Προσομοίωση και Μοντελοποίηση Δυναμικών Συστημάτων
Πρώτη εργασία

Θεόδωρος Κατζάλης
ΑΕΜ: 9282
katzalis@auth.gr

5/04/2021

Περιεχόμενα

1 Πρώτη άσκηση

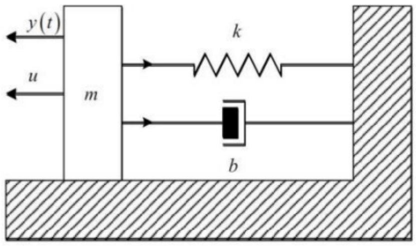


Figure 1: Σύστημα ελατήριο-μάζα-αποσβεστήρας

1.1 Μαθηματική δομή και γραμμική παραμετροποίηση

Για να βρούμε την μαθηματική δομή που περιγράφει το σύστημα, εφαρμόζουμε το δεύτερο νόμο του Νεύτωνα στο σύστημα ελατήριο-μάζα-αποσβεστήρας και έχουμε:

$$\Sigma F = m\ddot{y}$$
$$-ky - b\dot{y} + u = m\ddot{y}$$
$$\ddot{y} = -\frac{b}{m}\dot{y} - \frac{k}{m}y + \frac{u}{m}$$

Η παραπάνω διαφορική εξίσωση, θεωρώντας μηδενικές αρχικές συνθήκες, μπορεί να γραφτεί διαφορετικά στην γραμμική της μορφή ως:

$$\ddot{y} = \theta^{*T}\Delta$$
$$s^2 y = \theta^{*T}\Delta$$

όπου $\theta^* = \begin{bmatrix}\frac{b}{m} & \frac{k}{m} & \frac{1}{m}\end{bmatrix}^T$, $\Delta = \begin{bmatrix}-\Delta_1^T(s)y & \Delta_0^T(s)u\end{bmatrix}^T$ και $\Delta_i = \begin{bmatrix}s^i & s^{i-1}...1\end{bmatrix}^T$

Ο στόχος μας είναι να εκτιμήσουμε τις άγνωστες παραμέτρους (m, b, k) δηλαδή το διάνυσμα θ^*. Θεωρούμε πως δεν γνωρίζουμε τιμές ανώτερης τάξης της εξόδου y και της εισόδου u όπως συνηθίζεται εξαιτίας της δυσκολίας στον προσδιορισμό τους. Για αυτό το λόγο, εφαρμόζουμε ένα φίλτρο $\Lambda(s)$ για αποφύγουμε την ανάγκη να γνωρίζουμε αυτές τις τιμές. Πιο συγκεκριμένα:

Επιλέγουμε ένα ευσταθές φίλτρο με αρνητικές πραγματικές ρίζες ρ_1 και ρ_2, $\Lambda(s) = (s-\rho_1)(s-\rho_2) = s^2 - (\rho_1+\rho_2) + \rho_1 * \rho_2$ και διαιρούμε κατά μέλη:

$$z = \theta^{*T}\zeta$$

όπου

$$z = \frac{s^2 y}{\Lambda(s)}$$
$$\zeta = \begin{bmatrix}-\frac{\Delta_1^T(s)y}{\Lambda(s)} & \frac{\Delta_0^T(s)u}{\Lambda(s)}\end{bmatrix} = \begin{bmatrix}-\frac{[s \quad 1]}{\Lambda(s)}y & \frac{1}{\Lambda(s)}u\end{bmatrix}$$

Έχουμε

$$\Lambda(s) = s^2 + \lambda^T\Delta_1(s), \quad \lambda = \begin{bmatrix}-(\rho_1+\rho_2) & \rho_1 * \rho_2\end{bmatrix}^T$$

$$z = \frac{\Lambda(s) - \lambda^T \Delta_1(s)}{\Lambda(s)} y = y - \frac{\lambda^T \Delta_1(s)}{\Lambda(s)} y$$

$$y = z + \frac{\lambda^T \Delta_1(s)}{\Lambda(s)} y \qquad (1)$$

Ορίζουμε

$$\theta_1^* = \begin{bmatrix} \frac{b}{m} & \frac{k}{m} \end{bmatrix}^T, \quad \theta_2^* = \begin{bmatrix} \frac{1}{m} \end{bmatrix}$$

οπότε

$$z = \theta^{*T} \zeta$$

$$z = \theta_1^* \zeta_1 + \theta_2^* \zeta_2, \quad \zeta_1 = \frac{\Delta_1^T(s)}{\Lambda(s)} y, \quad \zeta_2 = \frac{\Delta_0^T(s)}{\Lambda(s)} u$$

$$\text{απο (1)} \implies y = \theta_1^{*T} \zeta_1 + \theta_2^{*T} \zeta_2 - \lambda^T \zeta_1 = \theta_\lambda^* \zeta, \quad \theta_\lambda^* = \begin{bmatrix} \theta_1^{*T} - \lambda^T & \theta_2^{*T} \end{bmatrix}^T$$

$$y = \begin{bmatrix} \frac{b}{m} - (-(\rho_1 + \rho_2)) & \frac{k}{m} - \rho_1 * \rho_2 & \frac{1}{m} \end{bmatrix}^T \zeta$$

Τέλος, έχουμε την γραμμική παραμετροποιημένη μορφή:

$$\boxed{y = \begin{bmatrix} \frac{b}{m} + (\rho_1 + \rho_2) & \frac{k}{m} - \rho_1 * \rho_2 & \frac{1}{m} \end{bmatrix}^T \zeta}$$

1.2 Σχεδιασμός αλγορίθμου ελαχίστων τετραγώνων

Έχοντας την γραμμικά παραμετροποιημένη μορφή που βρήκαμε προηγουμένως, θα χρησιμοποιήσουμε στην συνέχεια ως τεχνική εκτίμησης παραμέτρων, την μέθοδο των ελαχίστων τετραγώνων.

