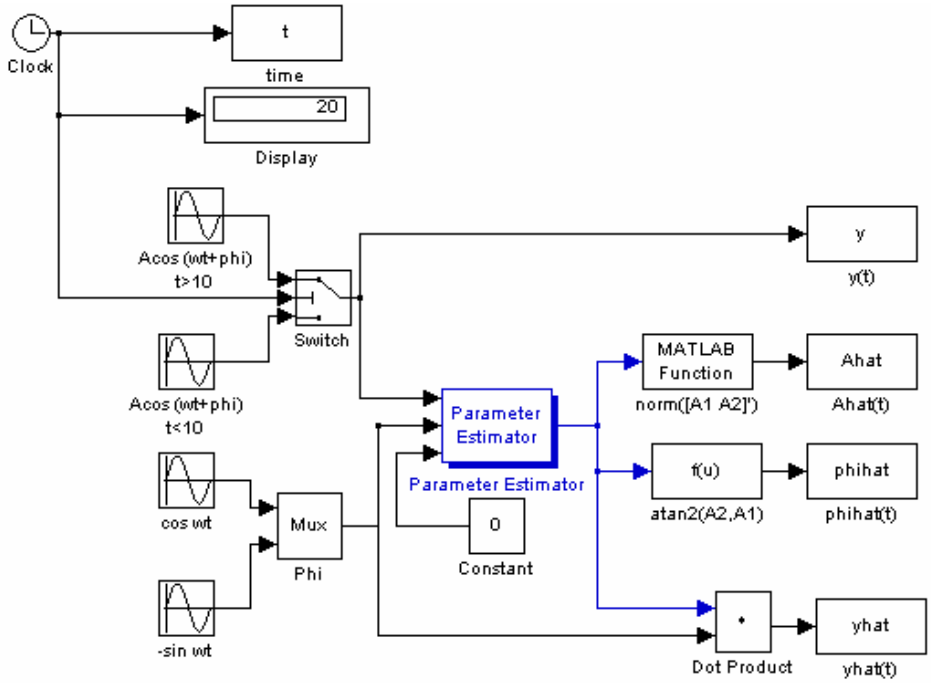

Figure 3C.2 *Simulink scheme for estimating the broadcast signal in Example* 3.16.1.

where $\theta^* = [b_1, b_0]^T$ and $\phi = \left[\frac{s}{s^2+2s+1}u, \frac{1}{s^2+2s+1}u\right]^T$. Let the actual values of the parameters be $b_1 = 2$ and $b_0 = 1.2$, and let the input signal of the system be $u(t) = \sin(t+\frac{\pi}{7}) + 0.5\cos 2t$. Let us use the gradient algorithm

$$\dot{\theta} = \Gamma\varepsilon\phi,$$
$$\varepsilon = \frac{y-\hat{y}}{m_s^2} = \frac{y-\theta^T\phi}{m_s^2}$$

for estimation. Choosing the initial value of the parameter estimate as $\theta(0) = [0,\ 0]^T$, the adaptive gain as $\Gamma = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$, and the normalizing signal as $m_s^2 = 1 + 0.1\phi^T\phi$, we use the function **ucgrad** as follows:

```
t_final = 50;
dt = 0.05;
t = 0:dt:t_final;
N_final = t_final/dt+1;
b = [2 1.2];
a = [1 2 1];
u = sin(t+pi/7) + 0.5*cos(2*t);
theta(:,1) = [0, 0]';
y(1) = 0;
% Estimation parameters:
[nx_est, x_est] = ucgrad('init', theta(:,1), [], [], []);
% Transfer function parameters:
[nx_tf, x_tf] = ufilt('init',b,a,0);
% Linear model parameters:
[nx_phi1, x_phi1] = ufilt('init',[1 0],a,0);
[nx_phi2, x_phi2] = ufilt('init',1,a,0);

for n_step = 2:N_final,
   dx_tf = ufilt('state',x_tf, u(n_step-1), b, a);
   x_tf = x_tf + dx_tf*dt;
   y(n_step)= ufilt('output',x_tf, u(n_step), b, a);
   dx_phi1 = ufilt('state',x_phi1, u(n_step-1), [1 0], a);
   x_phi1 = x_phi1 + dx_phi1*dt;
   phi1 = ufilt('output',x_phi1, u(n_step), [1 0], a);
   dx_phi2 = ufilt('state',x_phi2, u(n_step-1), 1, a);
   x_phi2 = x_phi2 + dx_phi2*dt;
   phi2 = ufilt('output',x_phi2, u(n_step), 1, a);
   phi = [phi1 phi2]';
   dx_est = ucgrad('state',x_est,y(n_step),phi,...
            [],0,10,[],0.1,[],[],[]);
   x_est = x_est + dx_est*dt;
   theta(:,n_step) = ucgrad('parameter',x_est,2,0);
   yhat(n_step)= theta(:,n_step)'*phi;
end
```

The results are plotted in Figure 3C.3. Alternatively, one can use the linear model (3.27) and reduce the dimension of the parameter vector using the knowledge of a_1 and a_0 to obtain

$$z = \theta^{*T}\phi,$$

where

$$z = \frac{s^2 + 2s + 1}{\Lambda(s)} y,$$
$$\theta^* = [b_1,\ b_0]^T,$$
$$\phi = \left[\frac{s}{\Lambda(s)}u,\ \frac{1}{\Lambda(s)}u\right]^T.$$

Choosing the filter polynomial as $\Lambda(s) = (s+2)^2$, the functions **`utf2lm`**, **`lmred`**, and **`ucgrad`** can be used for this purpose as follows:

```
t_final = 50;
dt = 0.05;
t = 0:dt:t_final;
N_final = t_final/dt+1;
b = [2 1.2];
a = [1 2 1];
u = sin(t+pi/7) + 0.5*cos(2*t);
theta(:,1) = [0, 0]';
y(1) = 0;

n = 2; m = 1;
Lambda = [1 4 4];
P = lmred('tf', [n m], [1 0], [2 1], [], []);
% Estimation parameters:
[nx_est, x_est] = ucgrad('init', theta(:,1), [], [], []);
% Transfer function parameters:
[nx_tf, x_tf] = ufilt('init',b,a);
% Linear model parameters:
[nx_lm, x_lm] = utf2lm('init',[n m], Lambda);
[z, phi] = utf2lm('output',x_lm, [u(1) y(1)], [n m],
Lambda);
[zr(1), phir(:,1)] = lmred(z,phi,P);

for n_step = 2:N_final,
   dx_tf = ufilt('state',x_tf, u(n_step-1), b, a);
   x_tf = x_tf + dx_tf*dt;
   y(n_step)= ufilt('output',x_tf, u(n_step), b, a);
   dx_lm = utf2lm('state',x_lm, [u(n_step-1) y(n_step-1)],
[n m], Lambda);
   x_lm = x_lm + dx_lm*dt;
   [z, phi] = utf2lm('output',x_lm,[u(n_step) y(n_step)],
[n m], Lambda);
   [zr(n_step), phir(:,n_step)] = lmred(z,phi,P);
   dx_est =ucgrad('state',x_est,zr(n_step),phir(:,n_step),...
      [],0,10,[],0.1,[],[],[]);
   x_est = x_est + dx_est*dt;
   theta(:,n_step) = ucgrad('parameter',x_est,2,0);
   zhat(n_step)= theta(:,n_step)'*phir(:,n_step);
end
```

The results for the alternative method are shown in Figure 3C.4. The same results could be obtained using the Simulink scheme depicted in Figure 3C.5, with appropriate selections in the menus of the blocks **`Parametric Model`***, **`Parameter Estimator`***, and **`Model Reduction`***. ■

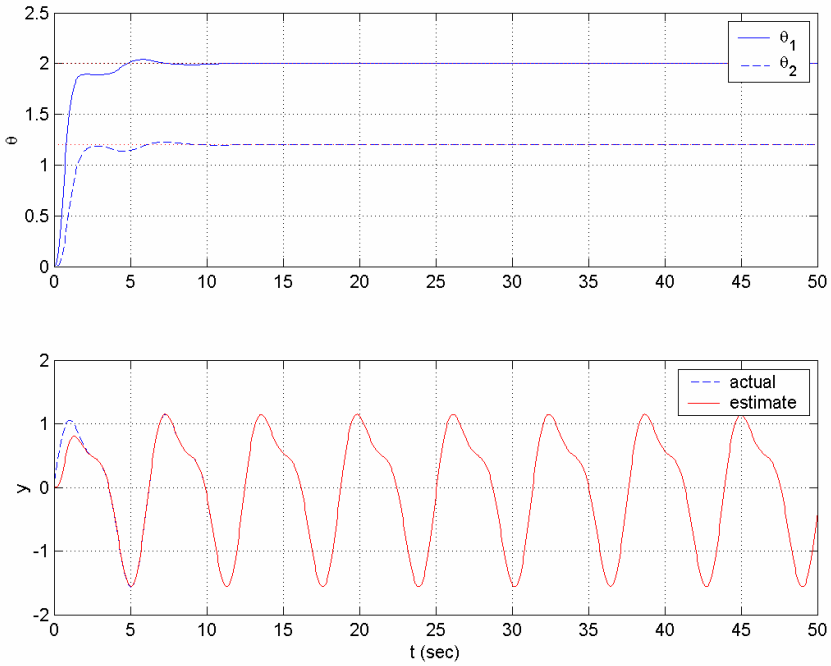

Figure 3C.3 *Simulation results for Example* 3.16.2.

* For details about this block, refer to the toolbox manual.

