

# A-Appendix

## A-Workshop1A

```
1 %% A.1
2 % RLS without noise
3
4 clear all; close all; clc
5 load('data1.mat');
6
7 % Constants
8 Fs = 8192; %sampling freq
9 D1 = 1 * Fs; % delay 1
10 D2 = 2.5 * Fs; % delay 2
11 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
12
13 % RLS with inital params = 0
14
15 init_params = [1; 0; 0];
16 P_init_scale = 1;
17
18 % ----- %
19 %      RLS: Different initial conditions.
20 % ----- %
21
22 % RLS with initial params = 0
23
24 [output, theta.hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
25
26 figure(1)
27 subplot(2,1,1)
28 hold on
29 plot(timespan, theta.hat(1,:))
30 plot(timespan, theta.hat(2,:))
31 plot(timespan, theta.hat(3,:))
32 hold off
33
34 title('Parameter values (RLS no noise) b_2=b_3 = 0')
35 xlabel('Seconds')
36 ylabel('Parameter value')
37 legend('b_1', 'b_2', 'b_3')
38 ylim([0 1])
39
40
41 % RLS with inital params = 1
42
43 init_params = [1; 1; 1];
44 P_init_scale = 1;
45
46 [output, theta.hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
47
48
49 subplot(2,1,2)
50 hold on
51 plot(timespan, theta.hat(1,:))
52 plot(timespan, theta.hat(2,:))
53 plot(timespan, theta.hat(3,:))
54 hold off
55
56 title('Parameter values (RLS no noise) b_2=b_3 = 1')
57 xlabel('Seconds')
58 ylabel('Parameter value')
59 legend('b_1', 'b_2', 'b_3')
60 ylim([0 1])
61
62 saveas(gcf, 'figures/q1a_params.png')
```

```

63
64 %%
65 % ----- %
66 %      RLS: P scaling
67 % ----- %
68 clear all
69 close all
70 clc
71 load('data1.mat');
72
73 Fs = 8192; %sampling freq
74 D1 = 1 * Fs; % delay 1
75 D2 = 2.5 * Fs; % delay 2
76 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
77
78
79 % RLS with P_init_scale = 1
80 init_params = [1; 0; 0];
81 P_init_scale = 1;
82
83 [output1, theta_hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
84
85 figure(2)
86 subplot(3,1,1)
87 hold on
88 plot(timespan,theta_hat(1,:))
89 plot(timespan,theta_hat(2,:))
90 plot(timespan, theta_hat(3,:))
91 hold off
92
93 title('Parameter values (RLS no noise) Init P scale = 1')
94 xlabel('Seconds')
95 ylabel('Parameter value')
96 legend('b_1', 'b_2', 'b_3')
97 ylim([0 1.01])
98
99
100 % RLS with P_init_scale = 0.01
101 init_params = [1; 0; 0];
102 P_init_scale = 0.01;
103
104 [output001, theta_hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
105
106 figure(2)
107 subplot(3,1,2)
108 hold on
109 plot(timespan,theta_hat(1,:))
110 plot(timespan,theta_hat(2,:))
111 plot(timespan, theta_hat(3,:))
112 hold off
113
114 title('Parameter values (RLS no noise) Init P scale = 0.01')
115 xlabel('Seconds')
116 ylabel('Parameter value')
117 legend('b_1', 'b_2', 'b_3')
118 ylim([0 1.01])
119
120 % RLS with P_init_scale = 100
121 init_params = [1; 0; 0];
122 P_init_scale = 100;
123
124 [output100, theta_hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
125
126 figure(2)
127 subplot(3,1,3)
128 hold on
129 plot(timespan,theta_hat(1,:))
130 plot(timespan,theta_hat(2,:))