Θεωρούμε ότι έχουμε ένα αρχείο καταγραφής των τιμών της εξόδου ύστερα απο προκαθορισμένες εισόδους:

$$Y = \begin{bmatrix} y(1) & y(2) & ... & y(N) \end{bmatrix}^T$$

και μορφή συστήματος, όπου $\hat{y}$ η έξοδος που εκτιμούμε:

$$\hat{y} = \theta \phi$$

Ορίζουμε έναν πίνακα καταγραφής των αποτελεσμάτων των τιμών της εισόδου για κάθε συνιστώσα του ϕ:

$$\Phi = \begin{bmatrix} \phi_1(1) & \phi_2(1) & ... & \phi_N(1) \\ \phi_1(2) & \phi_2(2) & ... & \phi_N(2) \\ & & ... & \\ \phi_1(N) & \phi_2(N) & ... & \phi_N(N) \end{bmatrix}$$

Στόχος μας είναι να βρούμε εκείνο το θ που θα ελαχιστοποιήσει το σφάλμα μεταξύ της εκτίμησης και των τιμών του πραγματικού συστήματος, όπου σφάλμα:

$$e = Y - \hat{Y} = Y - \Phi\theta$$

οπότε

$$\theta_0 = \arg\max_\theta \frac{|e|^2}{2} = \arg\max_\theta \frac{e^T e}{2}$$

Θέλουμε να βρούμε το ολικό ελάχιστο της συνάρτησης:

$$V = |\frac{Y - \Phi\theta}{2}|^2$$

η οποία είναι κυρτή συνάρτηση λόγω της γραμμικοποιημένης μορφής και θα έχει ένα ολικό ελάχιστο. Για να το βρούμε έχουμε:

$$\left.\frac{dV}{d\theta}\right|_{\theta=\theta_0} = 0$$

$$(Y - \Phi\theta)^T(-\Phi) = 0$$

$$\theta_0^T \Phi^T \Phi = Y^T \Phi$$

και αν $\Phi^T\Phi$ αντιστρέφεται,

$$\boxed{\theta_0^T = Y^T\Phi(\Phi^T\Phi)^{-1}}$$

1.3 Προσομοίωση αλγορίθμου

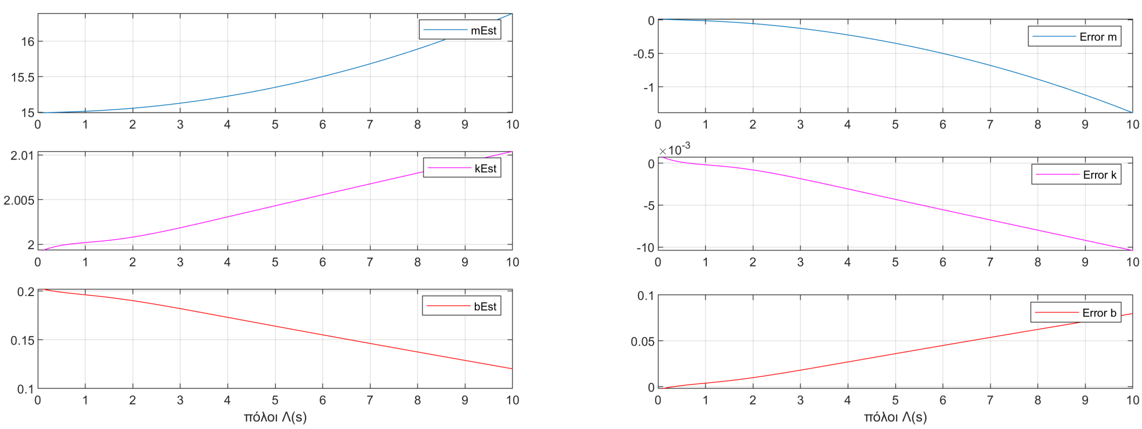
Στη συνέχεια προσομοιώσαμε τον αλγόριθμο ελαχίστων τετραγώνων σε `Matlab`:

- Για την λύση της διαφορικής εξίσωσης του συστήματος, προκειμένου να υπολογίσουμε τις μετρήσεις εξόδου Y, χρησιμοποιήσαμε την `ode45`.
- Για την εκτίμηση της απόκρισης του φίλτρου στις τιμές τις εξόδου, χρησιμοποιήσαμε την `lsim()`.
- Όπως υποδεικνύεται απο την εκφώνηση της άσκησης, χρησιμοποιήσαμε είσοδο `u(t) = 5sin(2t) + 10.5` και συλλέξαμε δεδομένα για χρονικό εύρος `0:0.1:10`.

1.3.1 Παρατηρήσεις

Για την επιλογή των ιδιοτιμών του φίλτρου, πραγματοποιήσαμε παραμετρική ανάλυση ως προς τους πόλους, βλέποντας το σφάλμα των εκτιμώμενων τιμών. Για λόγους ευκολίας στην ανάλυση, χρησιμοποιήσαμε μόνο διπλούς πόλους με εύρος παραμετρικής ανάλυσης `0.1:0.1:10`. Τελικά, **επιλέξαμε ως βέλτιστο διπλό πόλο την τιμή** `-0.5`.