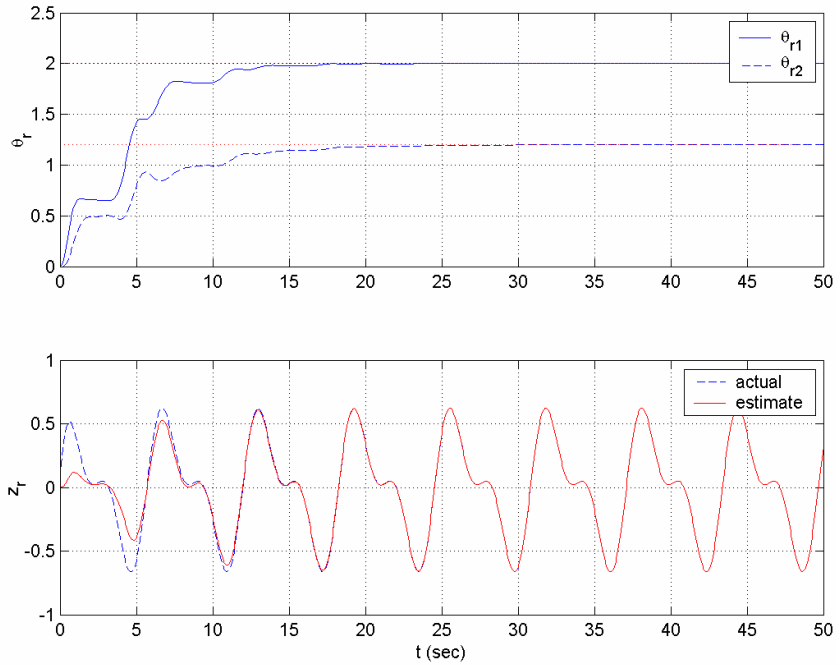

Figure 3C.4 *Simulation results for Example* 3.16.2 *with alternative parameterization.*

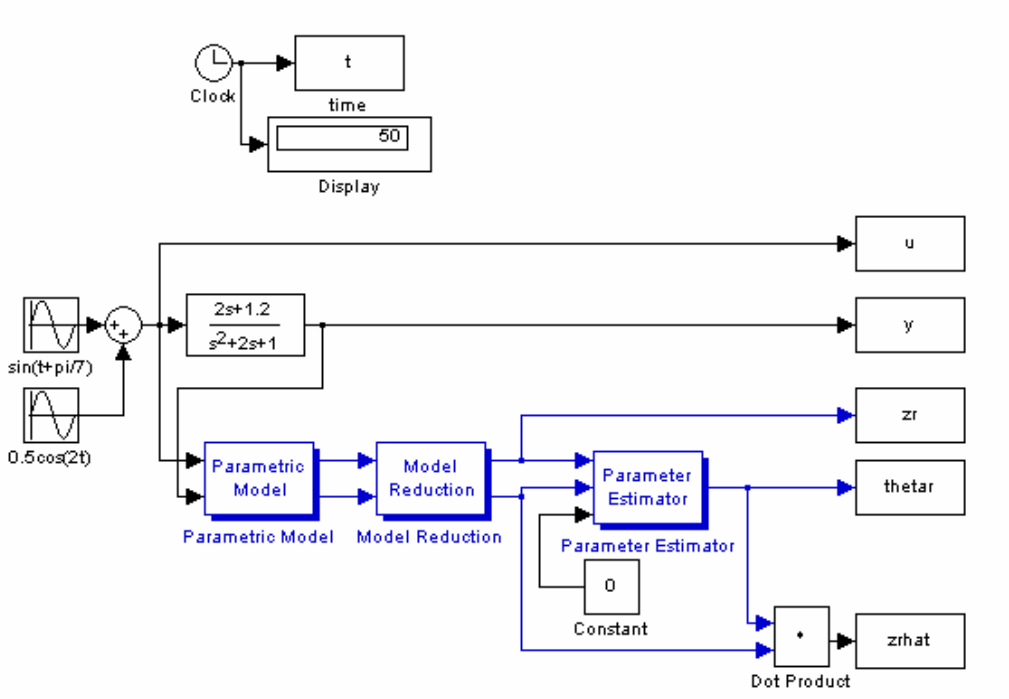

Figure 3C.5 *Simulink scheme for parameter estimation in Example* 3.16.2.

Example 3.16.3 Consider the mass–spring–dashpot system

$$M\ddot{x} = u - kx - f\dot{x}$$

of Example 2.1, where $M, f,$ and k denote the mass of the system, the damping coefficient, and the spring constant, respectively. Assume that $f = 0.1$ kg/sec, $k = 5$ kg / sec^2, $M = 20$ kg and $u = 0.1\cos\frac{\pi}{7}t + 0.05\sin\frac{\pi}{5}t$ kg m/sec^2. Let M be known, and let f and k be the unknown parameters to be estimated online. The system equation can be rewritten in the transfer function form

$$x = \frac{b_0}{s^2 + a_1 s + a_0}u,$$

where

$$b_0 = \frac{1}{M}, \quad a_1 = \frac{f}{m}, \quad a_0 = \frac{k}{M}.$$

Since M is known, b_0 is also known, and the following reduced-order linear parametric model can be used for parameter identification purposes:

$$z = \theta^{*T}\phi,$$

where

$$z = \frac{s^2}{\Lambda(s)}y - \frac{b_0}{\Lambda(s)}u,$$
$$\theta^* = [a_1,\ a_0]^T,$$
$$\phi = \left[-\frac{s}{\Lambda(s)}y,\ -\frac{1}{\Lambda(s)}y\right]^T.$$

Let us choose $\Lambda(s) = s^2 + 2s + 1,$ and let us use the gradient algorithm

$$\dot{\theta} = \Gamma\varepsilon\phi,$$
$$\varepsilon = \frac{z - \hat{z}}{m_s^2} = \frac{z - \theta^T\phi}{m_s^2}$$

with the initial parameter estimate $\theta(0) = [0,\ 0]^T$, the adaptive gain $\Gamma = 25I$, and the normalizing signal $m_s^2 = 1 + 0.1\phi^T\phi$. For implementation and simulation, we use the Simulink scheme shown in Figure 3C.6 with appropriate selections in the menus of the blocks **`Parametric Model`***, **`Parameter Estimator`***, and **`Model Reduction*`**. The results plotted in Figure 3C.7 indicate that the parameter estimates converge to their actual values, i.e., parameter identification is successful. ■

* For details about this block, refer to the toolbox manual.

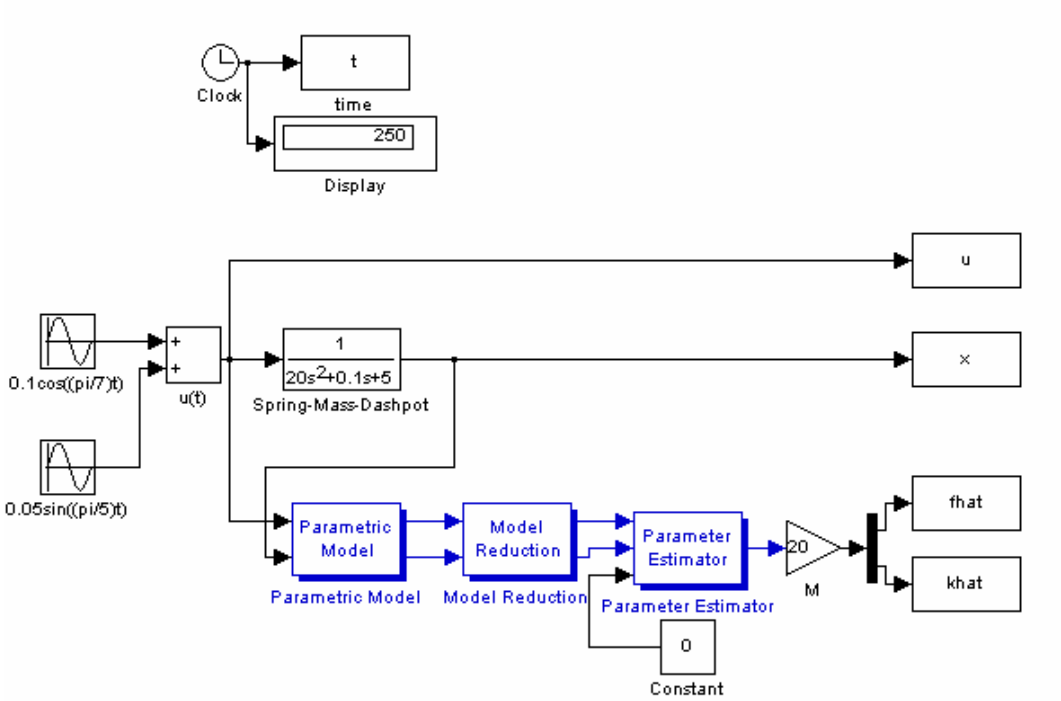

Figure 3C.6 *Simulink scheme for estimating the coefficients of the mass–spring–dashpot system in Example* 3.16.3.