```

```

131 plot(timespan, theta_hat(3,:))
132 hold off
133
134 title('Parameter values (RLS no noise) Init P scale = 100')
135 xlabel('Seconds')
136 ylabel('Parameter value')
137 legend('b_1', 'b_2', 'b_3')
138 ylim([0 1.01])
139
140 saveas(gcf, 'figures/qla.Pinit.png')
141
142 figure(6)
143 subplot(3,1,1)
144 plot(timespan, transpose(loudspeaker) - output1)
145 title('Amplitude Difference (RLS without noise) Pinit = 1*I')
146 xlabel('Seconds')
147 ylabel('Amplitude')
148
149 subplot(3,1,2)
150 plot(timespan, transpose(loudspeaker) - output001)
151 title('Amplitude Difference (RLS without noise) Pinit = 0.01*I')
152 xlabel('Seconds')
153 ylabel('Amplitude')
154
155 subplot(3,1,3)
156 plot(timespan, transpose(loudspeaker) - output100)
157 title('Amplitude Difference (RLS without noise) Pinit = 100*I')
158 xlabel('Seconds')
159 ylabel('Amplitude')
160
161 saveas(gcf, 'figures/qla.Pinit.output.diff.png')
162
163 %%
164 % ----- %
165 %      RLS: With Noise
166 % ----- %
167
168 clear all
169 close all
170 clc
171 load('data1.mat');
172
173 Fs = 8192; %sampling freq
174 D1 = 1 * Fs; % delay 1
175 D2 = 2.5 * Fs; % delay 2
176 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
177
178 init_params = [1; 0; 0];
179 P_init_scale = 1;
180
181 [output, theta_hat] = RLS_function(loudspeaker, mikel, init_params, P_init_scale);
182 [noisy_output, noisy_theta_hat] = RLS_function(loudspeaker, noisymikel, init_params, ...
183     P_init_scale);
184
185 figure(4)
186 subplot(2,1,1)
187 hold on
188 plot(timespan, theta_hat(1,:))
189 plot(timespan, theta_hat(2,:))
190 plot(timespan, theta_hat(3,:))
191 hold off
192
193 title('Parameter values (RLS no noise)')
194 xlabel('Seconds')
195 ylabel('Parameter value')
196 legend('b_1', 'b_2', 'b_3')
197 ylim([0 1])

```

```

198 subplot(2,1,2)
199 hold on
200 plot(timespan,noisy_theta_hat(1,:))
201 plot(timespan,noisy_theta_hat(2,:))
202 plot(timespan, noisy_theta_hat(3,:))
203 hold off
204
205 title('Parameter values (RLS with noise)')
206 xlabel('Seconds')
207 ylabel('Parameter value')
208 legend('b_1', 'b_2', 'b_3')
209 ylim([0 1])
210
211 saveas(gcf, 'figures/q1a.noise.params.png')
212
213 % ----- %
214 %     RLS: With noise, comparing output
215 % ----- %
216
217 figure(5)
218 subplot(2,1,1)
219 plot(timespan, transpose(loudspeaker) - output)
220 title('Amplitude Difference between Loudspeaker and output signal(RLS without noise)')
221 xlabel('Seconds')
222 ylabel('Amplitude')
223 ylim([-0.1 0.15])
224 subplot(2,1,2)
225 plot(timespan, transpose(loudspeaker) - noisy_output)
226 title('Amplitude Difference between Loudspeaker and output signal(RLS with noise)')
227 xlabel('Seconds')
228 ylabel('Amplitude')
229 ylim([-0.1 0.15])
230
231 saveas(gcf, 'figures/q1a.noise.diff.png')
232
233
234
235 %%
236 % ----- %
237 %     LMS: No Noise
238 % ----- %
239
240 clear all
241 clc
242 close all
243 load('data1.mat');
244
245 Fs = 8192; %sampling freq
246 D1 = 1 * Fs; % delay 1
247 D2 = 2.5 * Fs; % delay 2
248 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
249
250 % LMS no noise with mu = 3 b2=b3 = 0
251 init_params = [1; 0; 0];
252 mu = 0.3;
253 [output, theta_hat] = LMS.function(loudspeaker, mikel, init_params, mu);
254
255 % LMS no noise with mu = 3 b2=b3 = 1
256 init_params = [1; 1; 1];
257 [output2, theta_hat2] = LMS.function(loudspeaker, mikel, init_params, mu);
258
259
260 figure(2)
261 subplot(2,1,1)
262 hold on
263 plot(timespan,theta_hat(1,:))
264 plot(timespan,theta_hat(2,:))
265 plot(timespan, theta_hat(3,:))