Μπορούμε να παρατηρήσουμε ότι η επιλογή των πόλων κοντά στην αρχή του αριστερού μιγαδικού ημιεπιπέδου, έχει πολύ καλή απόκριση στην εκτίμηση των παραμέτρων εξαιτίας των ιδιοτήτων του μηχανικού μας συστήματος.



(a) Εκτιμήσεις

(b) Σφάλμα εκτιμήσεων

Figure 2: Παραμετρικές αναλύσεις ως προς τους πόλους του φίλτρου $\Lambda(s)$

Για διπλό πόλο με τιμή `-0.5` έχουμε:

$$\begin{aligned} m &= 15.000082 \quad (kg) \\ k &= 1.999883 \quad (kg/s^2) \\ b &= 0.198780 \quad (kg/s) \end{aligned}$$

Στη συνέχεια παραθέτουμε τον κώδικα `Matlab` για όλα τα παραπάνω.

1.3.2 Matlab κώδικας

```
%%% Theodoros Katzalis AEM:9282

clear;
clc;

m = 15;
b = 0.2;
k = 2;

u = @(t)5*sin(2*t) + 10.5;
timeRange = 0:0.1:10;

y0(1) = 0;
y0(2) = 0;
[time, y] = ode45(@(time, y)firstOrder(time, y, u, m, b, k), timeRange, y0);

polesRange = 0.1:0.1:10; % best pole around 0.5 based on plots
count = 0;
for i = polesRange
    count = count + 1;
    eig1 = -i;
    eig2 = -i;
    filterCoeff(:,count) = [-(eig1+eig2) eig1*eig2];
    thetaEstimate(:,count) = estimator(eig1, eig2, y, u, timeRange);

    theta(1,count) = thetaEstimate(1,count) + filterCoeff(1,count);
    theta(2,count) = thetaEstimate(2,count) + filterCoeff(2,count);
    theta(3,count) = thetaEstimate(3,count);

    mEst(count) = 1/theta(3,count);
    kEst(count) = theta(2,count) * mEst(count);
    bEst(count) = theta(1,count) * mEst(count);
end

figure();
plot(polesRange, mEst, polesRange, kEst, polesRange, bEst);
grid on;
legend('m', 'k', 'b');

figure();
title("Estimated parameters");
subplot(3,1,1);
plot(polesRange, mEst);
grid on;
legend("mEst");
subplot(3,1,2);
```

```
plot(polesRange, kEst, 'color', 'magenta');
grid on;
legend("kEst");
subplot(3,1,3);
plot(polesRange, bEst, 'color', 'red');
grid on;
legend("bEst");
xlabel('poles $\Lambda(s)$', 'interpreter','latex');
print('estimations', '-dpng', '-r300');

figure();
title("Estimation error");
subplot(3,1,1);
plot(polesRange, m - mEst);
grid on;
legend("Error m");
subplot(3,1,2);
plot(polesRange, k - kEst, 'color', 'magenta');
grid on;
legend("Error k");
subplot(3,1,3);
plot(polesRange, b - bEst, 'color', 'red');
grid on;
legend("Error b");
xlabel('poles $\Lambda(s)$', 'interpreter','latex');
print('errors', '-dpng', '-r300');

fprintf("Estimations:\nm = %f\nk = %f\nb = %f\n", mEst(1), kEst(1), bEst(1));

function theta = estimator(eig1, eig2, y, u, timeRange)

filter = [1 -(eig1+eig2) eig1*eig2];

phi = zeros(length(timeRange), 3);
phi(:,1) = lsim(tf(-[1 0], filter), y(:,1), timeRange);
phi(:,2) = lsim(tf(-[0 1], filter), y(:,1), timeRange);
phi(:,3) = lsim(tf([0 1], filter), u(timeRange), timeRange);

theta = y(:,1)' * phi / (phi' * phi);

end

function dy = firstOrder(t, y, u, m, b, k)

dy(1) = y(2);
dy(2) = -(b/m)*y(2) -(k/m)*y(1) + u(t)/m;

dy = dy';

end
```

2 Δεύτερη άσκηση

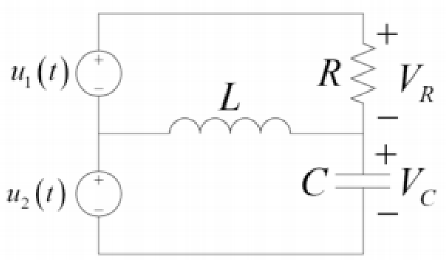


Figure 3: Ηλεκτρικό κύκλωμα προς ανάλυση

2.1 Εκτίμηση πίνακα μεταφοράς με την μέθοδο ελαχίστων τετραγώνων

Αρχικά πρέπει να βρούμε την μαθηματική δομή του συστήματος, εφαρμόζοντας κυκλωματική ανάλυση. Αξίζει να σημειωθεί ότι σε αυτό το σύστημα έχουμε δύο εισόδους u_1 και u_2 και δύο εξόδους V_R και V_R. Θεωρώντας τον κόμβο που ενώνονται το πηνίο, ο πυκνωτής και η αντίσταση ως E και τάση V_E αντίστοιχα, με βάση τον νόμο KCL έχουμε:

$$i_1(t) = i_2(t) + i_3(t)$$

$$\frac{V_R(t)}{R} = \frac{1}{L}\int_{-\infty}^{t} V_E(t)\,dt + C\dot{V_C}(t)$$

$$\frac{V_R(t)}{R} = \frac{1}{L}\int_{-\infty}^{t} (V_C - u_2(t))\,dt + C\dot{V_C}(t)$$

$$\begin{cases} \frac{V_R(t)}{R} = \frac{1}{L}\int_{-\infty}^{t}(V_C(t) - u_2(t))\,dt + C\dot{V_C}(t) \\ \frac{V_R(t)}{R} = \frac{1}{L}\int_{-\infty}^{t}(u_1(t) - V_R(t))\,dt + C\dot{V_C}(t) \end{cases}$$

$$\begin{cases} \frac{u_1(t)+u_2(t)-V_C(t)}{R} = \frac{1}{L}\int_{-\infty}^{t}(V_C(t) - u_2(t))\,dt + C\dot{V_C}(t) \\ \frac{V_R(t)}{R} = \frac{1}{L}\int_{-\infty}^{t}(u_1(t) - V_R(t))\,dt + C\dot{u_1}(t) + C\dot{u_2}(t) - C\dot{V_R}(t) \end{cases}$$

$$\begin{cases} \ddot{V_C}(t) + \frac{\dot{V_C}(t)}{RC} + \frac{V_C(t)}{LC} = \frac{\dot{u_1}(t)}{RC} + \frac{\dot{u_2}(t)}{RC} + \frac{u_2(t)}{LC} \\ \ddot{V_R}(t) + \frac{\dot{V_R}(t)}{RC} + \frac{V_R(t)}{LC} = \ddot{u_1}(t) + \frac{u_1(t)}{LC} + \ddot{u_2}(t) \end{cases} \tag{1}$$

$$\begin{cases} \ddot{V_C}(t) = -\frac{\dot{V_C}(t)}{RC} - \frac{V_C(t)}{LC} + \frac{\dot{u_1}(t)}{RC} + \frac{\dot{u_2}(t)}{RC} + \frac{u_2(t)}{LC} \\ \ddot{V_R}(t) = -\frac{\dot{V_R}(t)}{RC} - \frac{V_R(t)}{LC} + \ddot{u_1}(t) + \frac{u_1(t)}{LC} + \ddot{u_2}(t) \end{cases} \tag{2}$$

Για να βρούμε τον πίνακα μεταφοράς, κάνουμε μετασχηματισμό Laplace στην (1), θεωρώντας μηδενικές αρχικές συνθήκες και έχουμε:

$$\begin{cases} s^2V_C + \frac{sV_C}{RC} + \frac{V_C}{LC} = \frac{su_1}{RC} + \frac{su_2}{RC} + \frac{u_2}{LC} \\ s^2V_R(t) + \frac{sV_R}{RC} + \frac{V_R}{LC} = s^2u_1 + \frac{u_1}{LC} + s^2u_2 \end{cases}$$

και σε μορφή πίνακα:

$$\begin{bmatrix} V_C \\ V_R \end{bmatrix} = \begin{bmatrix} \frac{\frac{s}{RC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} & \frac{\frac{s}{RC}+\frac{1}{LC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} \\ \frac{s^2+\frac{1}{LC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} & \frac{s^2}{s^2+\frac{s}{RC}+\frac{1}{LC}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

οπότε πίνακας μεταφοράς

$$H(s) = \begin{bmatrix} \frac{\frac{s}{RC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} & \frac{\frac{s}{RC}+\frac{1}{LC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} \\ \frac{s^2+\frac{1}{LC}}{s^2+\frac{s}{RC}+\frac{1}{LC}} & \frac{s^2}{s^2+\frac{s}{RC}+\frac{1}{LC}} \end{bmatrix} \tag{3}$$

Το πρόβλημα εκτίμησης του πίνακα μεταφοράς ανάγεται στο πρόβλημα εκτίμησης των όρων $\frac{1}{RC}$ και $\frac{1}{LC}$, το οποίο με την σειρά του ανάγεται στην εκτίμηση των παραμέτρων ενός εκ των δύο διαφορικών εξισώσεων του συστήματος (2). Παρατηρώντας την (2), η μελέτη του V_R απαιτεί οχτώ στοιχεία για το θ^*, ενώ αντίστοιχα για το V_C έξι στοιχεία. Συνεπώς, θα χρησιμοποιήσουμε την διαφορική εξίσωση που περιλαμβάνει το V_C για την εκτίμηση των παραμέτρων της.

Εφαρμόζοντας παρόμοια μαθηματική ανάλυση που δείξαμε λεπτομερώς στην πρώτη άσκηση, μπορούμε να φέρουμε την διαφορική εξίσωση στην μορφή:

$$y = \theta_\lambda^* \zeta$$

όπου

$$\theta_\lambda^* = \begin{bmatrix} \frac{1}{RC} + (\rho_1 + \rho_2) & \frac{1}{LC} - \rho_1 * \rho_2 & \frac{1}{RC} & 0 & \frac{1}{RC} & \frac{1}{LC} \end{bmatrix}^T$$

$$\zeta = \begin{bmatrix} -\frac{[s \quad 1]}{\Lambda(s)} y & \frac{[s \quad 1]}{\Lambda(s)} u_1 & \frac{[s \quad 1]}{\Lambda(s)} u_2 \end{bmatrix}^T$$

$$\Lambda(s) = (s - \rho_1)(s - \rho_2) = s^2 - (\rho_1 + \rho_2)s + \rho_1 * \rho_2$$

