# lab06.Rmd

*Burhan A Hanif*

*March 15, 2019*

Load the Boston Housing data and create the vector $y$ and the design matrix $X$.

```
data(Boston, package = "MASS")
y = Boston$medv
intecp = rep(1, nrow(Boston))
X = as.matrix(cbind(intecp, Boston[, 1 : 13]))
```

Find the OLS estimate and OLS predictions without using `lm`.

```
b = solve(t(X)%*% X) %*% t(X) %*% y
b
```

```
##                  [,1]
## intecp   3.645949e+01
## crim    -1.080114e-01
## zn       4.642046e-02
## indus    2.055863e-02
## chas     2.686734e+00
## nox     -1.776661e+01
## rm       3.809865e+00
## age      6.922246e-04
## dis     -1.475567e+00
## rad      3.060495e-01
## tax     -1.233459e-02
## ptratio -9.527472e-01
## black    9.311683e-03
## lstat   -5.247584e-01
```

```
y_hat = X %*% b
y_hat
```

```
##           [,1]
## 1   30.0038434
## 2   25.0255624
## 3   30.5675967
## 4   28.6070365
## 5   27.9435242
## 6   25.2562845
## 7   23.0018083
## 8   19.5359884
## 9   11.5236369
## 10  18.9202621
## 11  18.9994965
## 12  21.5867957
## 13  20.9065215
## 14  19.5529028
## 15  19.2834821
## 16  19.2974832
## 17  20.5275098
```

```
## 18  16.9114013
## 19  16.1780111
## 20  18.4061360
## 21  12.5238575
## 22  17.6710367
## 23  15.8328813
## 24  13.8062853
## 25  15.6783383
## 26  13.3866856
## 27  15.4639765
## 28  14.7084743
## 29  19.5473729
## 30  20.8764282
## 31  11.4551176
## 32  18.0592329
## 33   8.8110574
## 34  14.2827581
## 35  13.7067589
## 36  23.8146353
## 37  22.3419371
## 38  23.1089114
## 39  22.9150261
## 40  31.3576257
## 41  34.2151023
## 42  28.0205641
## 43  25.2038663
## 44  24.6097927
## 45  22.9414918
## 46  22.0966982
## 47  20.4232003
## 48  18.0365509
## 49   9.1065538
## 50  17.2060775
## 51  21.2815254
## 52  23.9722228
## 53  27.6558508
## 54  24.0490181
## 55  15.3618477
## 56  31.1526495
## 57  24.8568698
## 58  33.1091981
## 59  21.7753799
## 60  21.0849356
## 61  17.8725804
## 62  18.5111021
## 63  23.9874286
## 64  22.5540887
## 65  23.3730864
## 66  30.3614836
## 67  25.5305651
## 68  21.1133856
## 69  17.4215379
## 70  20.7848363
## 71  25.2014886
```

```
## 72  21.7426577
## 73  24.5574496
## 74  24.0429571
## 75  25.5049972
## 76  23.9669302
## 77  22.9454540
## 78  23.3569982
## 79  21.2619827
## 80  22.4281737
## 81  28.4057697
## 82  26.9948609
## 83  26.0357630
## 84  25.0587348
## 85  24.7845667
## 86  27.7904920
## 87  22.1685342
## 88  25.8927642
## 89  30.6746183
## 90  30.8311062
## 91  27.1190194
## 92  27.4126673
## 93  28.9412276
## 94  29.0810555
## 95  27.0397736
## 96  28.6245995
## 97  24.7274498
## 98  35.7815952
## 99  35.1145459
## 100 32.2510280
## 101 24.5802202
## 102 25.5941347
## 103 19.7901368
## 104 20.3116713
## 105 21.4348259
## 106 18.5399401
## 107 17.1875599
## 108 20.7504903
## 109 22.6482911
## 110 19.7720367
## 111 20.6496586
## 112 26.5258674
## 113 20.7732364
## 114 20.7154831
## 115 25.1720888
## 116 20.4302559
## 117 23.3772463
## 118 23.6904326
## 119 20.3357836
## 120 20.7918087
## 121 21.9163207
## 122 22.4710778
## 123 20.5573856
## 124 16.3666198
## 125 20.5609982
```

```
## 126 22.4817845
## 127 14.6170663
## 128 15.1787668
## 129 18.9386859
## 130 14.0557329
## 131 20.0352740
## 132 19.4101340
## 133 20.0619157
## 134 15.7580767
## 135 13.2564524
## 136 17.2627773
## 137 15.8784188
## 138 19.3616395
## 139 13.8148390
## 140 16.4488147
## 141 13.5714193
## 142  3.9888551
## 143 14.5949548
## 144 12.1488148
## 145  8.7282236
## 146 12.0358534
## 147 15.8208206
## 148  8.5149902
## 149  9.7184414
## 150 14.8045137
## 151 20.8385815
## 152 18.3010117
## 153 20.1228256
## 154 17.2860189
## 155 22.3660023
## 156 20.1037592
## 157 13.6212589
## 158 33.2598270
## 159 29.0301727
## 160 25.5675277
## 161 32.7082767
## 162 36.7746701
## 163 40.5576584
## 164 41.8472817
## 165 24.7886738
## 166 25.3788924
## 167 37.2034745
## 168 23.0874875
## 169 26.4027396
## 170 26.6538211
## 171 22.5551466
## 172 24.2908281
## 173 22.9765722
## 174 29.0719431
## 175 26.5219434
## 176 30.7220906
## 177 25.6166931
## 178 29.1374098
## 179 31.4357197
```

```
## 180 32.9223157
## 181 34.7244046
## 182 27.7655211
## 183 33.8878732
## 184 30.9923804
## 185 22.7182001
## 186 24.7664781
## 187 35.8849723
## 188 33.4247672
## 189 32.4119915
## 190 34.5150995
## 191 30.7610949
## 192 30.2893414
## 193 32.9191871
## 194 32.1126077
## 195 31.5587100
## 196 40.8455572
## 197 36.1277008
## 198 32.6692081
## 199 34.7046912
## 200 30.0934516
## 201 30.6439391
## 202 29.2871950
## 203 37.0714839
## 204 42.0319312
## 205 43.1894984
## 206 22.6903480
## 207 23.6828471
## 208 17.8544721
## 209 23.4942899
## 210 17.0058772
## 211 22.3925110
## 212 17.0604275
## 213 22.7389292
## 214 25.2194255
## 215 11.1191674
## 216 24.5104915
## 217 26.6033477
## 218 28.3551871
## 219 24.9152546
## 220 29.6865277
## 221 33.1841975
## 222 23.7745666
## 223 32.1405196
## 224 29.7458199
## 225 38.3710245
## 226 39.8146187
## 227 37.5860575
## 228 32.3995325
## 229 35.4566524
## 230 31.2341151
## 231 24.4844923
## 232 33.2883729
## 233 38.0481048
```

```
## 234 37.1632863
## 235 31.7138352
## 236 25.2670557
## 237 30.1001074
## 238 32.7198716
## 239 28.4271706
## 240 28.4294068
## 241 27.2937594
## 242 23.7426248
## 243 24.1200789
## 244 27.4020841
## 245 16.3285756
## 246 13.3989126
## 247 20.0163878
## 248 19.8618443
## 249 21.2883131
## 250 24.0798915
## 251 24.2063355
## 252 25.0421582
## 253 24.9196401
## 254 29.9456337
## 255 23.9722832
## 256 21.6958089
## 257 37.5110924
## 258 43.3023904
## 259 36.4836142
## 260 34.9898859
## 261 34.8121151
## 262 37.1663133
## 263 40.9892850
## 264 34.4463409
## 265 35.8339755
## 266 28.2457430
## 267 31.2267359
## 268 40.8395575
## 269 39.3179239
## 270 25.7081791
## 271 22.3029553
## 272 27.2034097
## 273 28.5116947
## 274 35.4767660
## 275 36.1063916
## 276 33.7966827
## 277 35.6108586
## 278 34.8399338
## 279 30.3519266
## 280 35.3098070
## 281 38.7975697
## 282 34.3312319
## 283 40.3396307
## 284 44.6730834
## 285 31.5968909
## 286 27.3565923
## 287 20.1017415
```

```
## 288 27.0420667
## 289 27.2136458
## 290 26.9139584
## 291 33.4356331
## 292 34.4034963
## 293 31.8333982
## 294 25.8178324
## 295 24.4298235
## 296 28.4576434
## 297 27.3626700
## 298 19.5392876
## 299 29.1130984
## 300 31.9105461
## 301 30.7715945
## 302 28.9427587
## 303 28.8819102
## 304 32.7988723
## 305 33.2090546
## 306 30.7683179
## 307 35.5622686
## 308 32.7090512
## 309 28.6424424
## 310 23.5896583
## 311 18.5426690
## 312 26.8788984
## 313 23.2813398
## 314 25.5458025
## 315 25.4812006
## 316 20.5390990
## 317 17.6157257
## 318 18.3758169
## 319 24.2907028
## 320 21.3252904
## 321 24.8868224
## 322 24.8693728
## 323 22.8695245
## 324 19.4512379
## 325 25.1178340
## 326 24.6678691
## 327 23.6807618
## 328 19.3408962
## 329 21.1741811
## 330 24.2524907
## 331 21.5926089
## 332 19.9844661
## 333 23.3388800
## 334 22.1406069
## 335 21.5550993
## 336 20.6187291
## 337 20.1609718
## 338 19.2849039
## 339 22.1667232
## 340 21.2496577
## 341 21.4293931
```

```
## 342 30.3278880
## 343 22.0473498
## 344 27.7064791
## 345 28.5479412
## 346 16.5450112
## 347 14.7835964
## 348 25.2738008
## 349 27.5420512
## 350 22.1483756
## 351 20.4594409
## 352 20.5460542
## 353 16.8806383
## 354 25.4025351
## 355 14.3248663
## 356 16.5948846
## 357 19.6370469
## 358 22.7180661
## 359 22.2021889
## 360 19.2054806
## 361 22.6661611
## 362 18.9319262
## 363 18.2284680
## 364 20.2315081
## 365 37.4944739
## 366 14.2819073
## 367 15.5428625
## 368 10.8316232
## 369 23.8007290
## 370 32.6440736
## 371 34.6068404
## 372 24.9433133
## 373 25.9998091
## 374  6.1263250
## 375  0.7777981
## 376 25.3071306
## 377 17.7406106
## 378 20.2327441
## 379 15.8333130
## 380 16.8351259
## 381 14.3699483
## 382 18.4768283
## 383 13.4276828
## 384 13.0617751
## 385  3.2791812
## 386  8.0602217
## 387  6.1284220
## 388  5.6186481
## 389  6.4519857
## 390 14.2076474
## 391 17.2122518
## 392 17.2988727
## 393  9.8911664
## 394 20.2212419
## 395 17.9418118
```

```
## 396 20.3044578
## 397 19.2955908
## 398 16.3363278
## 399  6.5516232
## 400 10.8901678
## 401 11.8814587
## 402 17.8117451
## 403 18.2612659
## 404 12.9794878
## 405  7.3781636
## 406  8.2111586
## 407  8.0662619
## 408 19.9829479
## 409 13.7075637
## 410 19.8526845
## 411 15.2230830
## 412 16.9607198
## 413  1.7185181
## 414 11.8057839
## 415 -4.2813107
## 416  9.5837674
## 417 13.3666081
## 418  6.8956236
## 419  6.1477985
## 420 14.6066179
## 421 19.6000267
## 422 18.1242748
## 423 18.5217713
## 424 13.1752861
## 425 14.6261762
## 426  9.9237498
## 427 16.3459065
## 428 14.0751943
## 429 14.2575624
## 430 13.0423479
## 431 18.1595569
## 432 18.6955435
## 433 21.5272830
## 434 17.0314186
## 435 15.9609044
## 436 13.3614161
## 437 14.5207938
## 438  8.8197601
## 439  4.8675110
## 440 13.0659131
## 441 12.7060970
## 442 17.2955806
## 443 18.7404850
## 444 18.0590103
## 445 11.5147468
## 446 11.9740036
## 447 17.6834462
## 448 18.1269524
## 449 17.5183465
```

```
## 450 17.2274251
## 451 16.5227163
## 452 19.4129110
## 453 18.5821524
## 454 22.4894479
## 455 15.2800013
## 456 15.8208934
## 457 12.6872558
## 458 12.8763379
## 459 17.1866853
## 460 18.5124761
## 461 19.0486053
## 462 20.1720893
## 463 19.7740732
## 464 22.4294077
## 465 20.3191185
## 466 17.8861625
## 467 14.3747852
## 468 16.9477685
## 469 16.9840576
## 470 18.5883840
## 471 20.1671944
## 472 22.9771803
## 473 22.4558073
## 474 25.5782463
## 475 16.3914763
## 476 16.1114628
## 477 20.5348160
## 478 11.5427274
## 479 19.2049630
## 480 21.8627639
## 481 23.4687887
## 482 27.0988732
## 483 28.5699430
## 484 21.0839878
## 485 19.4551620
## 486 22.2222591
## 487 19.6559196
## 488 21.3253610
## 489 11.8558372
## 490  8.2238669
## 491  3.6639967
## 492 13.7590854
## 493 15.9311855
## 494 20.6266205
## 495 20.6124941
## 496 16.8854196
## 497 14.0132079
## 498 19.1085414
## 499 21.2980517
## 500 18.4549884
## 501 20.4687085
## 502 23.5333405
## 503 22.3757189
```

```
## 504 27.6274261
## 505 26.1279668
## 506 22.3442123
```