```

```

266 hold off
267
268 title('Parameter values (LMS no noise) mu = 0.3; b_2 = b_3 = 1')
269 xlabel('Seconds')
270 ylabel('Parameter value')
271 legend('b_1', 'b_2', 'b_3')
272 ylim([0 1.01])
273
274 subplot(2,1,2)
275 hold on
276 plot(timespan,theta.hat2(1,:))
277 plot(timespan,theta.hat2(2,:))
278 plot(timespan, theta.hat2(3,:))
279 hold off
280
281 title('Parameter values (LMS no noise) mu = 0.3; b_2 = b_3 = 1')
282 xlabel('Seconds')
283 ylabel('Parameter value')
284 legend('b_1', 'b_2', 'b_3')
285 ylim([0 1.01])
286
287 saveas(gcf,'figures/qla_lms_params.png')
288
289
290 %%
291 % ----- %
292 %      LMS:Changing Mu
293 % ----- %
294
295 clear all
296 clc
297 close all
298 load('data1.mat');
299
300 Fs = 8192; %sampling freq
301 D1 = 1 * Fs; % delay 1
302 D2 = 2.5 * Fs; % delay 2
303 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
304
305 % LMS no noise with mu = 3 b2=b3 = 0
306 init_params = [1; 0; 0];
307 mu = 0.3;
308 [output, theta.hat] = LMS_function(loudspeaker, mikel, init_params, mu);
309
310 % LMS no noise with mu = 0.03 b2=b3 = 0
311 mu = 0.03;
312 [output2, theta.hat2] = LMS_function(loudspeaker, mikel, init_params, mu);
313
314 % LMS no noise with mu = 3 b2=b3 = 0
315 mu = 3;
316 [output3, theta.hat3] = LMS_function(loudspeaker, mikel, init_params, mu);
317
318 figure(2)
319 subplot(3,1,1)
320 hold on
321 plot(timespan,theta.hat(1,:))
322 plot(timespan,theta.hat(2,:))
323 plot(timespan, theta.hat(3,:))
324 hold off
325
326 title('Parameter values (LMS no noise) mu = 0.3')
327 xlabel('Seconds')
328 ylabel('Parameter value')
329 legend('b_1', 'b_2', 'b_3')
330 ylim([0 1.01])
331
332 subplot(3,1,2)
333 hold on

```

```

334 plot(timespan,theta.hat2(1,:))
335 plot(timespan,theta.hat2(2,:))
336 plot(timespan, theta.hat2(3,:))
337 hold off
338
339 title('Parameter values (LMS no noise) mu = 0.03')
340 xlabel('Seconds')
341 ylabel('Parameter value')
342 legend('b_1', 'b_2', 'b_3')
343 ylim([0 1.01])
344
345 subplot(3,1,3)
346 hold on
347 plot(timespan,theta.hat3(1,:))
348 plot(timespan,theta.hat3(2,:))
349 plot(timespan, theta.hat3(3,:))
350 hold off
351
352 title('Parameter values (LMS no noise) mu = 3')
353 xlabel('Seconds')
354 ylabel('Parameter value')
355 legend('b_1', 'b_2', 'b_3')
356 ylim([0 1.01])
357
358 saveas(gcf,'figures/qla_lms_mu.png')
359
360
361 %% A.2
362 % ----- %
363 %     LMS with noise
364 % ----- %
365
366 clear all
367 close all
368 clc
369 load('data1.mat');
370
371 Fs = 8192; %sampling freq
372 D1 = 1 * Fs; % delay 1
373 D2 = 2.5 * Fs; % delay 2
374 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
375
376 init_params = [1; 0; 0];
377 mu = 0.3;
378
379 [output, theta.hat] = LMS.function(loudspeaker, mikel, init_params, mu);
380 [noisy_output, noisy_theta.hat] = LMS.function(loudspeaker, noisymikel, init_params, mu);
381
382 figure(4)
383 subplot(2,1,1)
384 hold on
385 plot(timespan,theta.hat(1,:))
386 plot(timespan,theta.hat(2,:))
387 plot(timespan, theta.hat(3,:))
388 hold off
389
390 title('Parameter values (LMS no noise)')
391 xlabel('Seconds')
392 ylabel('Parameter value')
393 legend('b_1', 'b_2', 'b_3')
394 ylim([0 1])
395
396 subplot(2,1,2)
397 hold on
398 plot(timespan,noisy_theta.hat(1,:))
399 plot(timespan,noisy_theta.hat(2,:))
400 plot(timespan, noisy_theta.hat(3,:))
401 hold off