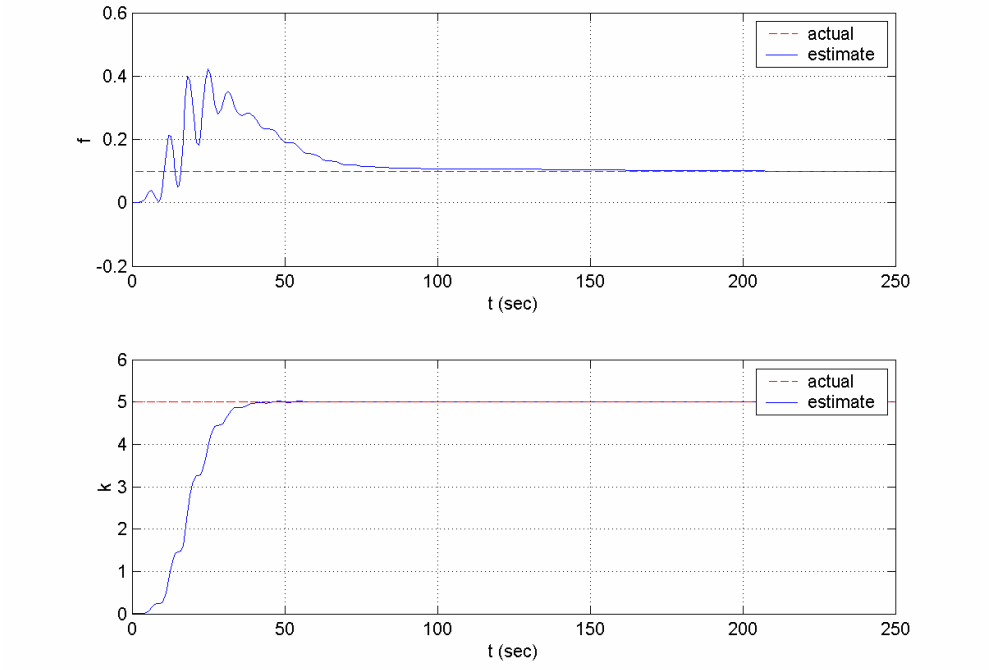

Figure 3C.7 *Simulation results for Example* 3.16.3.

The Simulink block **`Parameter Estimator`** can be used to simulate the gradient algorithm with integral cost function as well, as demonstrated in the examples below.

Example 3.16.4 Consider the broadcast signal of Example 3.16.1 keeping the parameter values the same. Let us perform the parameter estimation using the gradient algorithm with integral cost function. The Simulink block **`Parameter Estimator`**[*] may be used to perform the simulations as before. Entering the model order, the forgetting factor $\beta = 0.1$, the initial parameter vector, and the gain of Example 3.16.1, the results shown in Figure 3C.8 are obtained. ■

Example 3.16.5 Consider the mass–spring–dashpot system of Example 3.16.3 keeping the parameter values the same. We will perform the parameter estimation using the gradient algorithm with integral cost function. Entering the model order, the initial parameter vector, and the gain in the Simulink block **`Parameter Estimator`**[*] as in Example 3.16.3 and choosing the forgetting factor $\beta = 0.1$, the results shown in Figure 3C.9 are obtained. ■

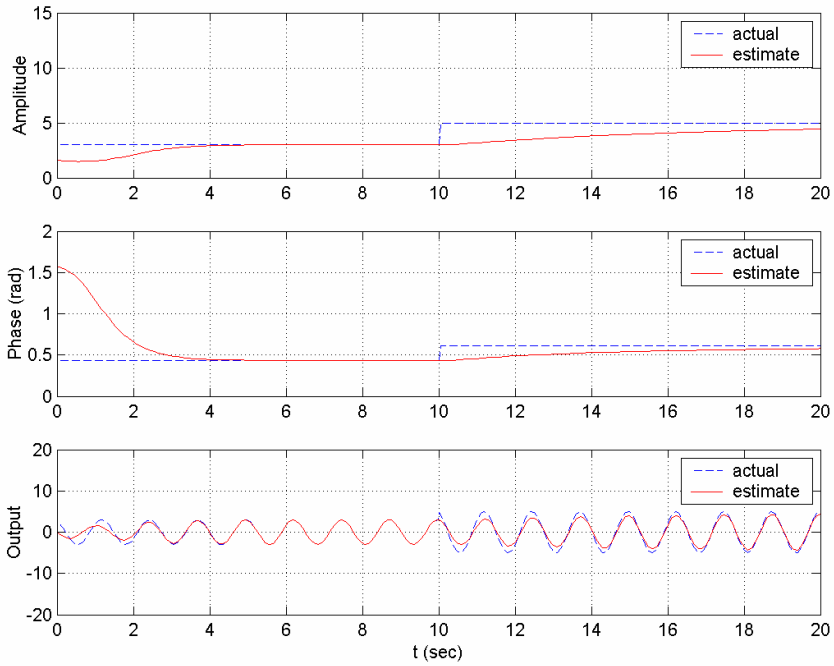

Figure 3C.8 *Actual and estimate values of the amplitude A, the phase φ, and the output signal y in Example* 3.16.4.

[*] For details about this block, refer to the toolbox manual.

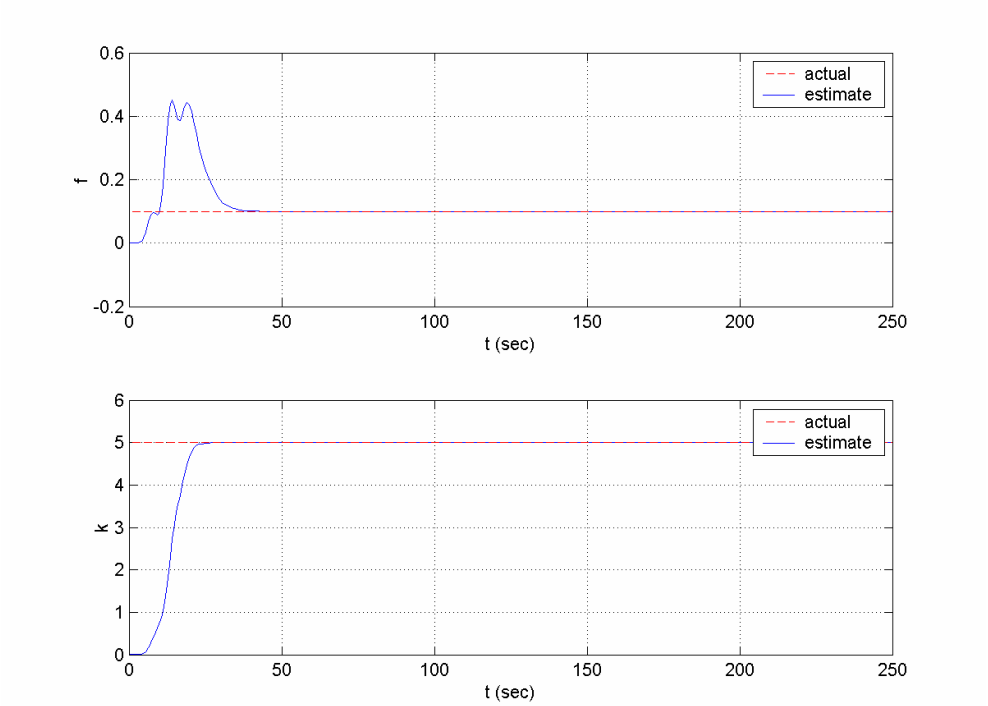

Figure 3C.9 *Simulation results for Example* 3.16.5.

3.16.2 LS Algorithm

The MATLAB functions **`ucrls`*** and **`urlsarg`*** or the Simulink block **`Parameter Estimator`** can be used to simulate any of the four LS algorithms described by (3.45)–(3.48), as demonstrated in Examples 3.16.6 and 3.16.7.

Example 3.16.6 Consider Example 3.16.1 with the same parameter values. Let us perform the same parameter estimation task using a recursive LS algorithm with forgetting factor, pure LS algorithm, and pure LS algorithm with covariance resetting. For the recursive LS algorithm with forgetting factor $\beta = 1$, we use **`ucrls`** and **`urlsarg`** to obtain the estimates of the parameters online as follows:

```
t_final = 20;
dt = 0.01;
t = 0:dt:t_final;
N_10 = 10/dt+1;
N_final = t_final/dt+1;
A = [3*ones(1,N_10), 5*ones(1,N_final-N_10)];
phi = [0.436*ones(1,N_10), 0.611*ones(1,N_final-N_10)];
omega = 5;
theta = [0, pi/2];
rho0 = 1;
P0 = rho0*eye(2);
ArgLS = urlsarg('constant forgetting', 1);
[nx, x] = ucrls('init', theta, P0, ArgLS);
for n_step = 1:N_final,
   omegat = omega*t(n_step);
   y(n_step)= A(n_step)*cos(omegat+phi(n_step));
```

* For details about this command, refer to the toolbox manual.

```
    yt = y(n_step);
    phit = [cos(omegat) -sin(omegat)]';
    dx = ucrls('state',x,yt,phit,0,ArgLS,[],[]);
    x = x + dx*dt;
    x = ucrls('update',x,[],ArgLS);
    theta = ucrls('parameter',x,2);
    Ahat(n_step) = sqrt(theta(1)^2+theta(2)^2);
    phihat(n_step) = acos(theta(1)/Ahat(n_step));
    yhat(n_step)= Ahat(n_step)*cos(omegat+phihat(n_step));
end
```

Note that the type and parameters of the LS algorithm are defined on the line `ArgLS = urlsarg('constant forgetting', 1);`. This line should be changed to `ArgLS = urlsarg('pure');` to apply a pure LS algorithm and to `ArgLS = urlsarg('size', 1, 0.3);` to apply a pure LS algorithm with covariance resetting ($\rho_0 = 1, \rho_1 = 0.3$). For the three methods, the initial covariance matrix is chosen as $P_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, the initial values of the estimated parameters are taken to be $\hat{A}_1(0) = 0$ and $\hat{A}_2(0) = \frac{\pi}{2}$, and the Euler approximation is used for differential equations as in Example 3.16.1. The results are plotted in Figures 3C.10, 3C.11, and 3C.12. As in the gradient algorithm case (Example 3.16.1), the Simulink block **`Parameter Estimator`** can also be used for simulation. We can use the scheme of Figure 3C.2 (described in Example 3.16.1) here, changing the entries of the **`Parameter Estimator`**[*] block appropriately. ■