Write a function spec'd as follows:

```r
#' Orthogonal Projection
#'
#' Projects vector a onto v.
#'
#' @param a   the vector to project
#' @param v   the vector projected onto
#'
#' @returns   a list of two vectors, the orthogonal projection parallel to v named a_parallel,
#'            and the orthogonal error orthogonal to v called a_perpendicular
orthogonal_projection = function(a, v){
  a_parrellel = ((v %*% t(v) %*% a) / sum(v^2))
  a_perpendecular = a - a_parrellel
  list("a_parallel" = a_parrellel , "a_perpendicular" = a_perpendecular)

}

orthogonal_projection(c(1,2,3,4), c(1,2,3,4))
```

```
## $a_parallel
##      [,1]
## [1,]    1
## [2,]    2
## [3,]    3
## [4,]    4
##
## $a_perpendicular
##      [,1]
## [1,]    0
## [2,]    0
## [3,]    0
## [4,]    0
```

```r
orthogonal_projection(c(1,2,3,4) , c(0,2,0 ,-1))
```

```
## $a_parallel
##      [,1]
## [1,]    0
## [2,]    0
## [3,]    0
## [4,]    0
##
## $a_perpendicular
##      [,1]
## [1,]    1
## [2,]    2
## [3,]    3
## [4,]    4
```

```r
result = orthogonal_projection(c(2,6,7,3), c(1,3,5,7))
t(result$a_parallel) %*% result$a_perpendicular
```

```
##                [,1]
## [1,] 7.105427e-15
```

```
result$a_parallel + result$a_perpendicular
```

```
##      [,1]
## [1,]    2
## [2,]    6
## [3,]    7
## [4,]    3
```

```
result$a_parallel / c(1,3,5,7)
```

```
##           [,1]
## [1,] 0.9047619
## [2,] 0.9047619
## [3,] 0.9047619
## [4,] 0.9047619
```