```

```

402
403 title('Parameter values (LMS with noise)')
404 xlabel('Seconds')
405 ylabel('Parameter value')
406 legend('b_1', 'b_2', 'b_3')
407 ylim([0 1])
408
409 saveas(gcf, 'figures/qla.lms.noise.params.png')
410
411 % ----- %
412 %     LMS: Noise, output graphs
413 % ----- %
414
415 figure(5)
416 subplot(2,1,1)
417 plot(timespan, transpose(loudspeaker) - output)
418 title('Amplitude Difference between Loudspeaker and output signal(LMS without noise)')
419 xlabel('Seconds')
420 ylabel('Amplitude')
421 ylim([-0.1 0.15])
422 subplot(2,1,2)
423 plot(timespan, transpose(loudspeaker) - noisy_output)
424 title('Amplitude Difference between Loudspeaker and output signal(LMS with noise)')
425 xlabel('Seconds')
426 ylabel('Amplitude')
427 ylim([-0.1 0.15])
428
429 saveas(gcf, 'figures/qla.lms.noise.diff.png')

```

## A-Workshop1B

```

1  %% B.1
2  % ----- %
3  %     RLS: Time Varying. No noise
4  % ----- %
5
6  clear all
7  close all
8  clc
9  load('data1.mat');
10 load('data2.mat');
11
12 Fs = 8192; %sampling freq
13 D1 = 1 * Fs; % delay 1
14 D2 = 2.5 * Fs; % delay 2
15 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
16
17
18 % RLS with inital params = 0
19
20 init_params = [1; 0; 0];
21 P_init_scale = 1;
22 lambda = 0.995;
23
24
25 [output, theta_hat] = RLS_function2(loudspeaker, mike2, init_params, P_init_scale, lambda);
26
27 figure(1)
28 hold on
29 plot(timespan, theta_hat(1,:))
30 plot(timespan, theta_hat(2,:))
31 plot(timespan, theta_hat(3,:))
32 hold off
33
34 title('Time varying Parameter values (RLS no noise)')

```

```

35 xlabel('Seconds')
36 ylabel('Parameter value')
37 legend('b_1', 'b_2', 'b_3')
38 ylim([0 1])
39
40 saveas(gcf, 'figures/q1b_params.png')
41
42 %%
43 % ----- %
44 %      RLS: Time Varying.  Lambda changing
45 % ----- %
46 clear all
47 close all
48 clc
49 load('data1.mat');
50 load('data2.mat')
51
52 Fs = 8192; %sampling freq
53 D1 = 1 * Fs; % delay 1
54 D2 = 2.5 * Fs; % delay 2
55 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
56
57
58 % RLS with P_init_scale = 1
59 init_params = [1; 0; 0];
60 P_init_scale = 1;
61 lambda = 0.999;
62
63 [output1, theta_hat] = RLS_function2(loudspeaker, mike2, init_params, P_init_scale, lambda);
64
65 figure(2)
66 subplot(3,1,1)
67 hold on
68 plot(timespan, theta_hat(1,:))
69 plot(timespan, theta_hat(2,:))
70 plot(timespan, theta_hat(3,:))
71 hold off
72
73 title('Time varying Parameter values (RLS no noise) Lambda = 0.999')
74 xlabel('Seconds')
75 ylabel('Parameter value')
76 legend('b_1', 'b_2', 'b_3')
77 ylim([0 1.01])
78
79
80 % RLS with lambda = 0.990;
81 lambda = 0.990;
82
83 [output001, theta_hat] = RLS_function2(loudspeaker, mike2, init_params, P_init_scale, lambda);
84
85 figure(2)
86 subplot(3,1,2)
87 hold on
88 plot(timespan, theta_hat(1,:))
89 plot(timespan, theta_hat(2,:))
90 plot(timespan, theta_hat(3,:))
91 hold off
92
93 title('Time varying Parameter values (RLS no noise) lambda = 0.990;')
94 xlabel('Seconds')
95 ylabel('Parameter value')
96 legend('b_1', 'b_2', 'b_3')
97 ylim([0 1.01])
98
99 % RLS with lambda = 0.999;
100 init_params = [1; 0; 0];
101 P_init_scale = 100;
102 lambda = 0.995;

