a) Ans:

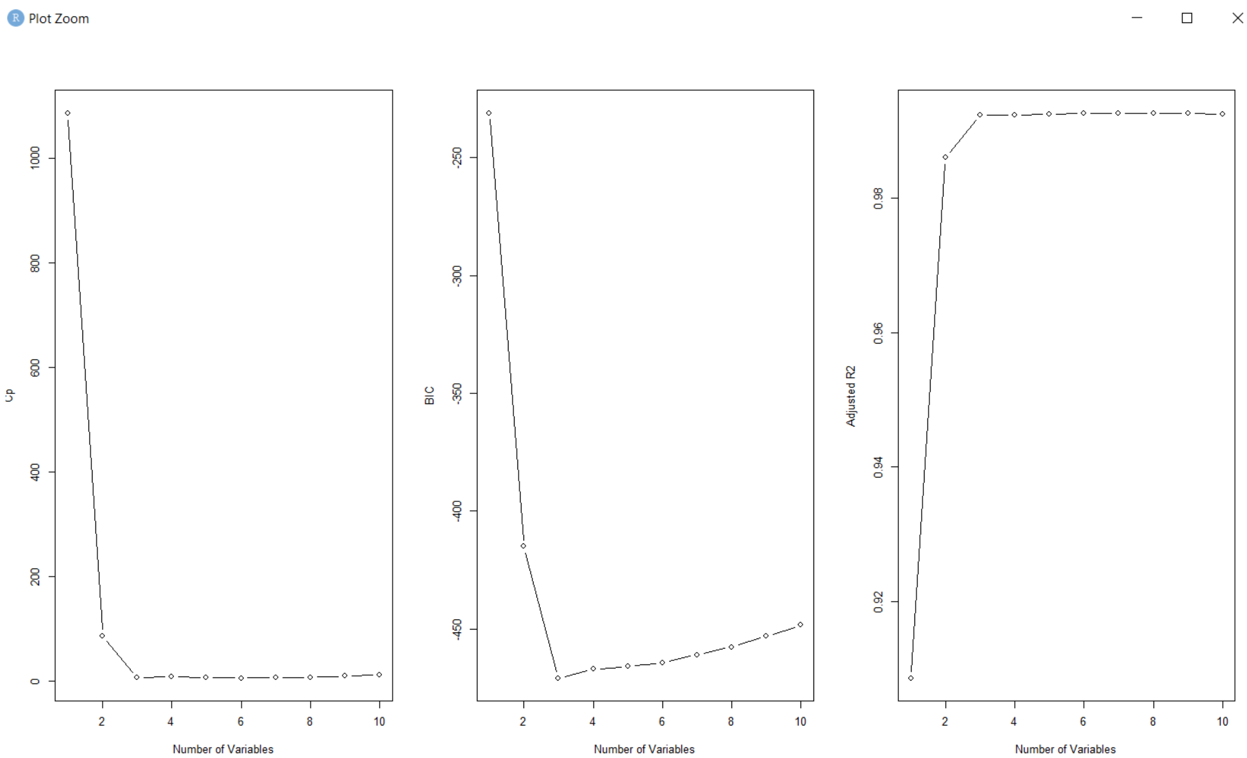
```
> n = 100
> X = rnorm(n)
> Epsilon = rnorm(n)
>
```

b) Ans:

```
> β0 = 1; β1 = 2; β2 = 3; β3 = 4
> Y <- β0 + β1*X + β2*X^2 + β3*X^3 + Epsilon
> Y
  [1]   9.11268604  -0.50328770  46.64200818   0.05754668  30.20492616  24.34156438
  [7]   0.82223478   3.53397272  -0.76005025   1.39627553  10.51294417   1.88403350
 [13]   1.46057424  -0.92160051   0.42375478  11.97992270   1.29905042   0.54365751
 [19]  -6.66357258   4.91391408  19.13191054  23.42737098   6.71856443 -20.15866600
 [25]   3.67884224   4.52841080  -1.14595307   0.66843265  -1.70978672   4.31485803
 [31]  -1.50649519  -9.70358411  -1.06009237  19.44949972   1.11023830   5.45179284
 [37]   9.86590802   0.98314072  -0.15484853   0.83421818   5.30452060  -0.77231186
 [43]   1.71572230   1.42819428 -34.40847174 -10.05665214   7.05769676   0.27510897
 [49]   6.85248060  40.01686832  24.23343629   4.36882129   1.65958637  -0.85836097
 [55]  25.88529455   4.16311864  -3.17132548  -1.35157575 -21.05889024   1.22985431
 [61] -54.16065954  20.04821140   1.15958422   0.97220471   0.73283733   4.54609506
 [67]   4.27975086   4.46444523   1.56781446  -5.20251447  -0.37232527   0.25452041
 [73]  -0.95385683   1.31611976  -2.26215292  -0.33502269   1.68046445   0.99317092
 [79]   1.92911870  35.76894388   5.68638145  11.27359527  -1.95433921   1.06533896
 [85]  15.82146580   3.34235027 -28.44248461   1.16681418 -21.54751035  -0.26866779
 [91]  17.10024919   6.58628618  11.07689075   2.12162723   4.62101693  -2.55323488
 [97]  28.27279003  -0.01043066  -0.01126861   6.69387686
>
```