Try to project onto the column space of $X$ by projecting on each vector of $X$ individually and adding up the projections. You can use the function `orthogonal_projection`.

```
sumOrthProj = rep(0 , nrow(X))
for (j in 1 : ncol(X)){
  sumOrthProj = sumOrthProj + orthogonal_projection(y , X[ , j]) $a_parallel
}
sumOrthProj
```

```
##             [,1]
##   [1,] 177.3425
##   [2,] 185.6013
##   [3,] 177.7175
##   [4,] 171.7247
##   [5,] 177.3255
##   [6,] 175.5639
##   [7,] 199.9166
##   [8,] 217.8005
##   [9,] 228.9885
##  [10,] 214.6665
##  [11,] 221.2131
##  [12,] 208.4377
##  [13,] 195.0379
##  [14,] 190.2555
##  [15,] 196.8100
##  [16,] 187.5918
##  [17,] 178.3824
##  [18,] 200.1527
##  [19,] 174.7700
##  [20,] 190.1822
##  [21,] 207.5072
##  [22,] 200.3113
##  [23,] 207.4244
##  [24,] 209.9919
##  [25,] 206.1339
##  [26,] 197.7069
##  [27,] 203.2936
##  [28,] 201.2678
```

```
## [29,] 204.2573
## [30,] 200.7458
## [31,] 209.7592
## [32,] 202.7670
## [33,] 204.3266
## [34,] 202.9708
## [35,] 200.5077
## [36,] 180.1832
## [37,] 178.9134
## [38,] 174.5852
## [39,] 172.9824
## [40,] 200.2527
## [41,] 197.6158
## [42,] 164.8309
## [43,] 164.5818
## [44,] 167.2289
## [45,] 177.5743
## [46,] 172.7788
## [47,] 177.4580
## [48,] 199.9197
## [49,] 214.8100
## [50,] 191.4060
## [51,] 197.3595
## [52,] 197.8394
## [53,] 183.7722
## [54,] 185.5095
## [55,] 238.9040
## [56,] 223.9247
## [57,] 224.8146
## [58,] 227.7115
## [59,] 202.2867
## [60,] 204.9981
## [61,] 215.0609
## [62,] 221.5148
## [63,] 210.9784
## [64,] 212.3270
## [65,] 202.2265
## [66,] 208.4783
## [67,] 216.4008
## [68,] 185.2580
## [69,] 193.9662
## [70,] 189.1984
## [71,] 173.8982
## [72,] 178.2755
## [73,] 172.0245
## [74,] 173.7446
## [75,] 177.0576
## [76,] 190.3045
## [77,] 198.7091
## [78,] 189.7211
## [79,] 198.8416
## [80,] 187.5456
## [81,] 185.7750
## [82,] 197.1954
```

```
##  [83,] 185.4817
##  [84,] 189.3117
##  [85,] 175.0620
##  [86,] 172.6920
##  [87,] 174.8218
##  [88,] 170.0319
##  [89,] 175.4574
##  [90,] 169.7422
##  [91,] 169.8164
##  [92,] 171.2225
##  [93,] 199.4326
##  [94,] 189.9640
##  [95,] 207.5727
##  [96,] 164.0186
##  [97,] 172.7763
##  [98,] 173.7402
##  [99,] 161.5778
## [100,] 169.9878
## [101,] 193.6994
## [102,] 190.0724
## [103,] 175.1964
## [104,] 197.7476
## [105,] 195.7364
## [106,] 199.5654
## [107,] 201.3244
## [108,] 194.5555
## [109,] 198.8687
## [110,] 200.4707
## [111,] 188.9506
## [112,] 196.9077
## [113,] 202.1632
## [114,] 205.5747
## [115,] 193.8179
## [116,] 197.9584
## [117,] 194.7796
## [118,] 195.0913
## [119,] 192.9965
## [120,] 193.0285
## [121,] 200.9191
## [122,] 204.0994
## [123,] 209.8765
## [124,] 217.8392
## [125,] 209.6441
## [126,] 205.2995
## [127,] 217.2223
## [128,] 220.6751
## [129,] 222.5382
## [130,] 222.9950
## [131,] 220.8952
## [132,] 221.0594
## [133,] 219.2358
## [134,] 221.6666
## [135,] 216.8342
## [136,] 225.2208
```

```
## [137,] 220.6441
## [138,] 221.6408
## [139,] 225.9042
## [140,] 224.1609
## [141,] 228.5457
## [142,] 238.0093
## [143,] 258.0364
## [144,] 230.2714
## [145,] 229.7619
## [146,] 219.5259
## [147,] 205.2940
## [148,] 229.5545
## [149,] 226.8236
## [150,] 220.8075
## [151,] 216.2096
## [152,] 211.2432
## [153,] 233.7632
## [154,] 210.4244
## [155,] 242.7886
## [156,] 226.1519
## [157,] 198.1168
## [158,] 199.0025
## [159,] 197.4570
## [160,] 210.8628
## [161,] 222.8134
## [162,] 197.3128
## [163,] 230.5425
## [164,] 233.4005
## [165,] 206.6117
## [166,] 196.0024
## [167,] 202.9185
## [168,] 193.4188
## [169,] 201.3122
## [170,] 204.5138
## [171,] 203.6622
## [172,] 205.5932
## [173,] 181.8463
## [174,] 177.5575
## [175,] 172.2604
## [176,] 162.4670
## [177,] 171.7943
## [178,] 174.4877
## [179,] 175.2607
## [180,] 161.1445
## [181,] 172.9028
## [182,] 162.9378
## [183,] 169.6778
## [184,] 170.2545
## [185,] 174.8399
## [186,] 171.5279
## [187,] 163.8397
## [188,] 191.8635
## [189,] 188.5807
## [190,] 195.2309
```

```
## [191,] 197.4695
## [192,] 199.4115
## [193,] 197.8452
## [194,] 189.1495
## [195,] 189.0224
## [196,] 202.8928
## [197,] 210.2695
## [198,] 212.7445
## [199,] 213.8893
## [200,] 223.1467
## [201,] 222.4576
## [202,] 211.4742
## [203,] 206.0255
## [204,] 210.5342
## [205,] 209.6636
## [206,] 175.8137
## [207,] 187.3693
## [208,] 198.3443
## [209,] 219.2908
## [210,] 236.2826
## [211,] 229.5539
## [212,] 233.3514
## [213,] 216.3153
## [214,] 177.8674
## [215,] 187.1588
## [216,] 180.4386
## [217,] 216.8047
## [218,] 195.9982
## [219,] 231.1663
## [220,] 226.0956
## [221,] 218.2176
## [222,] 230.4053
## [223,] 217.3990
## [224,] 186.6023
## [225,] 185.2547
## [226,] 188.6566
## [227,] 187.1807
## [228,] 184.9405
## [229,] 168.5670
## [230,] 165.6617
## [231,] 186.0929
## [232,] 186.2798
## [233,] 187.1694
## [234,] 186.2726
## [235,] 211.6461
## [236,] 183.6726
## [237,] 219.7629
## [238,] 187.3844
## [239,] 187.7249
## [240,] 195.8352
## [241,] 205.7590
## [242,] 207.0225
## [243,] 205.4562
## [244,] 187.0661
```

```