και το διάνυσμα θ^* που θέλουμε να εκτιμήσουμε

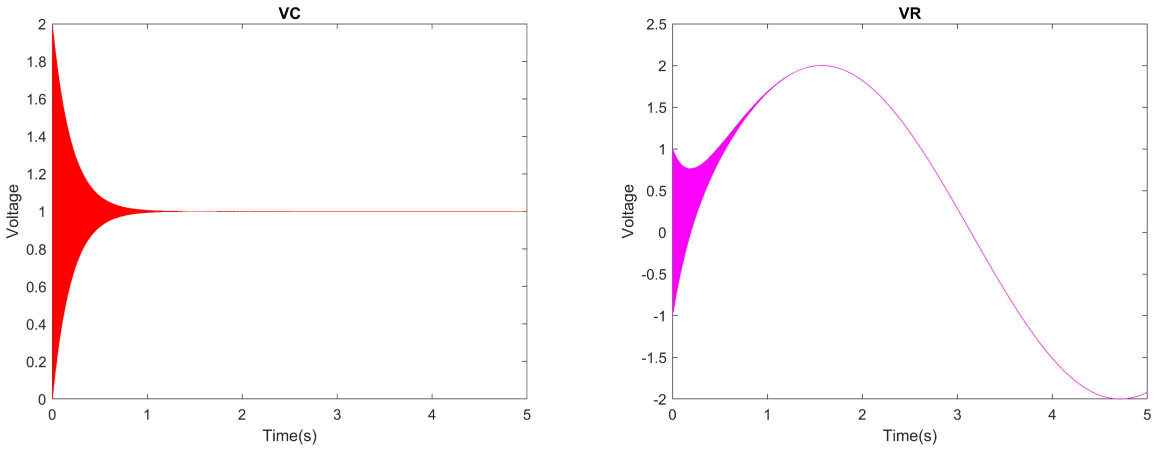
$$\theta^* = \begin{bmatrix} \frac{1}{RC} & \frac{1}{LC} & \frac{1}{RC} & 0 & \frac{1}{RC} & \frac{1}{LC} \end{bmatrix}^T$$

Στη συνέχεια, προσομοιώνουμε τον αλγόριθμο των ελαχίστων τετραγώνων με την χρήση `Matlab`. Σε αυτήν την περίπτωση μας δίνονται οι τιμές των εξόδων, οπότε για την εκτίμηση των παραμέτρων, έχουμε ότι χρειαζόμαστε για τον υπολογισμό του θ^*:

$$\theta_0^T = Y^T \Phi (\Phi^T \Phi)^{-1}$$

Ωστόσο, θα επιλύσουμε την διαφορική εξίσωση, αφού εκτιμήσουμε τις τιμές, έτσι ώστε να μπορούμε να συγκρίνουμε τις πραγματικές τιμές του συστήματος με το μοντέλο προσομοίωσής μας.

Αξίζει να σημειωθεί ότι στο συγκεκριμένο σύστημα **η επιλογή των χρονικών στιγμών δειγματοληψίας** των τιμών εξόδου παίζει πολύ σημαντικό ρόλο. Για να μελετήσουμε την συμπεριφορά και την εξέλιξη των δεδομένων δημιουργήσαμε τα εξής διαγράμματα:

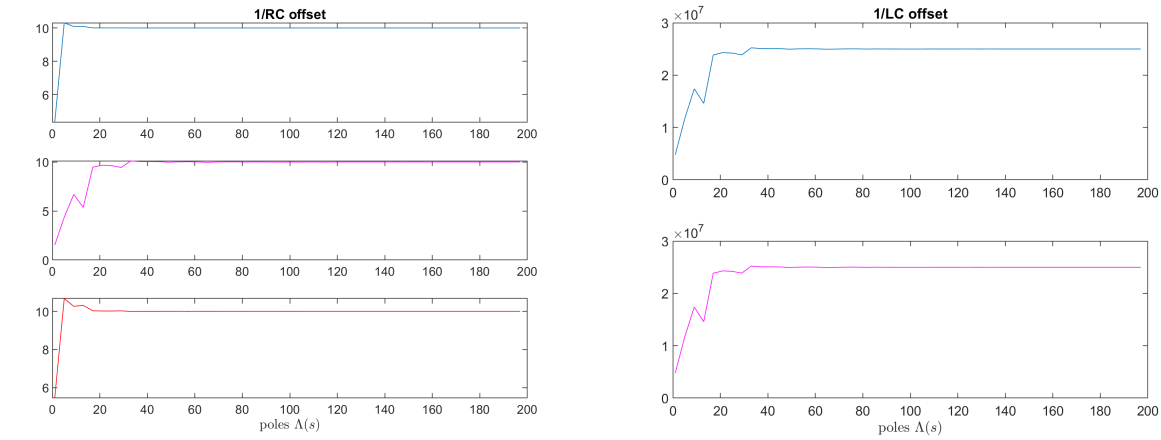


(a) Τιμές V_C (b) Τιμές V_R

Figure 4: Δειγματοληψία εξόδων

Παρατηρούμε σχετικά με τις τιμές V_C ότι υπάρχει ιδιαίτερα σημαντική πληροφορία σε πολύ μικρές χρονικές στιγμές στην αφετηρία του συστήματος, όπου συμβαίνουν απότομες ταλαντώσεις. Προκειμένου τα δεδομένα μας να συμπεριλάβουν αυτό το μεταβατικό φαινόμενο, το οποίο το θεωρούμε σημαντικό διότι προσδιορίζει τα χαρακτηριστικά του συστήματος, χρησιμοποιήσαμε ιδιαίτερα μικρό step, κατά την δειγματοληψία, της τάξης 1e-4 έως 1e-6.