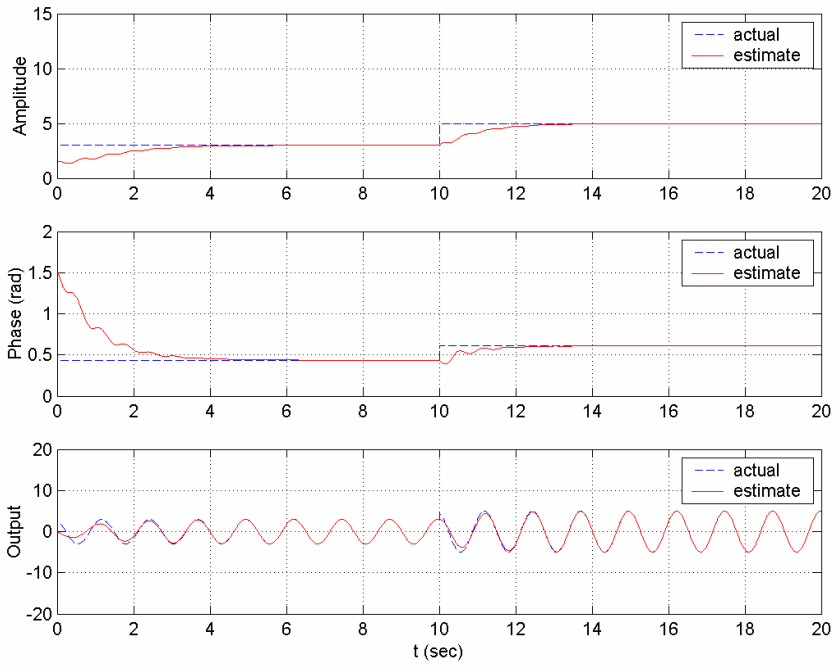

Figure 3C.10 *Actual and estimate values of the amplitude* A*, the phase* φ*, and the output signal* y *in Example* 3.16.6 *using a recursive LS algorithm with forgetting factor.*

[*] For details about this block, refer to the toolbox manual.

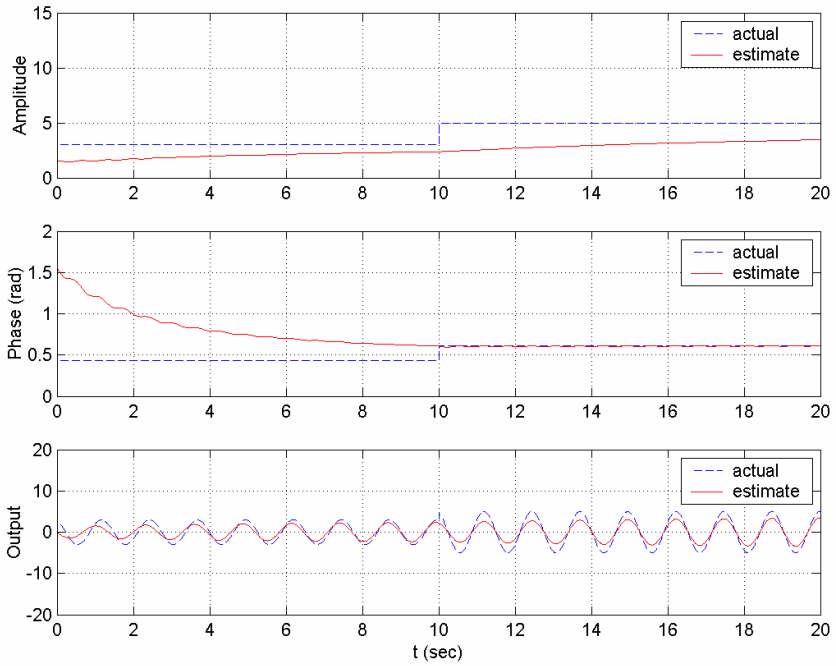

Figure 3C.11 *Actual and estimate values of the amplitude A, the phase φ, and the output signal y in Example* 3.16.6 *using a pure LS algorithm.*

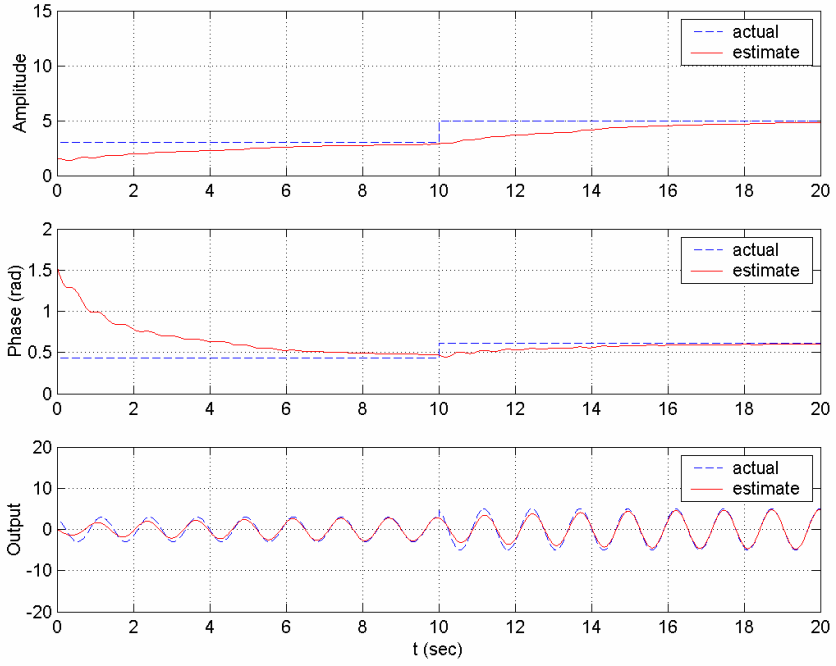

Figure 3C.12 *Actual and estimate values of the amplitude A, the phase φ, and the output signal y in Example* 3.16.6 *using a pure LS algorithm with covariance resetting.*

Example 3.16.7 Consider the mass–spring–dashpot system of Example 3.16.3 with the same parameter values. Let us use the recursive LS algorithm with constant forgetting factor for estimating the parameters. Let us choose the algorithm parameters as $\beta = 0.5$, $P_0 = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$, and $\theta(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. We use the Simulink scheme of Figure 3C.5, changing the entries of the block **`Parameter Estimator`*** appropriately while keeping **`Model Reduction`*** the same. The results are shown in Figure 3C.13.

Trying different forgetting factors and different initial covariance values, one can easily see that increasing the initial covariance values and the forgetting factor increases the speed of convergence at the expense of some transient. ■

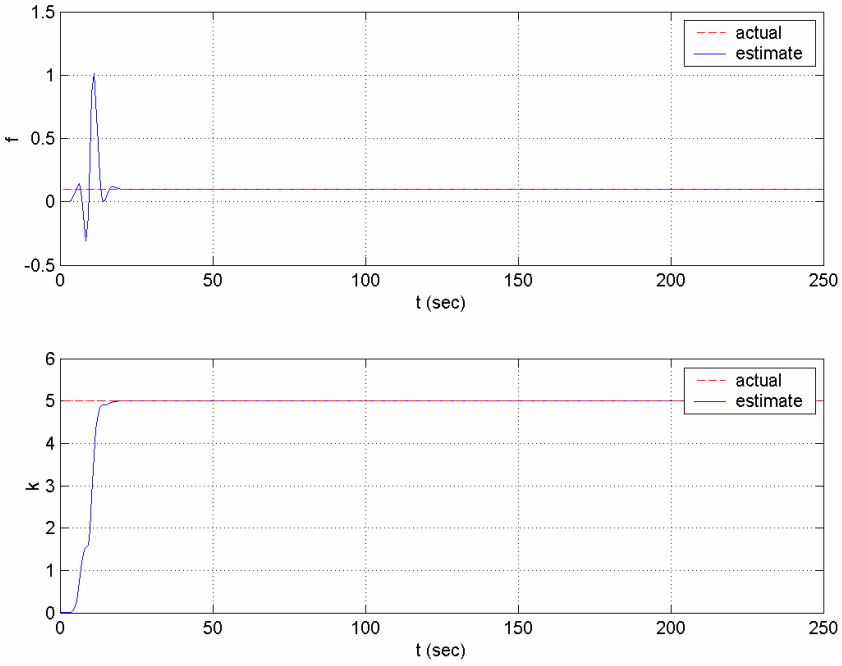

Figure 3C.13 *Simulation results for Example* 3.16.7.

3.16.3 Parameter Projection

The MATLAB function **`uparproj`*** can be used with **`ucgrad`**, or the Simulink block **`Parameter Estimator`**, and appropriate projection parameters to simulate the gradient algorithm with parameter projection, (3.58), as shown in Examples 3.16.8 and 3.16.9. The gradient algorithm with projection given by (3.58) may be simulated using the Simulink block **`Parameter Estimator`**, with appropriate projection parameters as demonstrated in Example 3.16.8.

* For details, refer to the toolbox manual.

Example 3.16.8 Consider the plant

$$y = \frac{b_1 s + b_0}{s^2 + a_1 s + a_0} u \,,$$

where the unknown coefficients b_1, b_0, a_1, and a_0 are to be estimated. The plant is re-written in the form of the linear parametric model as

$$z = \theta^{*T} \phi \,,$$

where $\theta^* = [b_1,\ b_0,\ a_1,\ a_0]^T$, $\phi = [\frac{s}{\Lambda(s)}u,\ \frac{1}{\Lambda(s)}u,\ -\frac{s}{\Lambda(s)}y,\ -\frac{1}{\Lambda(s)}y]^T$, $z = \frac{s^2}{\Lambda(s)}y$, and $\Lambda(s) = (s+2)^2$. For simulation purposes let $a_1 = 0.6$, $a_0 = 8.5$, $b_1 = 2.6$, and $b_0 = 2.0$, and the input signal $u(t) = \sin(t + \frac{\pi}{7})$, and use the gradient algorithm with instantaneous cost function described in section 3.6 with the initial parameter estimate vector $\theta(0) = [2.75,\ 3,\ 1.25,\ 6]^T$, the adaptive matrix $\Gamma = 100\ \mathrm{I}_4$, and the normalizing signal $m_s^2 = 1 + 0.1\phi^T\phi$. As seen in the simulation results shown in Figure 3C.14, the parameter estimates do not converge to the actual values, and the estimation errors are significant.