## [245,] 217.4666
## [246,] 223.5502
## [247,] 206.2827
## [248,] 218.5550
## [249,] 209.4659
## [250,] 200.0404
## [251,] 195.3339
## [252,] 190.3466
## [253,] 199.3440
## [254,] 205.3644
## [255,] 222.1666
## [256,] 221.0595
## [257,] 213.2367
## [258,] 187.4793
## [259,] 189.0053
## [260,] 187.2961
## [261,] 187.3086
## [262,] 187.7128
## [263,] 190.5718
## [264,] 193.0163
## [265,] 187.0617
## [266,] 176.7658
## [267,] 192.8986
## [268,] 183.2921
## [269,] 174.1655
## [270,] 218.1544
## [271,] 186.0494
## [272,] 173.8860
## [273,] 184.8274
## [274,] 216.3309
## [275,] 213.2610
## [276,] 188.0972
## [277,] 225.1553
## [278,] 216.3784
## [279,] 188.0233
## [280,] 168.7560
## [281,] 181.8882
## [282,] 175.4536
## [283,] 206.8150
## [284,] 229.4395
## [285,] 216.9399
## [286,] 199.9286
## [287,] 222.0373
## [288,] 207.9297
## [289,] 212.6062
## [290,] 208.1333
## [291,] 206.7195
## [292,] 207.9990
## [293,] 206.2275
## [294,] 180.3287
## [295,] 188.2384
## [296,] 185.3203
## [297,] 191.0578
## [298,] 201.6367
```

```
## [299,] 207.8678
## [300,] 207.6902
## [301,] 219.5948
## [302,] 200.4893
## [303,] 192.7012
## [304,] 190.4435
## [305,] 186.9611
## [306,] 188.2722
## [307,] 191.0158
## [308,] 190.0831
## [309,] 186.6019
## [310,] 187.5880
## [311,] 172.4100
## [312,] 175.4402
## [313,] 192.0128
## [314,] 188.4946
## [315,] 194.1573
## [316,] 192.5907
## [317,] 202.4118
## [318,] 196.7135
## [319,] 189.0506
## [320,] 190.8713
## [321,] 183.1577
## [322,] 183.1400
## [323,] 182.6539
## [324,] 192.0081
## [325,] 179.6587
## [326,] 174.9050
## [327,] 179.6762
## [328,] 190.1545
## [329,] 174.8901
## [330,] 170.9282
## [331,] 178.8968
## [332,] 196.6549
## [333,] 189.3779
## [334,] 181.9707
## [335,] 183.2280
## [336,] 180.7816
## [337,] 181.6643
## [338,] 187.8290
## [339,] 176.4815
## [340,] 179.7855
## [341,] 182.6749
## [342,] 193.6978
## [343,] 185.8772
## [344,] 214.3870
## [345,] 207.6686
## [346,] 194.1855
## [347,] 195.8920
## [348,] 230.1281
## [349,] 220.5068
## [350,] 206.0648
## [351,] 207.5866
## [352,] 227.5206
```

```
## [353,] 224.2947
## [354,] 239.7144
## [355,] 234.7533
## [356,] 231.9577
## [357,] 289.1578
## [358,] 282.9550
## [359,] 280.0488
## [360,] 250.4737
## [361,] 247.0233
## [362,] 251.5371
## [363,] 245.9809
## [364,] 276.7886
## [365,] 273.2703
## [366,] 228.3770
## [367,] 239.9344
## [368,] 227.4195
## [369,] 229.2493
## [370,] 264.1868
## [371,] 265.5169
## [372,] 242.0099
## [373,] 265.2474
## [374,] 269.9409
## [375,] 274.8646
## [376,] 260.0072
## [377,] 262.8997
## [378,] 261.5842
## [379,] 270.3467
## [380,] 265.0600
## [381,] 302.4003
## [382,] 264.8824
## [383,] 261.7359
## [384,] 261.8298
## [385,] 261.9130
## [386,] 272.0932
## [387,] 272.0982
## [388,] 273.9613
## [389,] 268.7954
## [390,] 257.9307
## [391,] 254.5188
## [392,] 252.9816
## [393,] 263.7590
## [394,] 253.1586
## [395,] 256.6291
## [396,] 257.4223
## [397,] 257.3733
## [398,] 256.9610
## [399,] 285.6157
## [400,] 260.3627
## [401,] 275.9831
## [402,] 263.6605
## [403,] 260.1970
## [404,] 265.1475
## [405,] 276.7342
## [406,] 294.1506
```

```
## [407,] 257.9665
## [408,] 243.8120
## [409,] 254.0643
## [410,] 247.5619
## [411,] 243.0513
## [412,] 239.6817
## [413,] 249.5482
## [414,] 252.5387
## [415,] 275.5651
## [416,] 254.0810
## [417,] 244.6125
## [418,] 254.2095
## [419,] 274.4554
## [420,] 241.1360
## [421,] 253.6314
## [422,] 249.4523
## [423,] 241.9776
## [424,] 232.9775
## [425,] 220.4736
## [426,] 243.5333
## [427,] 219.9358
## [428,] 242.3705
## [429,] 237.4656
## [430,] 244.7323
## [431,] 232.6273
## [432,] 239.8181
## [433,] 224.0374
## [434,] 237.2577
## [435,] 241.5633
## [436,] 251.5874
## [437,] 241.0389
## [438,] 250.0043
## [439,] 256.8508
## [440,] 262.4084
## [441,] 269.1437
## [442,] 263.0669
## [443,] 257.7711
## [444,] 263.1145
## [445,] 257.7443
## [446,] 246.7757
## [447,] 254.5338
## [448,] 259.7985
## [449,] 261.4657
## [450,] 256.4237
## [451,] 235.5824
## [452,] 258.2268
## [453,] 256.4019
## [454,] 263.4725
## [455,] 240.1774
## [456,] 236.4256
## [457,] 233.9410
## [458,] 232.0570
## [459,] 249.7345
## [460,] 254.1789
```

```
## [461,] 249.4194
## [462,] 253.3818
## [463,] 253.8711
## [464,] 251.9003
## [465,] 247.5891
## [466,] 236.3613
## [467,] 230.3284
## [468,] 252.8795
## [469,] 253.2550
## [470,] 244.7872
## [471,] 251.2661
## [472,] 247.6588
## [473,] 246.3856
## [474,] 242.4031
## [475,] 250.3428
## [476,] 255.1408
## [477,] 255.6807
## [478,] 261.4172
## [479,] 256.1483
## [480,] 250.0608
## [481,] 241.1623
## [482,] 241.5936
## [483,] 242.9918
## [484,] 233.8535
## [485,] 236.4484
## [486,] 240.7369
## [487,] 251.5929
## [488,] 237.1953
## [489,] 237.7785
## [490,] 242.3128
## [491,] 246.2328
## [492,] 241.2246
## [493,] 233.4760
## [494,] 187.6066
## [495,] 187.2372
## [496,] 188.8847
## [497,] 203.7429
## [498,] 197.1522
## [499,] 192.9157
## [500,] 195.7297
## [501,] 198.7412
## [502,] 186.1883
## [503,] 185.1754
## [504,] 187.6447
## [505,] 188.3535
## [506,] 185.6256
```