```



```

103
104 [output100, theta_hat] = RLS_function2(loudspeaker, mike2, init_params, P_init_scale, lambda);
105
106 figure(2)
107 subplot(3,1,3)
108 hold on
109 plot(timespan, theta_hat(1,:))
110 plot(timespan, theta_hat(2,:))
111 plot(timespan, theta_hat(3,:))
112 hold off
113
114 title('Time varying Parameter values (RLS no noise) lambda = 0.995')
115 xlabel('Seconds')
116 ylabel('Parameter value')
117 legend('b_1', 'b_2', 'b_3')
118 ylim([0 1.01])
119
120 saveas(gcf, 'figures/qlb.Pinit.png')
121
122 figure(6)
123 subplot(3,1,1)
124 plot(timespan, transpose(loudspeaker) - output1)
125 title('Amplitude Difference for Time Varying Echo Amplitude (RLS without noise) lambda = ...
    0.999;')
126 xlabel('Seconds')
127 ylabel('Amplitude')
128
129 subplot(3,1,2)
130 plot(timespan, transpose(loudspeaker) - output001)
131 title('Amplitude Difference for Time Varying Echo Amplitudes (RLS without noise) lambda = ...
    0.990;')
132 xlabel('Seconds')
133 ylabel('Amplitude')
134
135 subplot(3,1,3)
136 plot(timespan, transpose(loudspeaker) - output100)
137 title('Amplitude Difference for Time Varying Echo Amplitudes (RLS without noise) lambda = ...
    0.995;')
138 xlabel('Seconds')
139 ylabel('Amplitude')
140
141 saveas(gcf, 'figures/qlb.Pinit.output.diff.png')
142
143 %% B.1
144 % ----- %
145 %     RLS: Time Varying with noise
146 % ----- %
147
148 clear all
149 close all
150 clc
151 load('data1.mat');
152 load('data2.mat')
153
154 Fs = 8192; %sampling freq
155 D1 = 1 * Fs; % delay 1
156 D2 = 2.5 * Fs; % delay 2
157 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
158
159
160 init_params = [1; 0; 0];
161 P_init_scale = 1;
162 lambda = 0.995;
163
164 [output, theta_hat] = RLS_function2(loudspeaker, mike2, init_params, P_init_scale, lambda);
165 [noisy_output, noisy_theta_hat] = RLS_function2(loudspeaker, noisymike2, init_params, ...
    P_init_scale, lambda);
166

```

```

167 % sound(noisy_output)
168
169
170 figure(4)
171 subplot(2,1,1)
172 hold on
173 plot(timespan,theta_hat(1,:))
174 plot(timespan,theta_hat(2,:))
175 plot(timespan, theta_hat(3,:))
176 hold off
177
178 title('Time Varying Parameter values (RLS no noise)')
179 xlabel('Seconds')
180 ylabel('Parameter value')
181 legend('b_1', 'b_2', 'b_3')
182 ylim([0 1])
183
184 subplot(2,1,2)
185 hold on
186 plot(timespan,noisy_theta_hat(1,:))
187 plot(timespan,noisy_theta_hat(2,:))
188 plot(timespan, noisy_theta_hat(3,:))
189 hold off
190
191 title('Time varying Parameter values (RLS with noise)')
192 xlabel('Seconds')
193 ylabel('Parameter value')
194 legend('b_1', 'b_2', 'b_3')
195 ylim([0 1])
196
197 figure(5)
198 subplot(2,1,1)
199 plot(timespan, transpose(loudspeaker) - output)
200 title('Amplitude Difference between Loudspeaker and output signal for Time Varying Echo ...
    Amplitudes(RLS without noise)')
201 xlabel('Seconds')
202 ylabel('Amplitude')
203 ylim([-0.1 0.15])
204 subplot(2,1,2)
205 plot(timespan, transpose(loudspeaker) - noisy_output)
206 title('Amplitude Difference between Loudspeaker and output signalfor Time Varying Echo ...
    Amplitudes(RLS with noise)')
207 xlabel('Seconds')
208 ylabel('Amplitude')
209 ylim([-0.1 0.15])
210
211
212 %%
213 % ----- %
214 %     LMS: Time Varying
215 % ----- %
216
217 % Changing mu
218
219 clear all
220 close all
221 clc
222 load('data1.mat');
223 load('data2.mat');
224
225 Fs = 8192; %sampling freq
226 D1 = 1 * Fs; % delay 1
227 D2 = 2.5 * Fs; % delay 2
228 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis
229
230 init_params = [1; 0; 0];
231
232 % LMS no noise with mu = 3 b2=b3 = 0