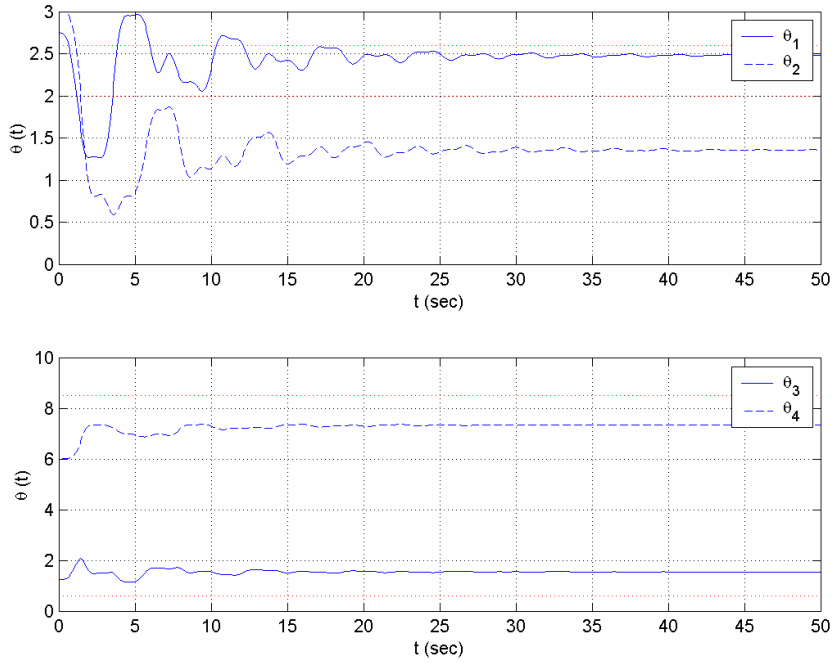

Figure 3C.14 *Simulation results for Example* 3.16.8 *using the gradient algorithm without projection.*

Let us now assume that we know that the parameter vector θ lies in the compact set $S^0 = \{\theta \in \Re^4 \,|\, 2.5 \le \theta_1 \le 3,\ 1.5 \le \theta_2 \le 4.5,\ 0.5 \le \theta_3 \le 2,\ 3 \le \theta_4 \le 9\}$. Based on this assumption, we can incorporate **`uparproj`** with **`ucgrad`** for parameter estimation as follows:

```
t_final = 50;
dt = 0.05;
t = 0:dt:t_final;
N_final = t_final/dt+1;
b = [2.6 2];
a = [1 0.6 8.5];
u = sin(t+pi/7);
theta(:,1) = [2.75, 3, 1.25, 6]';
y(1) = 0;

n = 2; m = 1;
Lambda = [1 4 4];
PJ = uparproj('hyperplane', [1 2 3 4],[2.5 1.5 0.5 3], [3
4.5 2 9]);
% Estimation parameters:
[nx_est, x_est] = ucgrad('init', theta(:,1), [], [], []);
% Transfer function parameters:
[nx_tf, x_tf] = ufilt('init',b,a);
% Linear model parameters:
[nx_lm, x_lm] = utf2lm('init',[n m], Lambda);
[z(1), phi(:,1)] = utf2lm('output',x_lm, [u(1) y(1)], [n
m], Lambda);
for n_step = 2:N_final,
   dx_tf = ufilt('state',x_tf, u(n_step-1), b, a);
   x_tf = x_tf + dx_tf*dt;
   y(n_step)= ufilt('output',x_tf, u(n_step), b, a);
   dx_lm = utf2lm('state',x_lm, [u(n_step-1) y(n_step-1)],
[n m], Lambda);
   x_lm = x_lm + dx_lm*dt;
   [z(n_step), phi(:,n_step)] =
utf2lm('output',x_lm,[u(n_step) y(n_step)], [n m], Lambda);
   dx_est =
ucgrad('state',x_est,z(n_step),phi(:,n_step),...
      [],0,10,[],0.1,[],[],[],[],PJ);
   x_est = x_est + dx_est*dt;
   x_est = ucgrad('update',x_est,4,0,PJ);
   theta(:,n_step) = ucgrad('parameter',x_est,4,0);
   zhat(n_step)= theta(:,n_step)'*phi(:,n_step);
end
```

As shown in Figure 3C.15, employing parameter projection, parameter estimation gives more successful results. This parameter estimation task could also be implemented using the Simulink block **Parameter Estimator*** with appropriate menu selections. ■

* For details about this block, refer to the toolbox manual.

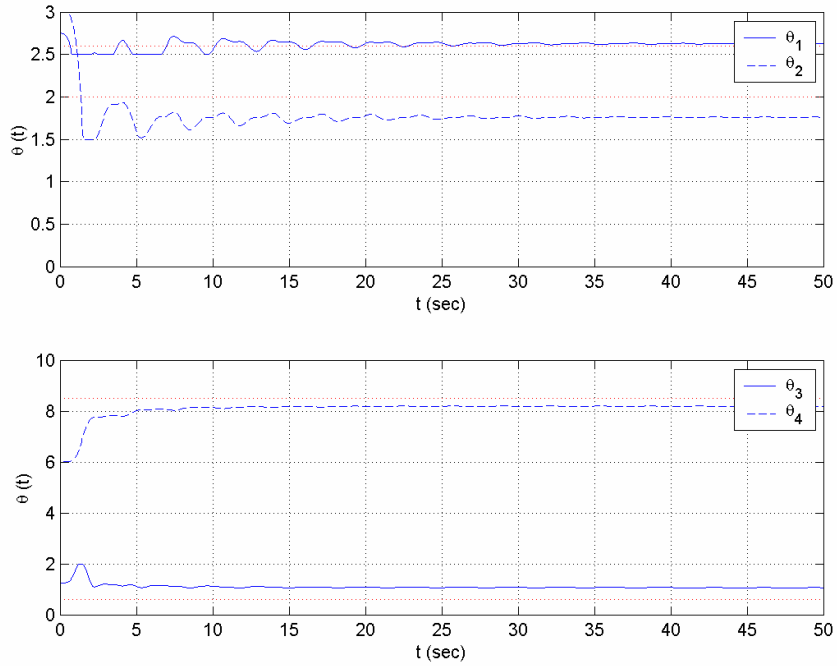

Figure 3C.15 *Simulation results for Example* 3.16.8 *using the gradient algorithm with projection.*

The MATLAB function **`uparproj`** incorporated with **`ucrls`** and **`urlsarg`**, or the Simulink block **`Parameter Estimator`**, with appropriate projection parameters, can be used to simulate (3.59), as shown in Example 3.16.9.

Example 3.16.9 Let us use the recursive LS algorithm with forgetting factor instead of the gradient algorithm for Example 3.16.8. Let the initial covariance matrix be $P_0 = 100\ \mathrm{I}_4$ and the forgetting factor $\beta = 0.7$. We use Simulink by making appropriate changes in the entries of the block `Parameter Estimator`, or by modifying the MATLAB code given in Example 3.16.8 using the following lines at appropriate places:

```
ArgLS = urlsarg('constant forgetting', 0.7);
[nx_est, x_est] = ucrls('init', theta(:,1), 100*eye(4),
ArgLS);

dx_est =
ucrls('state',x_est,z(n_step),phi(:,n_step),0,ArgLS,[],PJ);

x_est = ucrls('update',x_est,PJ,ArgLS);

theta(:,n_step) = ucrls('parameter',x_est,4);
```

Parameter estimation is much better in terms of both the estimation error and the convergence rate, as shown in Figure 3C.16. ■

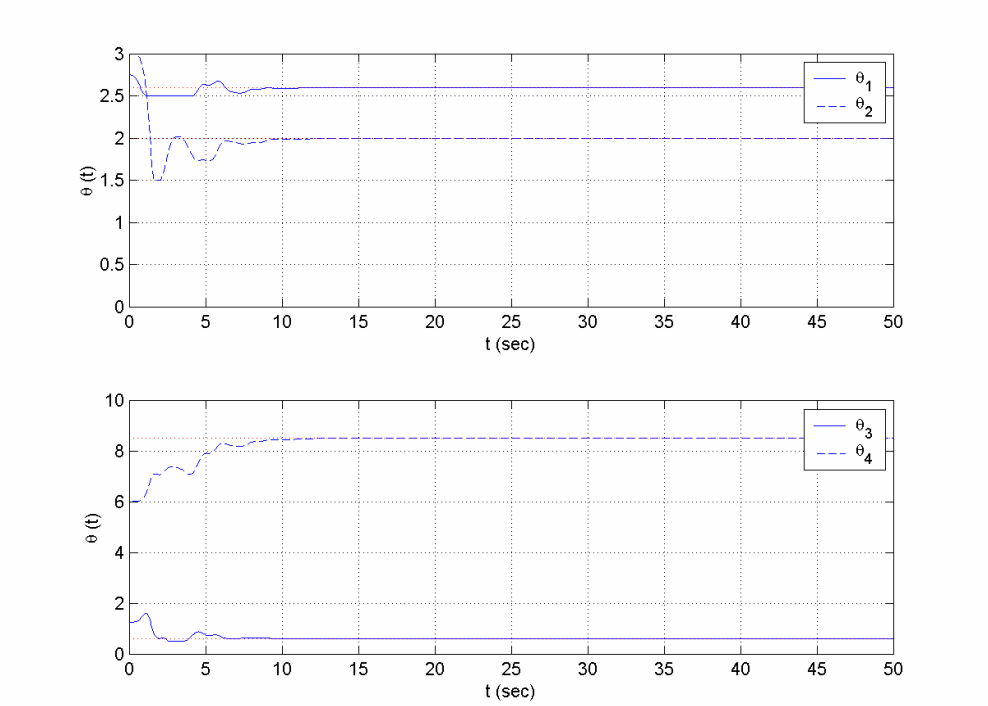

Figure 3C.16 *Simulation results for Example* 3.16.9.