How much double counting occurred? Measure the magnitude relative to the true LS orthogonal projection.

```
d = sumOrthProj / y_hat
d
```

```
##           [,1]
## 1     5.910661
## 2     7.416470
```

```
## 3      5.813919
## 4      6.002884
## 5      6.345853
## 6      6.951296
## 7      8.691341
## 8     11.148683
## 9     19.871200
## 10    11.345854
## 11    11.643105
## 12     9.655794
## 13     9.329048
## 14     9.730293
## 15    10.206143
## 16     9.721051
## 17     8.689919
## 18    11.835372
## 19    10.802933
## 20    10.332542
## 21    16.568954
## 22    11.335570
## 23    13.100863
## 24    15.209876
## 25    13.147691
## 26    14.768919
## 27    13.146267
## 28    13.683798
## 29    10.449348
## 30     9.615908
## 31    18.311396
## 32    11.227886
## 33    23.189793
## 34    14.210895
## 35    14.628384
## 36     7.566072
## 37     8.007963
## 38     7.554885
## 39     7.548862
## 40     6.386094
## 41     5.775690
## 42     5.882496
## 43     6.530023
## 44     6.795216
## 45     7.740312
## 46     7.819215
## 47     8.689039
## 48    11.084142
## 49    23.588503
## 50    11.124324
## 51     9.273747
## 52     8.252859
## 53     6.644966
## 54     7.713807
## 55    15.551774
## 56     7.187983
```

```
## 57    9.044364
## 58    6.877590
## 59    9.289698
## 60    9.722491
## 61   12.033009
## 62   11.966590
## 63    8.795372
## 64    9.414125
## 65    8.652110
## 66    6.866540
## 67    8.476144
## 68    8.774432
## 69   11.133699
## 70    9.102714
## 71    6.900316
## 72    8.199341
## 73    7.004984
## 74    7.226426
## 75    6.942073
## 76    7.940294
## 77    8.660062
## 78    8.122664
## 79    9.351979
## 80    8.362056
## 81    6.540045
## 82    7.304924
## 83    7.124111
## 84    7.554720
## 85    7.063346
## 86    6.214067
## 87    7.886034
## 88    6.566771
## 89    5.719955
## 90    5.505551
## 91    6.261892
## 92    6.246109
## 93    6.890952
## 94    6.532225
## 95    7.676570
## 96    5.729987
## 97    6.987225
## 98    4.855574
## 99    4.601449
## 100   5.270771
## 101   7.880296
## 102   7.426405
## 103   8.852713
## 104   9.735666
## 105   9.131698
## 106  10.764080
## 107  11.713379
## 108   9.375948
## 109   8.780735
## 110  10.139101
```

```
## 111   9.150301
## 112   7.423232
## 113   9.731909
## 114   9.923722
## 115   7.699714
## 116   9.689474
## 117   8.332015
## 118   8.235026
## 119   9.490487
## 120   9.283870
## 121   9.167556
## 122   9.082761
## 123  10.209300
## 124  13.309969
## 125  10.196204
## 126   9.131814
## 127  14.860871
## 128  14.538406
## 129  11.750455
## 130  15.865058
## 131  11.025317
## 132  11.388863
## 133  10.927960
## 134  14.066853
## 135  16.356883
## 136  13.046616
## 137  13.895850
## 138  11.447417
## 139  16.352287
## 140  13.627782
## 141  16.840221
## 142  59.668586
## 143  17.679838
## 144  18.954229
## 145  26.324018
## 146  18.239333
## 147  12.976190
## 148  26.958868
## 149  23.339506
## 150  14.914879
## 151  10.375446
## 152  11.542708
## 153  11.616815
## 154  12.173099
## 155  10.855253
## 156  11.249235
## 157  14.544679
## 158   5.983270
## 159   6.801784
## 160   8.247291
## 161   6.812143
## 162   5.365453
## 163   5.684315
## 164   5.577436
```

```
## 165   8.334924
## 166   7.723049
## 167   5.454290
## 168   8.377645
## 169   7.624672
## 170   7.672963
## 171   9.029523
## 172   8.463821
## 173   7.914424
## 174   6.107522
## 175   6.495013
## 176   5.288278
## 177   6.706342
## 178   5.988441
## 179   5.575209
## 180   4.894688
## 181   4.979287
## 182   5.868350
## 183   5.007036
## 184   5.493430
## 185   7.696028
## 186   6.925809
## 187   4.565691
## 188   5.740158
## 189   5.818237
## 190   5.656392
## 191   6.419455
## 192   6.583554
## 193   6.010027
## 194   5.890193
## 195   5.989549
## 196   4.967317
## 197   5.820175
## 198   6.512080
## 199   6.163124
## 200   7.415124
## 201   7.259434
## 202   7.220706
## 203   5.557519
## 204   5.008910
## 205   4.854505
## 206   7.748389
## 207   7.911605
## 208  11.108943
## 209   9.333790
## 210  13.894175
## 211  10.251368
## 212  13.677936
## 213   9.512994
## 214   7.052794
## 215  16.832092
## 216   7.361689
## 217   8.149528
## 218   6.912252
```

```
## 219   9.278101
## 220   7.616102
## 221   6.575950
## 222   9.691250
## 223   6.764018
## 224   6.273228
## 225   4.827985
## 226   4.738376
## 227   4.980056
## 228   5.708122
## 229   4.754171
## 230   5.303871
## 231   7.600438
## 232   5.595941
## 233   4.919282
## 234   5.012274
## 235   6.673621
## 236   7.269253
## 237   7.301067
## 238   5.726931
## 239   6.603715
## 240   6.888473
## 241   7.538683
## 242   8.719446
## 243   8.518055
## 244   6.826710
## 245  13.318163
## 246  16.684206
## 247  10.305690
## 248  11.003761
## 249   9.839479
## 250   8.307363
## 251   8.069536
## 252   7.601048
## 253   7.999475
## 254   6.857909
## 255   9.267645
## 256  10.189042
## 257   5.684631
## 258   4.329537
## 259   5.180554
## 260   5.352865
## 261   5.380558
## 262   5.050617
## 263   4.649308
## 264   5.603390
## 265   5.220233
## 266   6.258139
## 267   6.177355
## 268   4.488101
## 269   4.429672
## 270   8.485796
## 271   8.341916
## 272   6.392067
```

```
## 273   6.482513
## 274   6.097819
## 275   5.906462
## 276   5.565552
## 277   6.322659
## 278   6.210644
## 279   6.194772
## 280   4.779296
## 281   4.688133
## 282   5.110612
## 283   5.126844
## 284   5.135968
## 285   6.865861
## 286   7.308242
## 287  11.045675
## 288   7.689121
## 289   7.812485
## 290   7.733285
## 291   6.182611
## 292   6.045868
## 293   6.478338
## 294   6.984656
## 295   7.705271
## 296   6.512145
## 297   6.982427
## 298  10.319550
## 299   7.140008
## 300   6.508511
## 301   7.136281
## 302   6.927096
## 303   6.672039
## 304   5.806403
## 305   5.629823
## 306   6.119028
## 307   5.371305
## 308   5.811331
## 309   6.514872
## 310   7.952130
## 311   9.298012
## 312   6.527059
## 313   8.247498
## 314   7.378693
## 315   7.619630
## 316   9.376786
## 317  11.490406
## 318  10.705019
## 319   7.782840
## 320   8.950467
## 321   7.359624
## 322   7.364077
## 323   7.986783
## 324   9.871253
## 325   7.152633
## 326   7.090399
```

```
## 327   7.587434
## 328   9.831731
## 329   8.259594
## 330   7.047863
## 331   8.285094
## 332   9.840387
## 333   8.114268
## 334   8.218869
## 335   8.500449
## 336   8.767836
## 337   9.010689
## 338   9.739688
## 339   7.961549
## 340   8.460632
## 341   8.524500
## 342   6.386787
## 343   8.430817
## 344   7.737793
## 345   7.274381
## 346  11.736803
## 347  13.250632
## 348   9.105400
## 349   8.006187
## 350   9.303835
## 351  10.146250
## 352  11.073689
## 353  13.287098
## 354   9.436634
## 355  16.387820
## 356  13.977663
## 357  14.725115
## 358  12.455065
## 359  12.613567
## 360  13.041782
## 361  10.898330
## 362  13.286399
## 363  13.494326
## 364  13.681064
## 365   7.288281
## 366  15.990648
## 367  15.436951
## 368  20.995886
## 369   9.632030
## 370   8.092950
## 371   7.672383
## 372   9.702397
## 373  10.201896
## 374  44.062459
## 375 353.388173
## 376  10.274070
## 377  14.819090
## 378  12.928754
## 379  17.074551
## 380  15.744463
```

```
## 381  21.043938
## 382  14.335925
## 383  19.492258
## 384  20.045501
## 385  79.871474
## 386  33.757532
## 387  44.399396
## 388  48.759299
## 389  41.660873
## 390  18.154357
## 391  14.787073
## 392  14.624165
## 393  26.666119
## 394  12.519440
## 395  14.303409
## 396  12.678116
## 397  13.338453
## 398  15.729422
## 399  43.594644
## 400  23.908052
## 401  23.228046
## 402  14.802620
## 403  14.248572
## 404  20.428196
## 405  37.507191
## 406  35.823279
## 407  31.980928
## 408  12.201001
## 409  18.534607
## 410  12.469947
## 411  15.965974
## 412  14.131574
## 413 145.211267
## 414  21.391096
## 415 -64.364659
## 416  26.511600
## 417  18.300269
## 418  36.865340
## 419  44.642881
## 420  16.508682
## 421  12.940361
## 422  13.763437
## 423  13.064494
## 424  17.682921
## 425  15.073905
## 426  24.540455
## 427  13.455100
## 428  17.219690
## 429  16.655413
## 430  18.764439
## 431  12.810184
## 432  12.827557
## 433  10.407136
## 434  13.930590
```

```
## 435  15.134688
## 436  18.829396
## 437  16.599566
## 438  28.345929
## 439  52.768402
## 440  20.083432
## 441  21.182246
## 442  15.210066
## 443  13.754771
## 444  14.569707
## 445  22.383846
## 446  20.609287
## 447  14.393901
## 448  14.332165
## 449  14.925251
## 450  14.884619
## 451  14.258090
## 452  13.301809
## 453  13.798287
## 454  11.715381
## 455  15.718414
## 456  14.943886
## 457  18.439055
## 458  18.021972
## 459  14.530696
## 460  13.730143
## 461  13.093840
## 462  12.561008
## 463  12.838585
## 464  11.230805
## 465  12.185031
## 466  13.214758
## 467  16.023082
## 468  14.921109
## 469  14.911339
## 470  13.168825
## 471  12.459148
## 472  10.778469
## 473  10.972020
## 474   9.476923
## 475  15.272746
## 476  15.835979
## 477  12.451084
## 478  22.647788
## 479  13.337612
## 480  11.437749
## 481  10.275873
## 482   8.915264
## 483   8.505154
## 484  11.091519
## 485  12.153505
## 486  10.833144
## 487  12.799854
## 488  11.122687
```

```
## 489  20.055817
## 490  29.464586
## 491  67.203324
## 492  17.532022
## 493  14.655283
## 494   9.095363
## 495   9.083673
## 496  11.186260
## 497  14.539348
## 498  10.317491
## 499   9.057901
## 500  10.605788
## 501   9.709512
## 502   7.911682
## 503   8.275729
## 504   6.791971
## 505   7.208885
## 506   8.307545
```