Ιδιαίτερο ενδιαφέρον παρουσιάζει ακόμη η επιλογή του φίλτρου $\Lambda(s)$ και η επίδραση του στην απόκριση του συστήματος και στο σφάλμα μοντελοποίησης αλλά και στην απόκλιση των τιμών του διανύσματος εκτίμησης θ^* για τους **κοινούς όρους** που διαθέτει (**τρεις** φορές το $\frac{1}{RC}$ και **δύο** φορές το $\frac{1}{LC}$). Μια ορθή εκτίμηση θα είναι σίγουρα αυτή στην οποία οι κοινοί όροι συγκλίνουν μεταξύ τους αλλά και το σφάλμα μοντελοποίησης είναι σχετικά μικρό.



(a) Εξάρτηση των 1/RC με τους πόλους του $\Lambda(s)$ (b) Εξάρτηση των 1/LC με τους πόλους του $\Lambda(s)$

Figure 5: Παραμετρικές αναλύσεις των κοινών όρων του θ^* (εύρος δειγματοληψίας 0:1e-6:5 και για την παραμετρική ανάλυση των πόλων 1:4:200)

Με βάση τα παραπάνω διαγράμματα, καταλήξαμε ότι μια σωστή επιλογή των πόλων του φίλτρου θα πρέπει να συγκλίνει τα τρία $\frac{1}{RC}$ και τα δύο $\frac{1}{LC}$, το οποίο συμβαίνει μετά την τιμή -20 όπου όλες οι τιμές σταθεροποιούνται. Με τεχνική "trial and error" και παραμετρικές αναλύσεις, καταλήξαμε στον διπλό πόλο με τιμή -100, έχοντας το ακόλουθο σφάλμα:

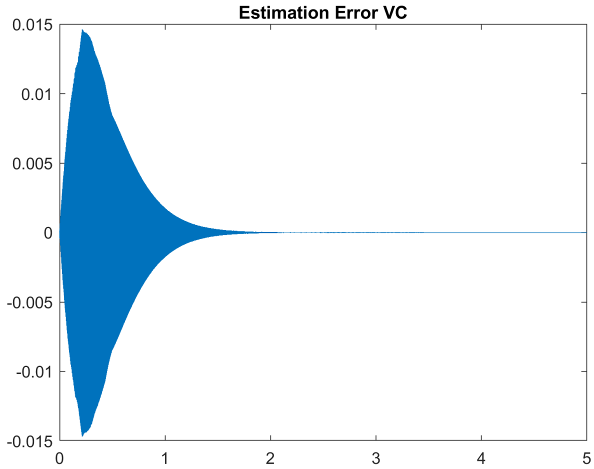


Figure 6: Σφάλμα V_C ως προς τον χρόνο (s)

Τιμές $\frac{1}{RC}$ και $\frac{1}{LC}$ για τις συγκεκριμένες τιμές:

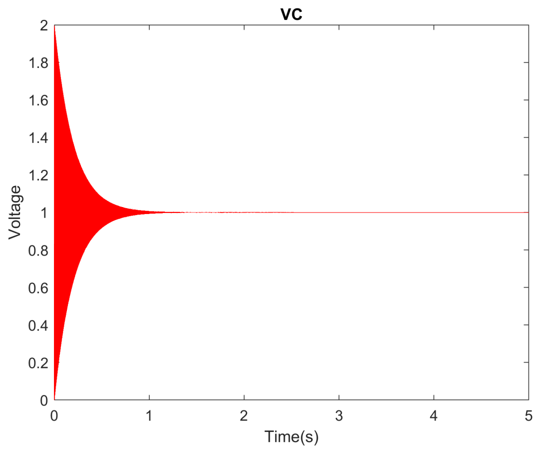
$$\frac{1}{RC} = 9.999635$$
$$\frac{1}{LC} = 24992478.895320$$

Κάνοντας αντικατάσταση των τιμών αυτών στον πίνακα μεταφοράς (3) που βρήκαμε προηγουμένως, έχουμε το ζητούμενο.

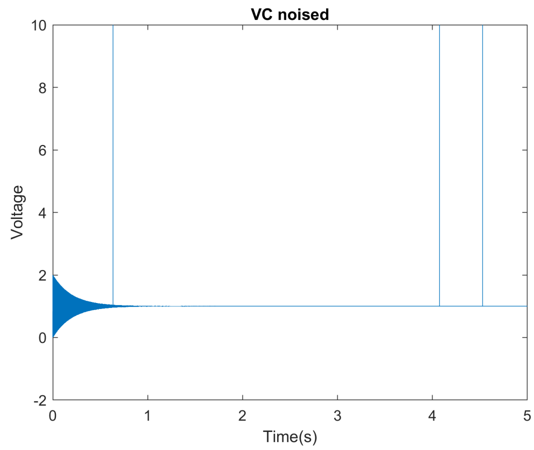
2.2 Επίδραση outliers

Η μέθοδος των ελαχίστων τετραγώνων επηρεάζεται σε πολύ σημαντικό βαθμό απο την **ποιότητα** της πληροφορίας των δεδομένων που θα χρησιμοποιηθούν ως κριτήριο για την ελαχιστοποίηση του σφάλματος και την εύρεση της καλύτερης εκτίμησης. Αυτό συμβαίνει διότι η τεχνική αυτή, στον υπολογισμό του σφάλματος εκτίμησης του οποίου θέλουμε να ελαχιστοποιήσουμε, αντιμετωπίζει **ισοδύναμα** όλες τις τιμές του αρχείου καταγραφής μας. Συνεπώς, τιμές οι οποίες συμπεριλαμβάνουν **θόρυβο**, μπορούν ανάλογα την τάξη μεγέθους του θορύβου, να δημιουργήσουν σημαντική απόκλιση στην εκτίμηση των παραμέτρων. Προκειμένου να δούμε αυτήν την επίδραση, εσκεμμένα σε τρεις τυχαίες χρονικές στιγμές των δεδομένων V_C, προσθέσαμε τις τιμές 100, 110 και 130.