3.16.4 Robust Parameter Identification

The MATLAB function **`urobust`** can be incorporated with **`ucgrad`**, or, as before, the Simulink block **`Parameter Estimator`** with appropriate parameters can be used to simulate the robust gradient laws with leakage based on the instantaneous cost function, as shown in Example 3.16.10. For integral cost function, the robust gradient law with leakage can be simulated using the Simulink block **`Parameter Estimator`** with appropriate parameters. For the robust LS laws with leakage, one may either incorporate the MATLAB function **`urobust`** with **`ucrls`** and **`urlsarg`**, or use the Simulink block **`Parameter Estimator`** with appropriate parameters, as demonstrated in Example 3.16.10. The robust adaptive laws with dead zone can be simulated using the MATLAB commands or the Simulink blocks of the toolbox.

Example 3.16.10 Consider the plant

$$y = \theta^* u + d\,,$$

where θ^* is the unknown scalar parameter of the plant, d is some unknown bounded disturbance, and u and y are input and output signals of the system available for measurement. Let $\theta^* = 2$, $u = (1+t)^{-1/2}$, and $d = (1+t)^{-1/4}\left(\frac{5}{4} - 2(1+t)^{-1/4}\right)$. If we neglect $d(t)$, we can design a gradient adaptive law to estimate the value of θ^* as follows:

$$\dot{\theta} = \gamma \varepsilon_1 u, \qquad \varepsilon_1 = y - \theta u,$$

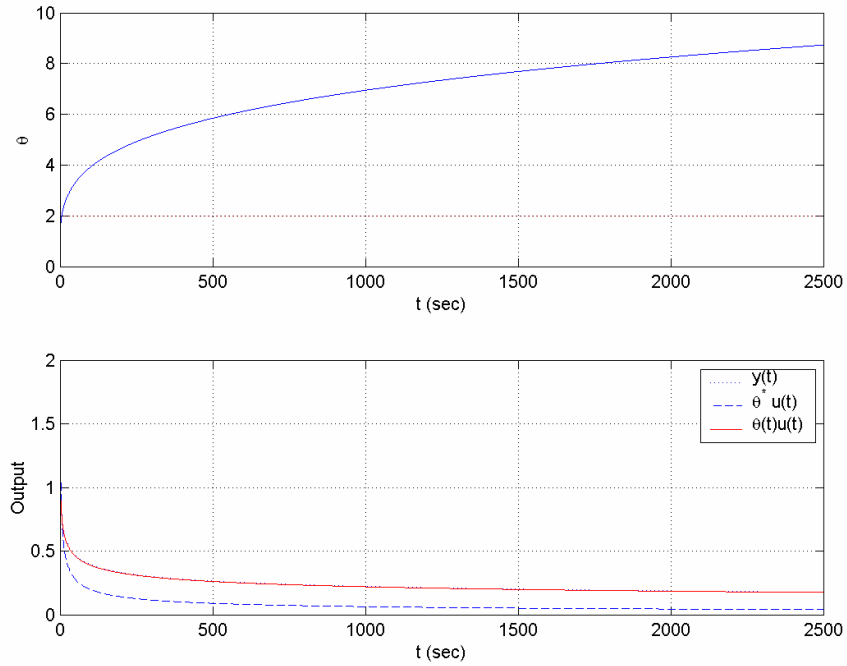

Figure 3C.17 *Simulation results for Example* 3.16.10 *using the pure gradient algorithm with instantaneous cost function (with no robustness modification).*

where $\gamma > 0$ and $\theta(t)$ is the online estimate of θ^*. If we use this law directly in the existence of $d(t)$, we get the result shown in Figure 3C.17 for $\gamma = 20$.

As can be seen in Figure 3C.17, although the estimation of the output signal is acceptable in terms of convergence, estimate of the parameter θ^* diverges. To prevent the divergence, we can apply robustness modifications. If we have no idea about the range of θ^*, we can apply the fixed σ-modification. Simulation for $\sigma = 0.05$ can be performed using the following code:

```
t_final = 250;
dt = 0.1;
t = 0:dt:t_final;
N_final = t_final/dt+1;
theta_s = 2;
u = (1+t).^(-0.5);
d = ((1+t).^(-0.25)).*(1.25-2*((1+t).^(-0.25)));
theta_0 = 1.5;
Arg = urobust('csigma', 0.005);
[nx, x] = ucgrad('init', theta_0, [], [], []);
for n_step = 1:N_final,
   y(n_step)= theta_s*u(n_step)+d(n_step);
   dx = ucgrad('state',x,y(n_step),u(n_step),[],0,20,[],0,[],[],Arg);
   x = x + dx*dt;
   theta(n_step) = ucgrad('parameter',x,1,0);
   yhat(n_step)= theta(n_step)*u(n_step);
end
```

The result is plotted in Figure 3C.18.

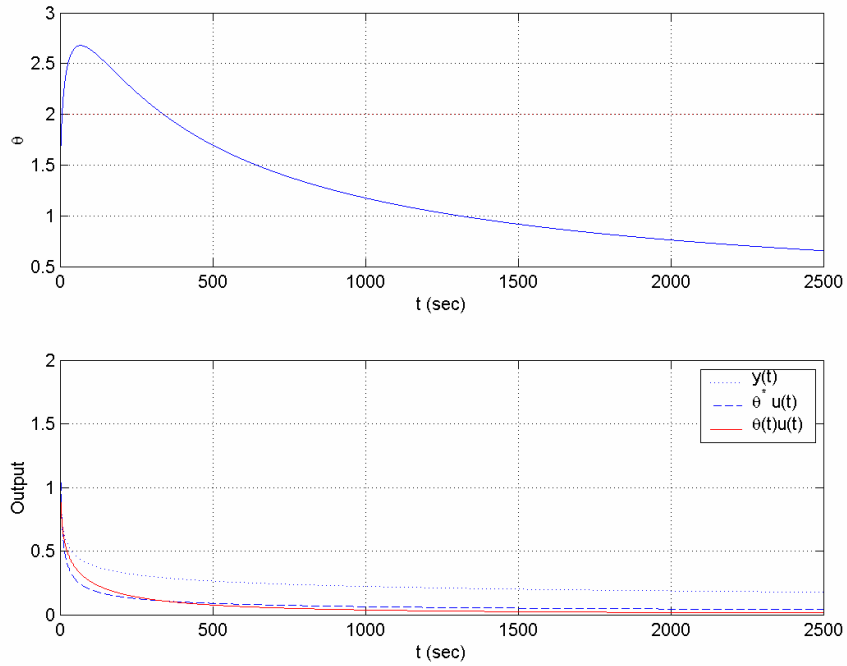

Figure 3C.18 *Simulation results for Example* 3.16.10 *using the gradient algorithm with instantaneous cost function with fixed* σ *-modification.*

Furthermore, if we know that the value of θ^* is about 1.5, with an error limit of ± 1.0, we can apply the switching σ-modification with $\theta_0 = 1.5$, $\sigma_0 = 0.2$, $M_0 = 1.0$. We can perform the corresponding simulation by changing the line `Arg = urobust('csigma', 0.005);` of the code above to `Arg = urobust('csigma', 0.2,theta_0,1.0);`. The result is plotted in Figure 3C.19.

The benefit of using robustness modifications is obvious from Figures 3C.17–3C.19. It is also worth noting that choosing the $s_\ell(t)$ larger makes the adaptive law more conservative; i.e., for large values of σ, σ_0, or ν_0, the effects of the pure adaptive laws are lost. This can be seen by trying different values for σ and σ_0 in this example.

The simulations above can be performed using the Simulink block **`Parameter Estimator*`** as well. Robustness modification parameters can be entered after selecting a robust modification method from the menu. ■

* For details about this block, refer to the toolbox manual.

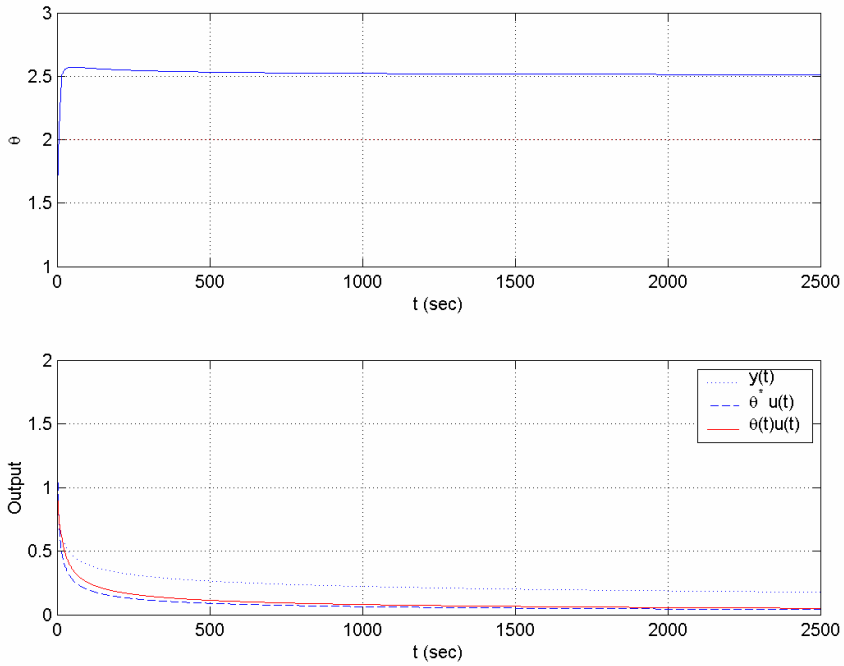

Figure 3C.19 *Simulation results for Example* 3.16.10 *using the gradient algorithm with instantaneous cost function with switching* σ *-modification.*

3.16.5 Gradient Algorithms Based on a Bilinear Model

The MATLAB function **`ucgrad`** or the Simulink block **`Parameter Estimator`** can be used to implement the gradient algorithms based on B-SPM and B-DPM with known sign of ρ^*. In addition to the known sign of ρ^*, if a lower bound for $|\rho^*|$ is known as well, one may use the MATLAB function **`ucgradbk`** or modify the selections in the Simulink block **`Parameter Estimator.`** Examples 3.16.11 and 3.16.12 demonstrate the implementation of the bilinear model-based gradient algorithms using the toolbox.