Convert $X$ into $Q$ where $Q$ has the same column space as $X$ but has orthogonal columns. You can use the function `orthogonal_projection`. This is essentially gram-schmidt.

```r
Q = matrix(NA , nrow= nrow(X) , ncol(X))
Q[ , 1]=X[ , 1]
for (j in 2 :ncol(X)) {
  Q[ , j]= X[ , j]
  for (j0 in 1 : (j-1)) {
    Q[ , j] = Q[ , j] - (orthogonal_projection(X[ ,j] , Q[ , j0]) $a_parallel)

  }

}

pacman::p_load(Matrix)
rankMatrix(Q)
```

```
## [1] 14
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 1.123546e-13
```

```r
dim(Q)
```

```
## [1] 506  14
```

```r
ncol(X)
```

```
## [1] 14
```

Make $Q$'s columns orthonormal.

```r
for( j in 1 : ncol(Q)){
  Q[ , j] = Q[ , j] / sqrt(sum(Q[ , j]^2))
}
```

Verify $Q^T$ is the inverse of $Q$.

```
veri_inv = t(Q) %*% Q
head(veri_inv)
```

```
##               [,1]          [,2]          [,3]          [,4]          [,5]
## [1,]  1.000000e+00 -1.170938e-16  7.329207e-17 -3.932090e-15  3.044440e-16
## [2,] -1.170938e-16  1.000000e+00  1.566672e-17  6.763727e-17  4.510281e-17
## [3,]  7.329207e-17  1.566672e-17  1.000000e+00 -5.826231e-17  3.794708e-19
## [4,] -3.932090e-15  6.763727e-17 -5.826231e-17  1.000000e+00  1.051744e-15
## [5,]  3.044440e-16  4.510281e-17  3.794708e-19  1.051744e-15  1.000000e+00
## [6,]  7.548107e-15  6.550750e-16  5.526721e-17  3.046028e-14 -2.882202e-15
##               [,6]          [,7]          [,8]          [,9]         [,10]
## [1,]  7.548107e-15 -1.379756e-14 -2.475017e-15 -1.269384e-15  1.098514e-15
## [2,]  6.550750e-16  1.082847e-15 -7.361733e-17  7.773730e-16 -2.138047e-16
## [3,]  5.526721e-17 -2.208520e-16  5.084908e-17  2.385245e-18 -9.540979e-18
## [4,]  3.046028e-14  3.826098e-14 -2.164291e-15  3.891581e-14 -6.627464e-15
## [5,] -2.882202e-15 -2.479679e-15 -1.329232e-16 -2.511229e-15  3.783866e-17
## [6,]  1.000000e+00 -6.696465e-14 -4.081119e-14 -3.385638e-14 -2.339584e-14
##              [,11]         [,12]         [,13]         [,14]
## [1,]  1.463239e-15 -1.382228e-14  1.006416e-14 -6.628812e-15
## [2,]  7.455516e-16  1.229485e-15  2.636644e-16  8.515324e-16
## [3,]  4.065758e-17  2.602085e-17 -5.095750e-17 -1.021318e-16
## [4,]  2.017742e-14  6.014552e-14  2.289555e-14  2.996148e-14
## [5,] -2.035237e-15 -4.422678e-15 -1.515213e-15 -2.182933e-15
## [6,] -6.567132e-14 -8.574749e-14 -3.213684e-14 -6.870822e-14
```

Project $Y$ onto $Q$ and verify it is the same as the OLS fit.

```
head(cbind(Q %*% t(Q) %*% y , y_hat))
```

```
##       [,1]     [,2]
## 1 30.00384 30.00384
## 2 25.02556 25.02556
## 3 30.56760 30.56760
## 4 28.60704 28.60704
## 5 27.94352 27.94352
## 6 25.25628 25.25628
```

Project $Y$ onto the columns of $Q$ one by one and verify it sums to be the projection onto the whole space.

```
proj_col_Q = rep(0 , ncol(Q))
for (j in 1 : ncol(Q)) {
  proj_col_Q = proj_col_Q + orthogonal_projection(y , Q[ , j])$a_pararllel
}
proj_Q = Q %*% t(Q) %*% y
pacman::p_load(testthat)
expect_equal(proj_col_Q ,proj_Q)
```

```
## Error: `proj_col_Q` not equal to `proj_Q`.
## Lengths differ: 0 is not 506
```

Verify the OLS fit squared lengh is the sum of squared lengths of each of the orthogonal projections.

```
sum_sq_length_col_Q = 0
for (j in 1 : ncol(Q)){
  col_proj = orthogonal_projection(y, Q[, j])$a_parallel
  sum_sq_length_col_Q = sum_sq_length_col_Q + sum(col_proj^2)
```

```
}
OLS_sq_length = sum(proj_Q^2)
expect_equal(sum_sq_length_col_Q, OLS_sq_length)
```