Για να οπτικοποιήσουμε την διαφορά στα δείγματα των εξόδων V_C παραθέτουμε τα ακόλουθα διαγράμματα (για λόγους ευκρίνειας στο δεύτερο διάγραμμα η κλίμακα του y άξονα τερματίζει στην τιμή 10. Οι τιμές των spikes είναι όμως 100, 110 και 130):



(a) Πραγματικές τιμές V_C



(b) Τιμές V_C με outliers

Με βάση λοιπόν την συμπερίληψη των outliers στα δεδομένα μας και για διπλό πόλο με τιμή `-100`, χρησιμοποιώντας το ίδιο εύρος συλλογής δεδομένων με τα προηγούμενα αποτελέσματα `0:1e-6:5` έχουμε:

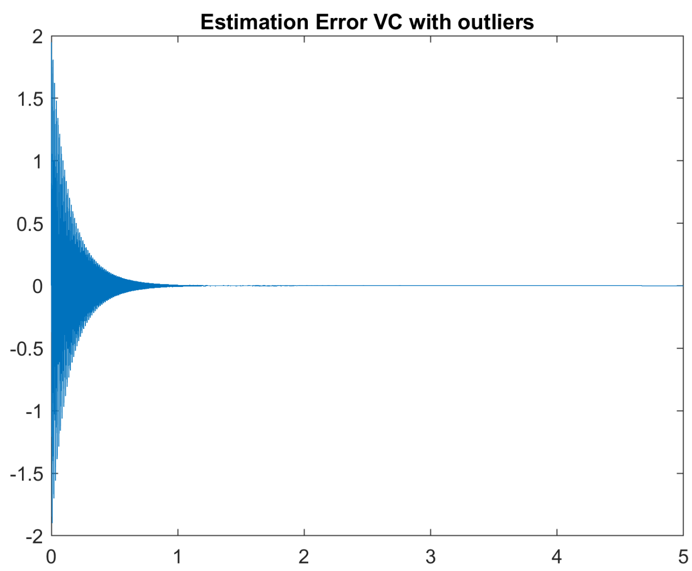


Figure 8: Σφάλμα V_C με outliers ως προς τον χρόνο (s)

Το παραπάνω διάγραμμα επαληθεύει την στενή εξάρτηση της απόδοσης του αλγορίθμου με την ποιότητα των δεδομένων, διότι το σφάλμα πλέον γίνεται ιδιαίτερα σημαντικό συγκριτικά με το μέγεθος της εξόδου. Αξίζει βέβαια να σημειωθεί ότι αρχικά εξαιτίας των outliers, τα τρία διαφορετικά $\frac{1}{RC}$ και αντίστοιχα τα $\frac{1}{LC}$ απέκλιναν σημαντικά μεταξύ τους, οπότε θεωρήσαμε τελικές τιμές τον μέσο όρο τους, για τις οποίες τιμές υπολογίστηκε η διαφορική εξίσωση.

Τιμές $\frac{1}{RC}$ και $\frac{1}{LC}$ συμπεριλαμβάνοντας outliers:

$$\frac{1}{RC_1} = 31.339811$$
$$\frac{1}{RC_2} = 0.004005$$
$$\frac{1}{RC_3} = 31.076201$$
$$\frac{1}{LC_1} = 229205.765704$$
$$\frac{1}{LC_2} = 229218.426813$$
$$\text{Mean}\frac{1}{RC} = 20.806672$$
$$\text{Mean}\frac{1}{LC} = 229212.096259$$