```

```

233 mu = 0.3;
234 [output, theta.hat] = LMS.function(loudspeaker, mike2, init_params, mu);
235
236 % LMS no noise with mu = 0.03 b2=b3 = 0
237 mu = 0.03;
238 [output2, theta.hat2] = LMS.function(loudspeaker, mike2, init_params, mu);
239
240 % LMS no noise with mu = 3 b2=b3 = 0
241 mu = 3;
242 [output3, theta.hat3] = LMS.function(loudspeaker, mike2, init_params, mu);
243
244 figure(2)
245 subplot(3,1,1)
246 hold on
247 plot(timespan, theta.hat(1,:))
248 plot(timespan, theta.hat(2,:))
249 plot(timespan, theta.hat(3,:))
250 hold off
251
252 title('Time Varying Parameter values (LMS no noise) \mu = 0.3')
253 xlabel('Seconds')
254 ylabel('Parameter value')
255 legend('b_1', 'b_2', 'b_3')
256 ylim([0 1.01])
257
258 subplot(3,1,2)
259 hold on
260 plot(timespan, theta.hat2(1,:))
261 plot(timespan, theta.hat2(2,:))
262 plot(timespan, theta.hat2(3,:))
263 hold off
264
265 title('Time Varying Parameter values (LMS no noise) \mu = 0.03')
266 xlabel('Seconds')
267 ylabel('Parameter value')
268 legend('b_1', 'b_2', 'b_3')
269 ylim([0 1.01])
270
271 subplot(3,1,3)
272 hold on
273 plot(timespan, theta.hat3(1,:))
274 plot(timespan, theta.hat3(2,:))
275 plot(timespan, theta.hat3(3,:))
276 hold off
277
278 title('Time Varying Parameter values (LMS no noise) \mu = 3')
279 xlabel('Seconds')
280 ylabel('Parameter value')
281 legend('b_1', 'b_2', 'b_3')
282 ylim([0 1.01])
283
284 saveas(gcf, 'figures/qlb_lms_mu.png')
285
286 %%
287 % ----- %
288 %     LMS: Time Varying and Noise
289 % ----- %
290
291 clear all
292 close all
293 clc
294 load('data1.mat');
295 load('data2.mat');
296
297 Fs = 8192; %sampling freq
298 D1 = 1 * Fs; % delay 1
299 D2 = 2.5 * Fs; % delay 2
300 timespan = [0: 1/Fs: length(loudspeaker)/Fs - 1/Fs]; % Timespan for time-axis

```

```

301
302 init_params = [1; 0; 0];
303 mu = 0.3;
304
305 [output, theta_hat] = LMS_function(loudspeaker, mike2, init_params, mu);
306 [noisy_output, noisy_theta_hat] = LMS_function(loudspeaker, noisymike2, init_params, mu);
307
308 figure(4)
309 subplot(2,1,1)
310 hold on
311 plot(timespan, theta_hat(1,:))
312 plot(timespan, theta_hat(2,:))
313 plot(timespan, theta_hat(3,:))
314 hold off
315
316 title('Time Varying Parameter values (LMS no noise)')
317 xlabel('Seconds')
318 ylabel('Parameter value')
319 legend('b_1', 'b_2', 'b_3')
320 ylim([0 1])
321
322 subplot(2,1,2)
323 hold on
324 plot(timespan, noisy_theta_hat(1,:))
325 plot(timespan, noisy_theta_hat(2,:))
326 plot(timespan, noisy_theta_hat(3,:))
327 hold off
328
329 title('Time Varying Parameter values (LMS with noise)')
330 xlabel('Seconds')
331 ylabel('Parameter value')
332 legend('b_1', 'b_2', 'b_3')
333 ylim([0 1])
334
335 saveas(gcf, 'figures/q1b.lms.noise.params.png')
336
337 figure(5)
338 subplot(2,1,1)
339 plot(timespan, transpose(loudspeaker) - output)
340 title('Time Varying Params: Amplitude Difference between Loudspeaker and output signal (LMS ...
    without noise)')
341 xlabel('Seconds')
342 ylabel('Amplitude')
343 ylim([-0.1 0.15])
344 subplot(2,1,2)
345 plot(timespan, transpose(loudspeaker) - noisy_output)
346 title('Time Varying Params: Amplitude Difference between Loudspeaker and output signal (LMS ...
    with noise)')
347 xlabel('Seconds')
348 ylabel('Amplitude')
349 ylim([-0.1 0.15])
350
351 saveas(gcf, 'figures/q1b.lms.noise.diff.png')