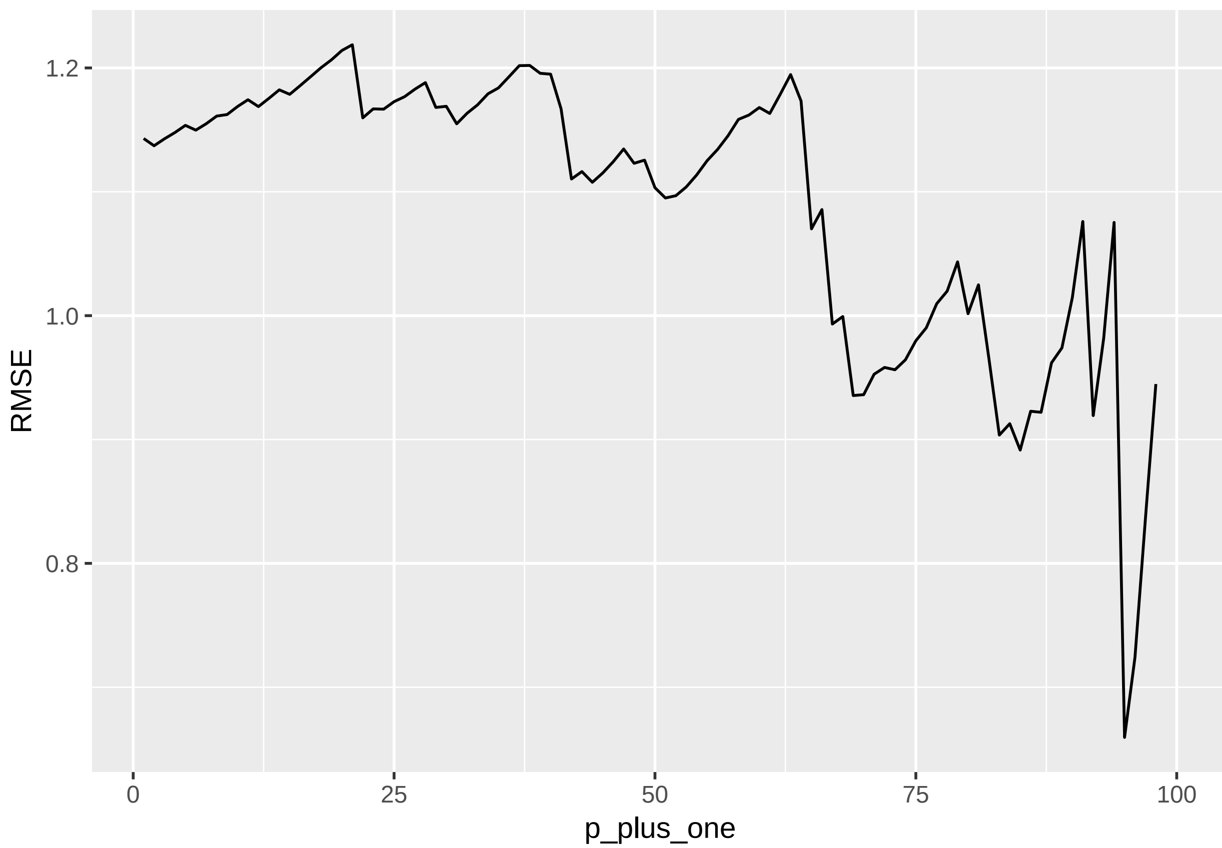
Rewrite the "The monotonicity of SSR" demo from the lec06 notes. Comment every line in detail. Write about what the plots means.

```
n = 100
y = rnorm(n)
Rmses = array(NA, n) #the array that stores RMSE values
X = matrix(NA, nrow = n, ncol = 0) #create a data matrix to store the values
#for every new p that is generated careate another random coloum
for (p_plus_one in 1 : n){          #looping  through all columns of X
  X = cbind(X, rnorm(n))            #adding the random coloum
  Rmses[p_plus_one] = summary(lm(y ~ X))$sigma  #get our rmse
}
pacman::p_load(ggplot2)
```

```
## Installing package into 'C:/Users/burha/Documents/R/win-library/3.6'
## (as 'lib' is unspecified)

## also installing the dependencies 'zeallot', 'colorspace', 'utf8', 'vctrs', 'labeling', 'munsell', 'R

## Warning: unable to access index for repository http://www.stats.ox.ac.uk/pub/RWin/bin/windows/contri
##   cannot open URL 'http://www.stats.ox.ac.uk/pub/RWin/bin/windows/contrib/3.6/PACKAGES'

## package 'zeallot' successfully unpacked and MD5 sums checked
## package 'colorspace' successfully unpacked and MD5 sums checked
## package 'utf8' successfully unpacked and MD5 sums checked
## package 'vctrs' successfully unpacked and MD5 sums checked
## package 'labeling' successfully unpacked and MD5 sums checked
## package 'munsell' successfully unpacked and MD5 sums checked
## package 'RColorBrewer' successfully unpacked and MD5 sums checked
## package 'fansi' successfully unpacked and MD5 sums checked
## package 'pillar' successfully unpacked and MD5 sums checked
## package 'pkgconfig' successfully unpacked and MD5 sums checked
## package 'gtable' successfully unpacked and MD5 sums checked
## package 'lazyeval' successfully unpacked and MD5 sums checked
## package 'plyr' successfully unpacked and MD5 sums checked
## package 'reshape2' successfully unpacked and MD5 sums checked
## package 'scales' successfully unpacked and MD5 sums checked
## package 'tibble' successfully unpacked and MD5 sums checked
## package 'viridisLite' successfully unpacked and MD5 sums checked
## package 'ggplot2' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\burha\AppData\Local\Temp\RtmpkBKAEh\downloaded_packages

##
## ggplot2 installed
```

```
base = ggplot(data.frame(p_plus_one = 1 : n, RMSE = Rmses))
base + geom_line(aes(x = p_plus_one, y = RMSE))
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```

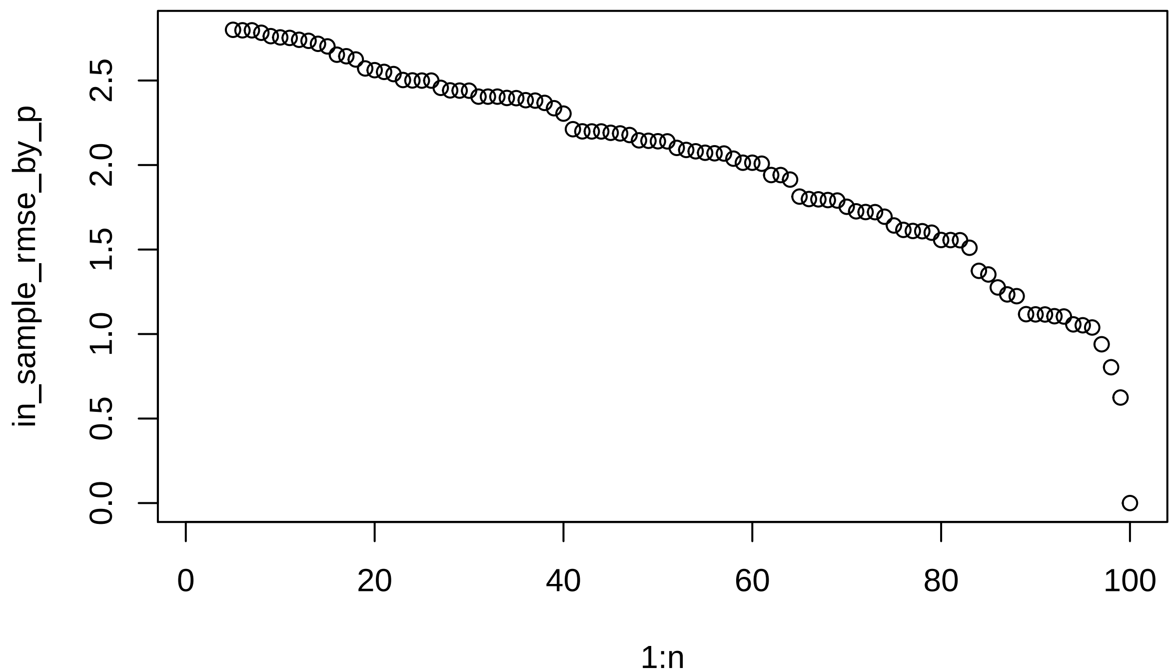



```
#The RMSE value decreases as new coloums are added
#Eventually R^2 becomes 1 that in turn makes RMSE = 0.
```

Rewrite the "Overfitting" demo from the lec06 notes. Comment every line in detail. Write about what the plots means.

```
#make real betas
bbeta = c(1, 2, 3, 4)
#build a random training data set

n = 100 #assign our n sample set
X = cbind(1, rnorm(n), rnorm(n), rnorm(n)) #Intercept + 3 random rnorm columns
y = X %*% bbeta + rnorm(n, 0, 0.3)

#build test data
n_star = 100
X_star = cbind(1, rnorm(n), rnorm(n), rnorm(n_star))
y_star = X_star %*% bbeta + rnorm(n, 0, 0.3)
#caluclarte and store  the betas each time you model on themm
#made a design matrix with p+1 columns
all_betas = matrix(NA, n, n)
all_betas[4, 1 : 4] = coef(lm(y ~ 0 + X)) #fourth row of beta matrix are the beta values from when we h
in_sample_rmse_by_p = array(NA, n)        #Store the In-Sample RMSE
for (j in 5 : n){
  X = cbind(X, rnorm(n))                  #add aanother random column to X
  lm_mod = lm(y ~ 0 + X)                  #create another new linear model w/o intercept
```

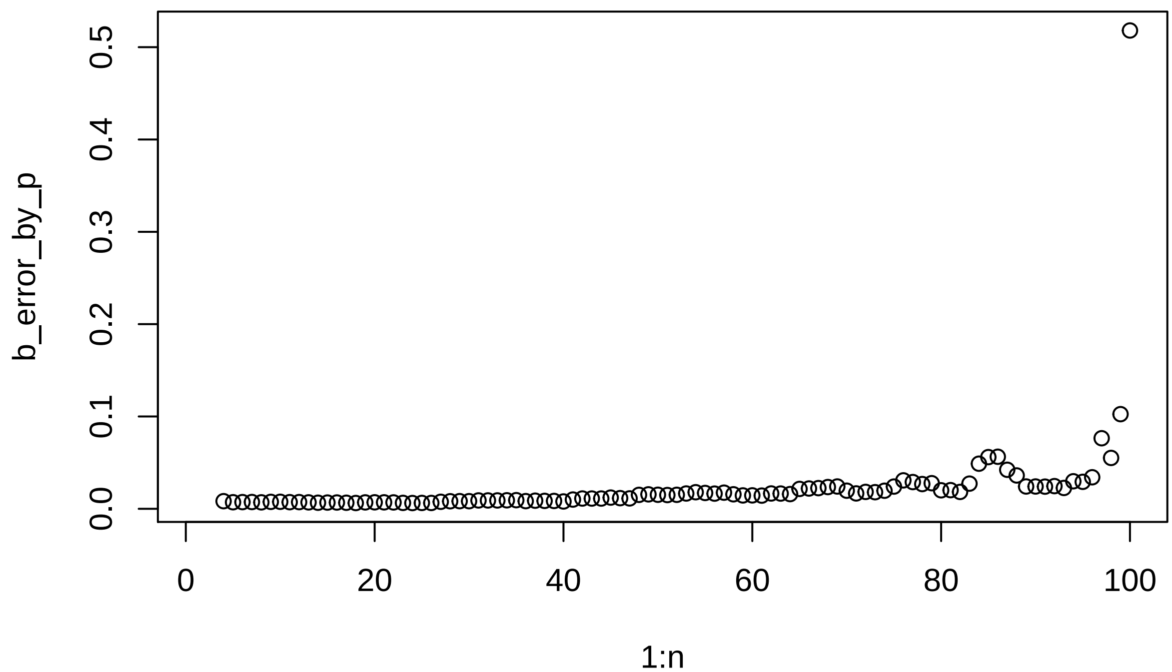
```
  all_betas[j, 1 : j] = coef(lm_mod)      #add betas to the beta_matrix in the column
                                          # add the right vsalues to the coloums