2.3 Matlab κώδικας

Για όλα τα παραπάνω παρατίθεται ο κώδικας Matlab.

```matlab
%%% Theodoros Katzalis AEM:9282

clear;
clc;

format long;

% samples
timeEnd = 5;
timeRange = 0:0.000001:timeEnd;
N = length(timeRange);

VR = zeros(1,N);
VC = zeros(1,N);

% collect data
index = 0;
for t = timeRange
    index = index + 1;
    Vout = v(t);
    VC(index) = Vout(1);
    VR(index) = Vout(2);
end

% inputs
input1 = @(t)2*sin(t);
input2 = ones(1,N);

% filter design
polesRange = [100];
%polesRange = 1:4:200;
```

```
simulSize = length(polesRange);
thetaLinear = zeros(6,simulSize);
filterCoeff = zeros(2,simulSize);
count = 0;

% parametric analysis
for i = polesRange
    count = count + 1;
    fprintf("iteation id %d/%d\n",count, simulSize);
    eig1 = -i;
    eig2 = -i;
    filterCoeff(:,count) = [-(eig1+eig2) ; eig1*eig2];
    thetaLinear(:,count) = estimator(eig1, eig2, N, VC, input1, input2, ...
        timeRange);
end

invRC1 = thetaLinear(1,:) + filterCoeff(1,:);
invLC1 = thetaLinear(2,:) + filterCoeff(2,:);
invRC2 = thetaLinear(3,:);
invRC3 = thetaLinear(5,:);
invLC2 = thetaLinear(6,:);

fprintf('1/RC1=%f\n1/RC2=%f\n1/RC3=%f\n1/LC1=%f\n1/LC2=%f\n', invRC1(end), ...
        invRC2(end), invRC3(end), invLC1(end), invLC2(end));

% calculate init conditions
Vdiff = v(1e-10);
Vinit = v(0);
x0(1) = Vinit(1);
x0(2) = (Vdiff(1) - Vinit(1))/1e-10; % approximate derivative

% solve ode and compare the results with data
[time,x] = ode45(@(time,x)firstOrder(time,x, invRC1(end), invLC1(end)), ...
    timeRange, x0);

%PLOTS

% error VR and VC estimated
figure();
plot(timeRange, VC(:) - x(:,1));
title("Estimation Error VC");
print('VCerror', '-dpng', '-r300');

VRhat = ones(1,length(time));
VRhat = VRhat + input1(time)' - x(:,1)';
figure();
plot(timeRange, abs(VR(:)) - abs(VRhat(:)));
title("Estimation Error VR");

% poles parametric analysis
figure();
subplot(311);
plot(polesRange, invRC1);
title("1/RC offset");
subplot(312);
plot(polesRange, invRC2, 'color', 'magenta');
subplot(313);
plot(polesRange, invRC3, 'color', 'red');
xlabel('poles $\Lambda(s)$', 'interpreter', 'latex');
print('RCoffset', '-dpng', '-r300');

figure();
subplot(211);
```

```
plot(polesRange, invLC1);
title("1/LC offset");
subplot(212);
plot(polesRange, invLC2, 'color', 'magenta');
xlabel('poles $\Lambda(s)$', 'interpreter', 'latex');
print('LCoffset', '-dpng', '-r300');

%plot output
figure();
plot(timeRange,VC, 'color','red');
title("VC");
xlabel("Time(s)");
ylabel("Voltage");
print('vc', '-dpng', '-r300');

figure();
plot(timeRange,VR, 'color', 'magenta');
title("VR");
xlabel("Time(s)");
ylabel("Voltage");
print('vr', '-dpng', '-r300');


%%%%%% QUESTION 2

fprintf("\n...OUTLIERS...\n");
% add three outliers to three random positions in the data
rng(0,'twister');
randomIndex = randi([1 length(VC)], 1, 3);
%outlier  = randi([1e2 1e3], 1, 3);
outlier = [100 110 130];
VCnoised = VC;
VCnoised(randomIndex) = VCnoised(randomIndex) + outlier;

thetaLinear = estimator(-100, -100, N, VCnoised, input1, input2, timeRange);

invRC1 = thetaLinear(1) + 200;
invLC1 = thetaLinear(2) + 1e4;
invRC2 = thetaLinear(3);
invRC3 = thetaLinear(5);
invLC2 = thetaLinear(6);

meanRC = (invRC1 + invRC2 + invRC3) / 3;
meanLC = (invLC1 + invLC2) / 2;

fprintf('1/RC1=%f\n1/RC2=%f\n1/RC3=%f\n1/LC1=%f\n1/LC2=%f\n', invRC1, ...
        invRC2, invRC3, invLC1, invLC2);
fprintf("mean 1/RC = %f\nmean 1/LC = %f\n", meanRC, meanLC);

% calculate init conditions
Vdiff = v(1e-10);
Vinit = v(0);
x0(1) = Vinit(1);
x0(2) = (Vdiff(1) - Vinit(1))/1e-10; % approximate derivative

% solve ode and compare the results with data
[time,x] = ode45(@(time,x)firstOrder(time, x, meanRC, meanLC), timeRange, x0);

% error VR and VC estimated
figure();
plot(timeRange, VC(:) - x(:,1));
title("Estimation Error VC with outliers");
figure();
```

```
plot(timeRange, x(:,1));
title("VC estimation with outliers");
print('vcestnoised', '-dpng', '-r300');

figure();
plot(timeRange,VCnoised);
title("VC noised");
xlabel("Time(s)");
ylabel("Voltage");
axis([0 5 -2 10])
print('vcnoised', '-dpng', '-r300');

function dx = firstOrder(t,x,invRC,invLC)

% diff(Vc,t,2) + diff(Vc,t,1)*1/RC + Vc/LC = diff(u1,t,1)/RC + diff(u2,t,1)+ u2/LC
% u1(t) = 2sin(t)
% u2(t) = 1

dx(1) = x(2);
dx(2) = -x(2)*invRC - x(1)*invLC + 2*cos(t)*invRC + 0 + invLC;

dx = dx';

end

function theta = estimator(eig1, eig2, N, VC, input1, input2, timeRange)

filter = [1 -(eig1+eig2) eig1*eig2];

phi = zeros(N, 6);
phi(:,1) = lsim(tf(-[1 0], filter), VC, timeRange);
phi(:,2) = lsim(tf(-[0 1], filter), VC, timeRange);
phi(:,3) = lsim(tf([1 0],filter), input1(timeRange), timeRange);
phi(:,4) = lsim(tf([0 1],filter), input1(timeRange), timeRange);
phi(:,5) = lsim(tf([1 0],filter), input2, timeRange);
phi(:,6) = lsim(tf([0 1],filter), input2, timeRange);

theta = VC * phi / (phi' * phi);

end
```