Example 3.16.11 Consider the first-order differential equation

$$a_1\dot{y} + a_0 y = u = \sin\left(0.5t + \frac{\pi}{5}\right),$$

where the coefficients a_1 and a_0 are unknown and need to be estimated using the histories of u and y. We can rewrite this equation in the bilinear parametric form, after filtering both sides with the Hurwitz polynomial $\Lambda(s) = s+1$, as

$$z = \rho^*\left(\theta^{*T}\phi + z_0\right),$$

where $z = \frac{s}{\Lambda(s)}y$, $\rho^* = \frac{1}{a_1}$, $\theta^* = a_0$, $\phi = -\frac{1}{\Lambda(s)}y$, and $z_0 = \frac{1}{\Lambda(s)}u$.

In order to estimate $\rho^* = \frac{1}{a_1}$ and $\theta^* = a_0$, we can use the following estimation algorithm:

$$\dot{\theta} = \Gamma \varepsilon \phi \operatorname{sgn}\left(\rho^*\right),$$
$$\dot{\rho} = \gamma \varepsilon \xi,$$
$$\varepsilon = \frac{z - \rho\xi}{1 + C_s \phi^2}, \qquad \xi = \theta\phi + z_0.$$

Let us assume that the actual values of the parameters are $a_1 = -2$ and $a_0 = -1$. Assume also that the sign of a_1 (hence $\operatorname{sgn}(\rho^*)$) is known. Choosing the design coefficients as $C_s = 0.1$, $\Gamma = 1$, and $\gamma = 1$, and the initial parameters as $\theta(0) = 0$ and $\rho(0) = -0.000001$, we can simulate the system and use **`ucgrad`** to obtain the estimates of the parameters online as follows:

```
t_final = 250;
dt = 0.5;
t = 0:dt:t_final;
N_final = t_final/dt+1;
N_10 = 10/dt+1;
a1 = -2;
a0 = -1;
num_dif = 1/a1;
den_dif = [a1 a0]./a1;
u = sin(0.5*t+pi/5);
theta_s = a0;
rho_s = 1/a1;
theta(1) = 0;
rho(1) = -0.000001;
z(1) = 0;
y(1) = 0;
zhat(1) = 0;
Lambda = [1 1];
gamma = 1; Gamma = 1; GammaSignRho = -Gamma;


% Estimation parameters:
[nx_est, x_est] = ucgrad('init', theta(1), rho(1), [], []);
% Transfer function parameters:
[nx_tf, x_tf] = ufilt('init',num_dif,den_dif,0);
% Linear model parameters:
[nx_z_phi, x_z_phi] = ufilt('init',1,Lambda,1:-1:0);
[nx_z0, x_z0] = ufilt('init',1,Lambda,0);

for n_step = 2:N_final,
   dx_tf = ufilt('state',x_tf, u(n_step-1), num_dif, den_dif);
   x_tf = x_tf + dx_tf*dt;
   y(n_step)= ufilt('output',x_tf, u(n_step), num_dif, den_dif);
   dx_z_phi = ufilt('state',x_z_phi, y(n_step-1), 1, Lambda);
   x_z_phi = x_z_phi + dx_z_phi(:)*dt;
   z(n_step) = ufilt('output',x_z_phi, y(n_step), 1,
   Lambda, 1);
   phi = -ufilt('output',x_z_phi, y(n_step), 1, Lambda, 0);
   dx_z0 = ufilt('state',x_z0, u(n_step-1), 1, Lambda);
   x_z0 = x_z0 + dx_z0*dt;
   z0 = ufilt('output',x_z0, u(n_step), 1, Lambda);
```

```
    dx_est =
    ucgrad('state',x_est,z(n_step),phi,z0,0,GammaSignRho,...
                        gamma,0.1,[],[],[],[]);
    x_est = x_est + dx_est*dt;
    [theta(n_step), rho(n_step)] =
    ucgrad('parameter',x_est,1,1);
    zhat(n_step)= rho(n_step)*(theta(n_step)*phi + z0);
end
```

The results are plotted in Figure 3C.20. If, furthermore, it is known that $\rho^* \leq -0.1$, we can get a better result by setting $\rho(0) = -0.1$ and using

```
rho(1) = -0.1;
rhomin = -0.1;

gamma = 1; Gamma =1;

[nx_est, x_est] = ucgradbk('init', theta(1), rho(1),
rhomin);

dx_est = ...
ucgradbk('state',x_est,z(n_step),phi,z0,0,Gamma,gamma,0.1,rhomin);
x_est = x_est + dx_est*dt;
x_est = ucgradbk('update',x_est,rhomin);
[theta(n_step), rho(n_step)] = ucgradbk('parameter',x_est,1);
```

instead of the corresponding lines in the code before. The results are shown in Figure 3C.21. These two simulations could also be performed using the Simulink block **Parameter Estimator***. ■

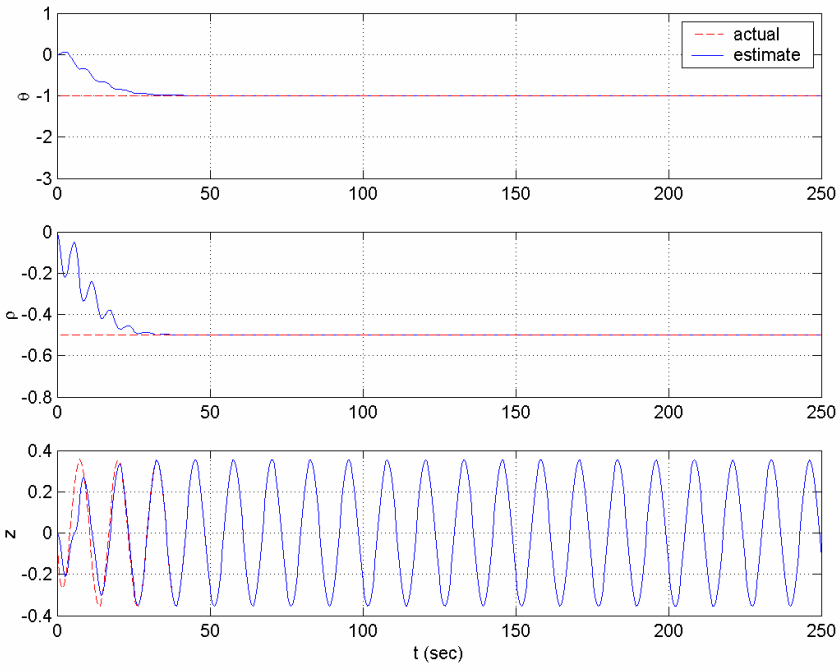

Figure 3C.20 *Simulation results for Example* 3.16.11 *assuming only* $\text{sgn}(\rho^*)$ *is known.*

* For details about this block, refer to the toolbox manual.

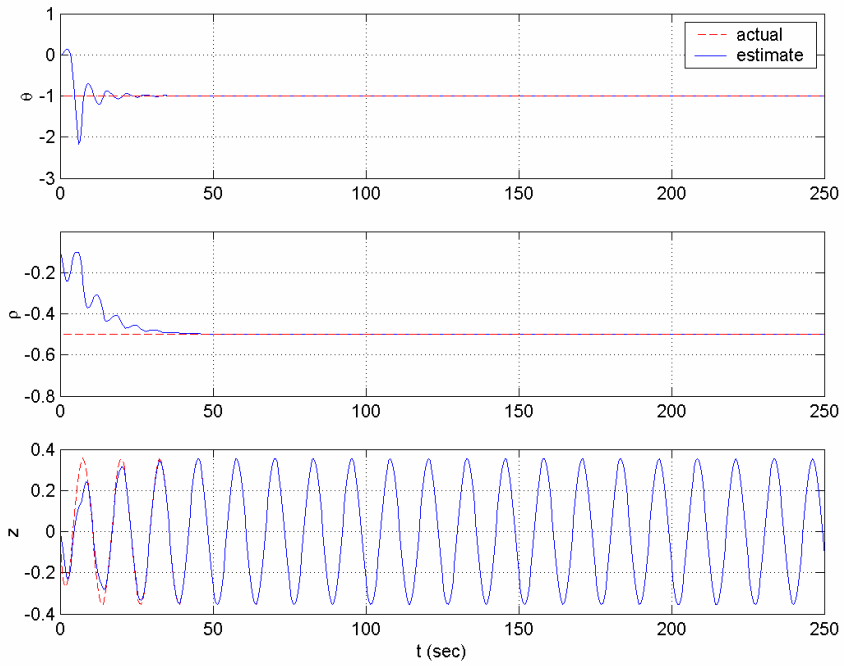

Figure 3C.21 *Simulation results for Example* 3.16.11 *assuming that both* $\operatorname{sgn}(\rho^*)$ *and a lower boundary for* $|\rho^*|$ *are known.*

Example 3.16.12 Consider the mass–spring–dashpot system shown in Figure 3C.22, where k is the spring constant, f is the viscous-friction or damping coefficient, m is the mass of the system, u is the forcing input, and x is the displacement of the mass M. We will use the differential equation

$$M\ddot{x} = u - k\,x - f\,\dot{x}$$

to describe the system. Our task is to estimate the values of the unknown coefficients M, k, and f.