  y_hat = X %*% all_betas[j, 1 : j]       #produce a prediction for our linear model
                                           #new model(using new betas)

  in_sample_rmse_by_p[j] = sqrt(sum((y - y_hat)^2))    #calculate RMSE for the
  #prediction of "each new linear model"
}
plot(1 : n, in_sample_rmse_by_p)
```



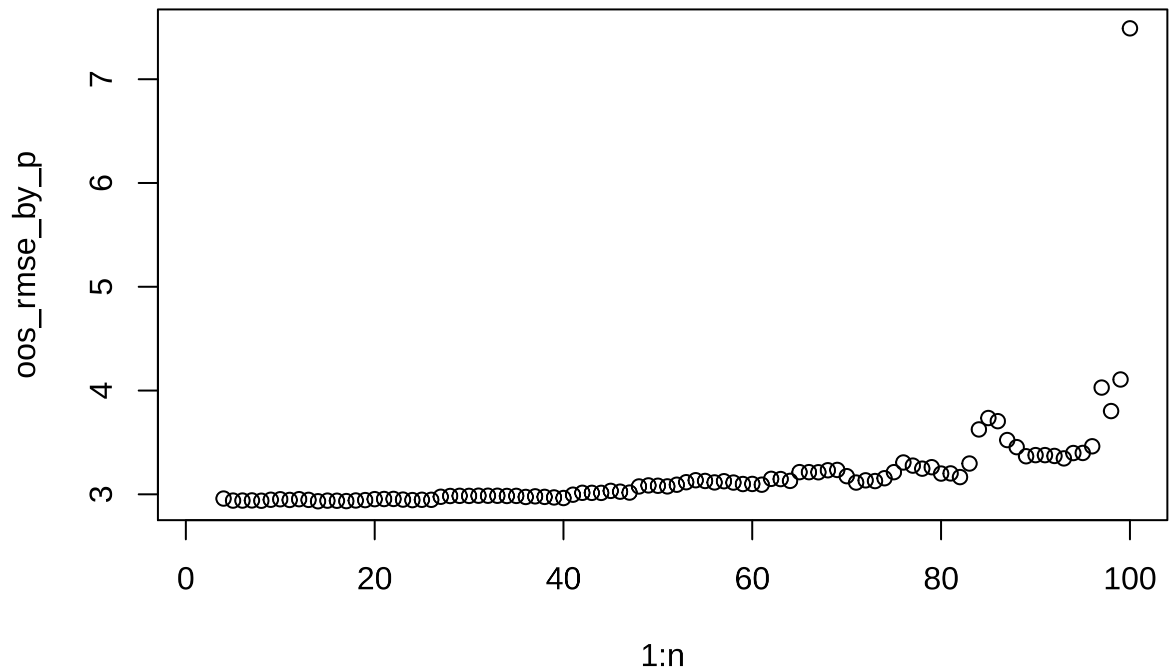

```
#RMSE decreases as the columns get added the nosise when our design matrix approches n
#not real, since we are just testing on given data,
# our model passes through every point in our data
head(all_betas[4 : n, 1 : 4]) #take the first four beta values for each of our models
```

```
##           [,1]     [,2]     [,3]     [,4]
## [1,] 0.9444929 2.002714 2.961398 3.939395
## [2,] 0.9484427 1.997793 2.966938 3.942215
## [3,] 0.9489054 1.999440 2.967366 3.940111
## [4,] 0.9486936 1.999664 2.966749 3.940093
## [5,] 0.9496581 1.996639 2.967535 3.941746
## [6,] 0.9478798 1.998377 2.964468 3.940569
```

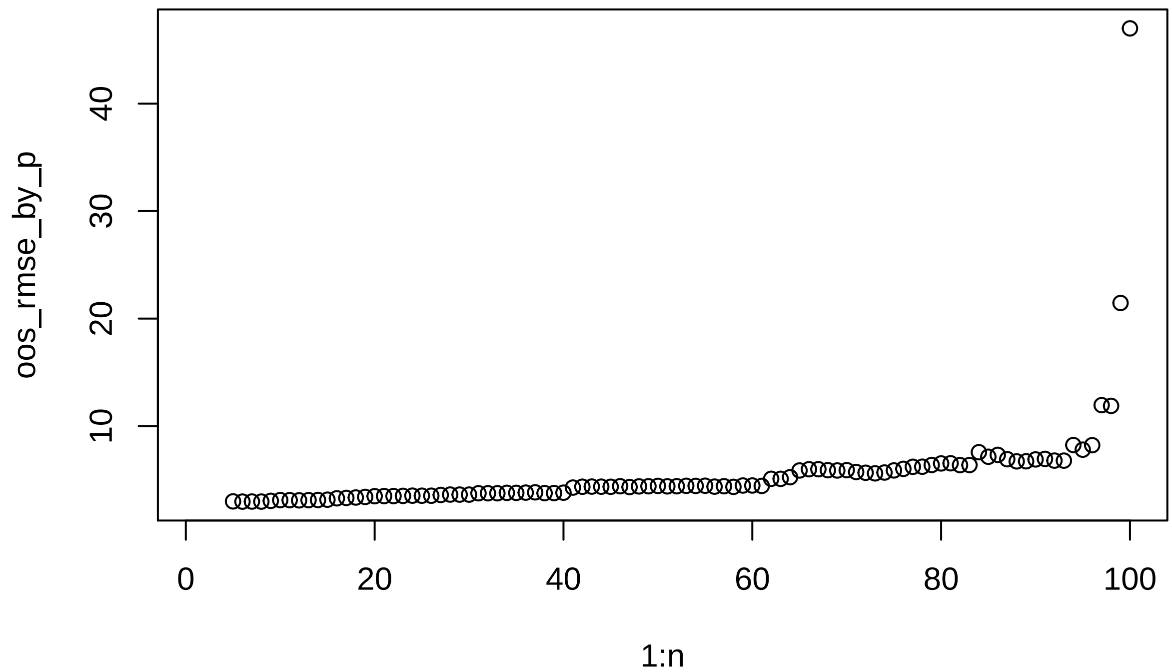
```
b_error_by_p = rowSums((all_betas[, 1 : 4] - matrix(rep(bbeta, n), nrow = n, byrow = TRUE))^2)
#compare our model betas to the real betas
plot(1 : n, b_error_by_p)
```




```
#inspect our out of sample error in the case of only the first four features of the data set
oos_rmse_by_p = array(NA, n)     #store our out of sample RMSE = oosRMSE
for (j in 4 : n){
  y_hat_star = X_star %*% all_betas[j, 1 : 4] #predict on the data using the betas from the "first four
  oos_rmse_by_p[j] = sqrt(sum((y_star - y_hat_star)^2)) #calculate RMSE
}
plot(1 : n, oos_rmse_by_p)
```




```
#look at out of sample error
#in our case of the random features
oos_rmse_by_p = array(NA, n)
for (j in 5 : n){
  X_star = cbind(X_star, rnorm(n))              #add a another random column
  y_hat_star = X_star %*% all_betas[j, 1 : j] #prediction
  oos_rmse_by_p[j] = sqrt(sum((y_star - y_hat_star)^2)) #oosRMSE for when you add random columns
}
plot(1 : n, oos_rmse_by_p)
```




```
#oosRMSE actually tests how well our model performs
#the predicting power of our model
#significantly decreases as we increase the columns of X
```