```

## A-RLS Function 1

```

1 function [ output, theta_hat ] = RLS_function(loudspeaker, mikel, init_params, P_init_scale)
2
3 Fs = 8192; %sampling freq
4 D1 = 1 * Fs; % delay 1
5 D2 = 2.5 * Fs; % delay 2
6
7 P = P_init_scale * eye(3);
8
9 theta_hat = zeros(3, length(loudspeaker));

```

```

10 loudspeakerDelay1 = 0;
11 loudspeakerDelay2 = 0;
12
13 output = zeros(1,length(loudspeaker));
14
15
16 for i=1:length(loudspeaker)
17
18     if (i > D1)
19         loudspeakerDelay1 = loudspeaker(i - D1);
20     end
21
22     if (i > D2)
23         loudspeakerDelay2 = loudspeaker(i - D2);
24     end
25
26     phi = [loudspeaker(i); loudspeakerDelay1; loudspeakerDelay2];
27
28     P = P - (P*phi*transpose(phi)*P)/(1 + transpose(phi)*P*phi);
29
30     G = P*phi;
31
32     if ( i > 1)
33         theta_hat(:,i) = theta_hat(:,i-1) + G*(mikel(i) - transpose(phi)*theta_hat(:,i-1));
34     else
35         theta_hat(:,i) = init_params + G*(mikel(i) - transpose(phi)*init_params);
36     end
37
38     output(i) = mikel(i) - theta_hat(2,i)*loudspeakerDelay1 - ...
39                 theta_hat(3,i)*loudspeakerDelay2;
40
41 end
42 end

```

## A-RLS Function with Lambda

```

1 function [ output, theta_hat ] = RLS_function2(loudspeaker, mikel, init_params, ...
2         P_init_scale, lambda)
3
4 % ----- %
5 %     RLS function with lambda value
6 % ----- %
7
8 Fs = 8192; %sampling freq
9 D1 = 1 * Fs; % delay 1
10 D2 = 2.5 * Fs; % delay 2
11
12 P = P_init_scale * eye(3);
13
14 theta_hat = zeros(3,length(loudspeaker));
15 loudspeakerDelay1 = 0;
16 loudspeakerDelay2 = 0;
17
18 output = zeros(1,length(loudspeaker));
19
20 for i=1:length(loudspeaker)
21
22     if (i > D1)
23         loudspeakerDelay1 = loudspeaker(i - D1);
24     end
25
26     if (i > D2)
27         loudspeakerDelay2 = loudspeaker(i - D2);

```

```

28     end
29
30     phi = [loudspeaker(i); loudspeakerDelay1; loudspeakerDelay2];
31
32     P = (1/lambda)*(P - (P*phi*transpose(phi)*P)/(lambda + transpose(phi)*P*phi));
33
34     G = P*phi;
35
36     if ( i > 1)
37         theta_hat(:,i) = theta_hat(:,i-1) + G*(mikel(i) - transpose(phi)*theta_hat(:,i-1));
38     else
39         theta_hat(:,i) = init_params + G*(mikel(i) - transpose(phi)*init_params);
40     end
41
42     output(i) = mikel(i) - theta_hat(2,i)*loudspeakerDelay1 - ...
        theta_hat(3,i)*loudspeakerDelay2;
43 end
44
45
46 end

```

## A-LMS Function

```

1  function [ output, theta_hat ] = LMS_function(loudspeaker, mikel, init_params, mu)
2
3  % ----- %
4  %     LMS: function
5  % ----- %
6
7  Fs = 8192; %sampling freq
8  D1 = 1 * Fs; % delay 1
9  D2 = 2.5 * Fs; % delay 2
10
11  theta_hat = zeros(3,length(loudspeaker));
12  loudspeakerDelay1 = 0;
13  loudspeakerDelay2 = 0;
14
15  output = zeros(1,length(loudspeaker));
16
17
18  for i=1:length(loudspeaker)
19
20      if (i > D1)
21          loudspeakerDelay1 = loudspeaker(i - D1);
22      end
23
24      if (i > D2)
25          loudspeakerDelay2 = loudspeaker(i - D2);
26      end
27
28      phi = [loudspeaker(i); loudspeakerDelay1; loudspeakerDelay2];
29
30      G = mu*phi;
31
32      if ( i > 1)
33          theta_hat(:,i) = theta_hat(:,i-1) + G*(mikel(i) - transpose(phi)*theta_hat(:,i-1));
34      else
35          theta_hat(:,i) = init_params + G*(mikel(i) - transpose(phi)*init_params);
36      end
37
38      output(i) = mikel(i) - theta_hat(2,i)*loudspeakerDelay1 - ...
          theta_hat(3,i)*loudspeakerDelay2;
39  end
40
41