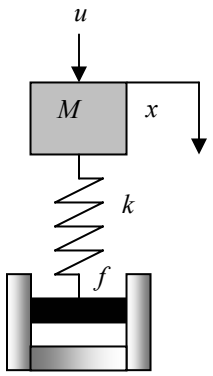

Figure 3C.22 *Mass–spring–dashpot system.*

We can rewrite the system equation as

$$x = \rho^* \left(u - M\, \ddot{x} - f\, \dot{x} \right),$$

where $\rho^* = \frac{1}{k}$. If we filter each side of this equation with $\Lambda(s) = (s+1)^2$, we obtain the bilinear parametric model

$$z = \rho^* \left(\theta^{*T} \phi + z_1 \right),$$

where $z = \frac{1}{\Lambda(s)} x$, $\theta^* = [M, f]^T$, $\phi = \left[-\frac{s^2}{\Lambda(s)} x,\ -\frac{s}{\Lambda(s)} x \right]^T$, and $z_1 = \frac{1}{\Lambda(s)} u$. Since $k > 0$, we know that $\text{sgn}(\rho^*) = 1$, and we can use the adaptive law for known $\text{sgn}(\rho^*)$ for estimation.

Now, let us assume that $f = 0.1$ kg/sec, $k = 5$ kg/sec^2, $M = 20$ kg, and $u = 0.1\cos\frac{\pi}{7}t + 0.05\sin\frac{\pi}{5}t$ kg m/sec^2. Choosing the design coefficients as $C_s = 0.1$, $\Gamma = \begin{bmatrix} 10000 & 0 \\ 0 & 100 \end{bmatrix}$, and $\gamma = 100$, and the initial parameters as $\theta_0 = [10,\ 0.2]^T$ and $\rho_0 = 0.05$, we can simulate the system and the parameter estimation process using the Simulink block **`Parameter Estimator*`** with appropriate choices and parameters in the scheme of Figure 3C.23. The results are shown in Figure 3C.24.

If we furthermore know that $\rho^* \geq 0.1$, we can apply the adaptive law for known sign and bound of ρ^* to our system. The simulation can be done using the scheme of Figure 3C.23 with appropriate modifications in the selections and parameters of **`Parameter Estimator`** and changing the value of ρ_0 to 0.1. The results are shown in Figure 3C.25. ■

* For details about this block, refer to the toolbox manual.

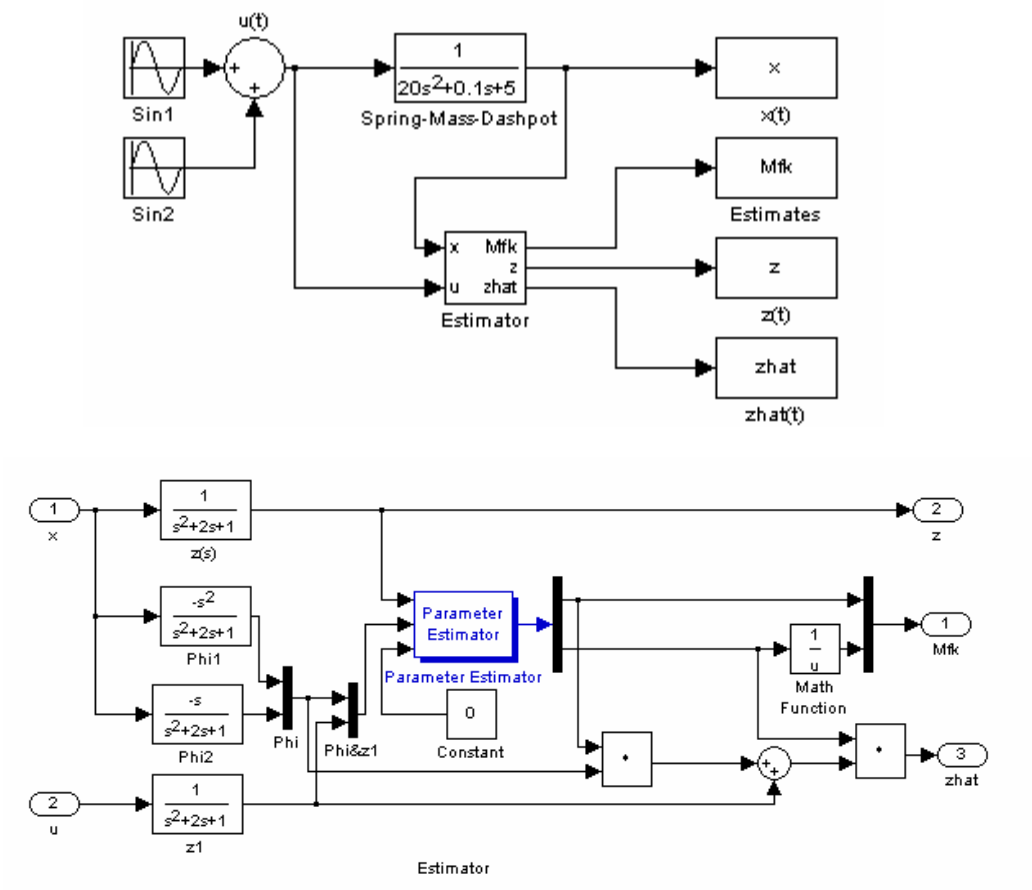

Figure 3C.23 *Simulink scheme for estimating the coefficients of the mass–spring–dashpot system in Example* 3.16.12.

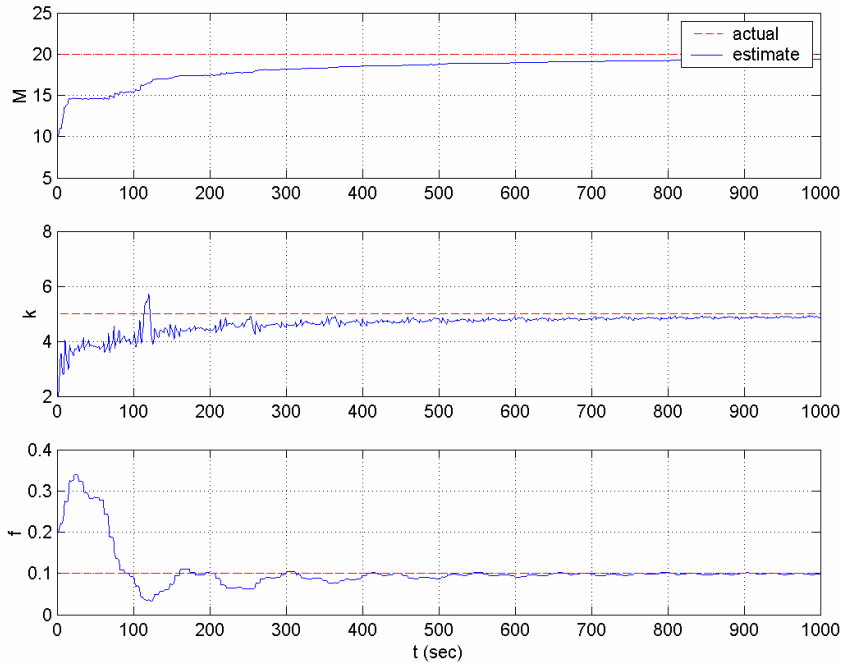

Figure 3C.24 *Simulation results for Example* 3.16.12 *assuming only* $\text{sgn}(\rho^*)$ *is known.*

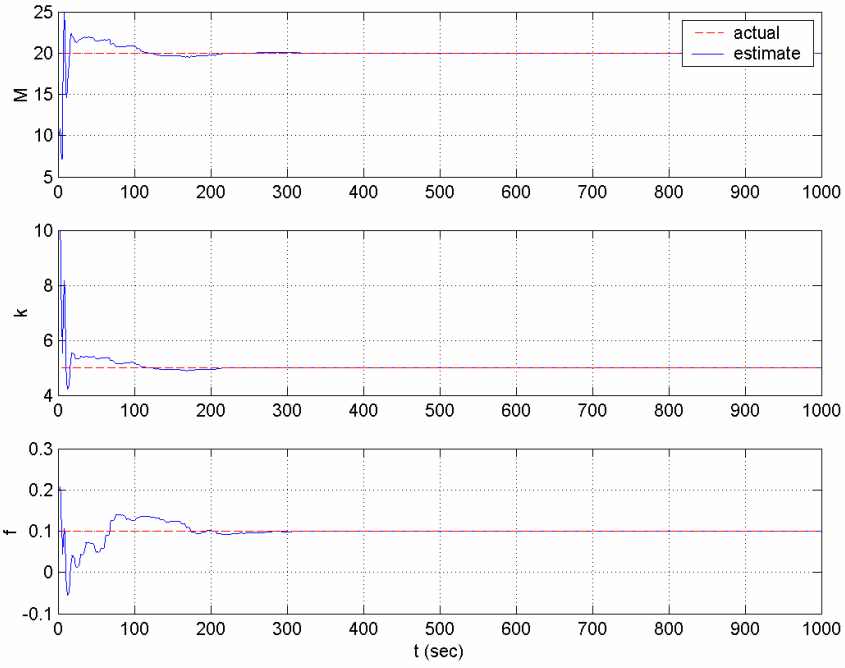

Figure 3C.25 *Simulation results for Example* 3.16.12 *assuming that both* $\text{sgn}(\rho^*)$ *and a lower boundary for* $|\rho^*|$ *are known.*

Bibliography

[1] P. A. IOANNOU AND J. SUN, *Robust Adaptive Control*, Prentice-Hall, Englewood Cliffs, NJ, 1996; also available at

http://www-rcf.usc.edu/~ioannou/Robust_Adaptive_Control.htm.