```

## B- Gain Estimate

```

1  #include "SP2WS1.h"
2
3
4  // input signal history
5  float insignal1[100], insignal2[100];
6
7
8
9  // function prototypes
10 void gainestimateLMS(float, float, float, float[2]);
11 void gainestimateRLS(float, float, float, float[2]);
12
13
14 void gainestimate(float in1, float in2, float out, float gain[2])
15 {
16     // record input signal history for checking signal magnitude using plot facility
17     for (int i = 99; i > 0; i--)
18     {
19         insignal1[i] = insignal1[i-1];
20         insignal2[i] = insignal2[i-1];
21     }
22     insignal1[0] = in1;
23     insignal2[0] = in2;
24
25
26     // estimate gain
27     gain[0] = 0;
28     gain[1] = 1;
29     // gainestimateLMS(in1, in2, out, gain);
30     gainestimateRLS(in1, in2, out, gain);
31 }
32
33 void gainestimateLMS(float in1, float in2, float out, float gain[2])
34 {
35     // TODO: Implement gain estimation using LMS algorithm
36     static float theta_hat1 = 0, theta_hat2 = 1.0;
37
38     float mu = 1E-20;
39
40     float err = out - (in1 * theta_hat1 + in2 * theta_hat2);
41
42     theta_hat1 = theta_hat1 + (mu * in1) * err;
43     theta_hat2 = theta_hat2 + (mu * in2) * err;
44
45     gain[0] = theta_hat1;
46     gain[1] = theta_hat2;
47
48 }
49
50 void gainestimateRLS(float in1, float in2, float out, float gain[2])
51 {
52     // TODO: Implement gain estimation using RLS algorithm
53     static float theta_hat1 = 0, theta_hat2 = 1.0, p11 = 1E-25, p12 = 0, p21 = 0, p22 = ...
54         1E-25; // p11 = p22 = 1 for identity
55
56     float lambda = 0.9999;
57
58     float err = out - (in1 * theta_hat1 + in2 * theta_hat2);
59
60     float p11_in1 = p11 * in1;

```

```

61     float p11_in2 = p11 * in2;
62     float p12_in1 = p12 * in1;
63     float p12_in2 = p12 * in2;
64     float p21_in1 = p21 * in1;
65     float p21_in2 = p21 * in2;
66     float p22_in1 = p22 * in1;
67     float p22_in2 = p22 * in2;
68
69     float inv_lambda = 1.0 / lambda;
70
71     float denom = (lambda + in1 * (p11_in1 + p21_in2) + in2 * (p12_in1 + p22_in2));
72
73     float p11_new = inv_lambda * (p11 - (p11_in1 * (p11_in1 + p12_in2) + p21_in2 * ...
74         (p11_in1 + p12_in2)) / denom);
75     float p12_new = inv_lambda * (p12 - (p12_in1 * (p11_in1 + p12_in2) + p22_in2 * ...
76         (p11_in1 + p12_in2)) / denom);
77     float p21_new = inv_lambda * (p21 - (p11_in1 * (p21_in1 + p22_in2) + p21_in2 * ...
78         (p21_in1 + p22_in2)) / denom);
79     float p22_new = inv_lambda * (p22 - (p12_in1 * (p21_in1 + p22_in2) + p22_in2 * ...
80         (p21_in1 + p22_in2)) / denom);
81
82     p11 = p11_new;
83     p12 = p12_new;
84     p21 = p21_new;
85     p22 = p22_new;
86
87     theta_hat1 = theta_hat1 + ((p11 * in1) + (p12 * in2)) * err;
88     theta_hat2 = theta_hat2 + ((p21 * in1) + (p22 * in2)) * err;
89
90     gain[0] = theta_hat1;
91     gain[1] = theta_hat2;
92     // then save p values for next round
93 }

```