c) Ans:

```
> library(leaps)
> data = data.frame(Y = Y,X1 = X,X2 = X^2,X3 = X^3,X4 = X^4,X5 = X^5,X6 = X^6,X7 = X^7,X8 = X^8,X9
= X^9,X10 = X^10)
> best_subset = regsubsets(Y ~ ., data = data, nvmax = 10)
> summary.reg = summary(best_subset)
> summary.reg$cp
 [1] 1086.104639    85.498864     6.186395     7.383294     6.092477     5.288440     6.128827
 [8]     7.002717     9.000439    11.000000
> summary.reg$bic
 [1] -231.1328 -415.0159 -470.7408 -466.9570 -465.7894 -464.2097 -460.8829 -457.5350 -452.9324
[10] -448.3278
> summary.reg$adjr2
 [1] 0.9086702 0.9859889 0.9922560 0.9922385 0.9924210 0.9925678 0.9925825 0.9925947 0.9925126
[10] 0.9924285
> par(mfrow = c(1,3))
> plot(summary.reg$cp, xlab = "Number of Variables", ylab = "Cp", type = "b")
> plot(summary.reg$bic, xlab = "Number of Variables", ylab = "BIC", type = "b")
> plot(summary.reg$adjr2, xlab = "Number of Variables", ylab = "Adjusted R2", type = "b")
> coef(best_subset, 3)
(Intercept)          X1          X2          X3 
   1.113556    2.059186    2.836236    3.974698 
> coef(best_subset, 4)
(Intercept)          X1          X2          X3          X5 
 1.13667646  1.85868682  2.79845558  4.16948947 -0.03207058 
> 
```



From the output we know that the model with 3 variables (X, X^2, X^3) is the best choice.

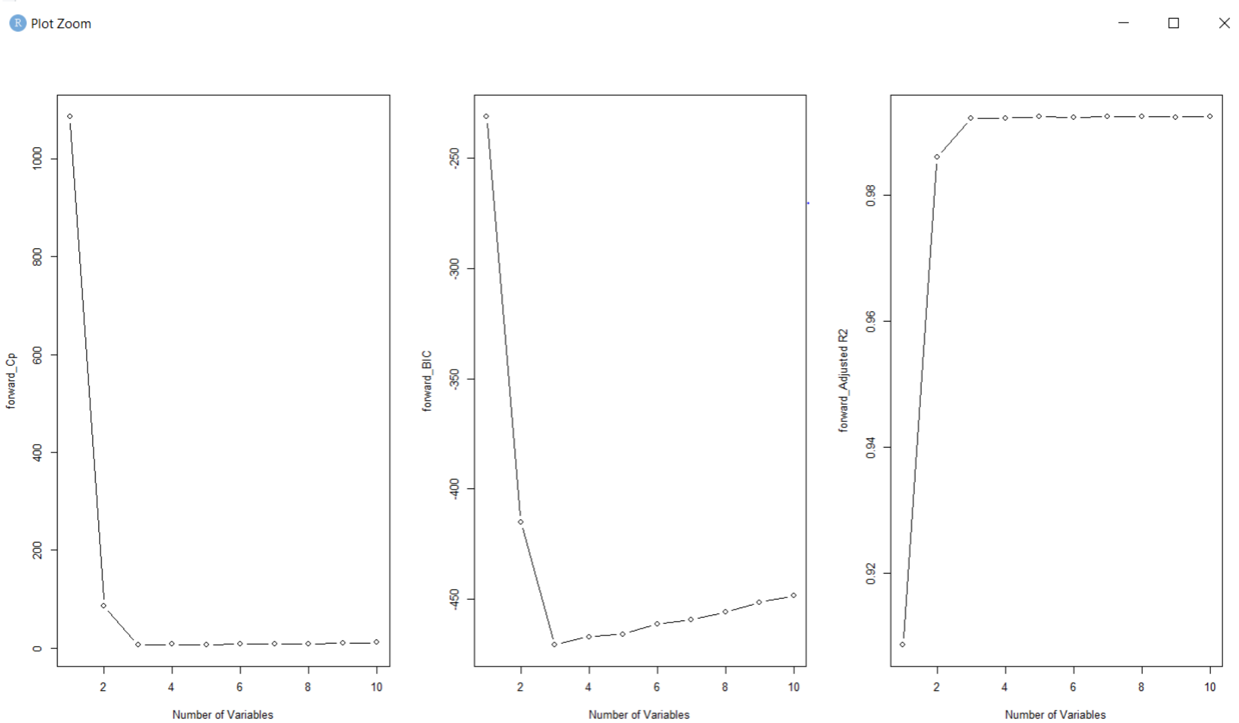
- We choose the model with the lowest BIC (because lower BIC = better model with good fit and less complexity). Best BIC is at 3 variables (-470.74) because it's the smallest (most negative) value.
- We want Adjusted R^2 to be as high as possible, but we also look for where it stops improving a lot. Adjusted R^2 improves a lot up to 3 variables, but then flattens after that (0.992 → 0.992 → 0.992). This means adding more variables doesn't make the model better after 3 variables.
- Here we choose a model with Cp close to the number of variables. Best Cp is at 3 or 4 variables, but again, 3 is enough because it's closer and matches BIC.
- Coefficients for the 3-variable model match the true cubic relationship we simulated (X, X^2, X^3).

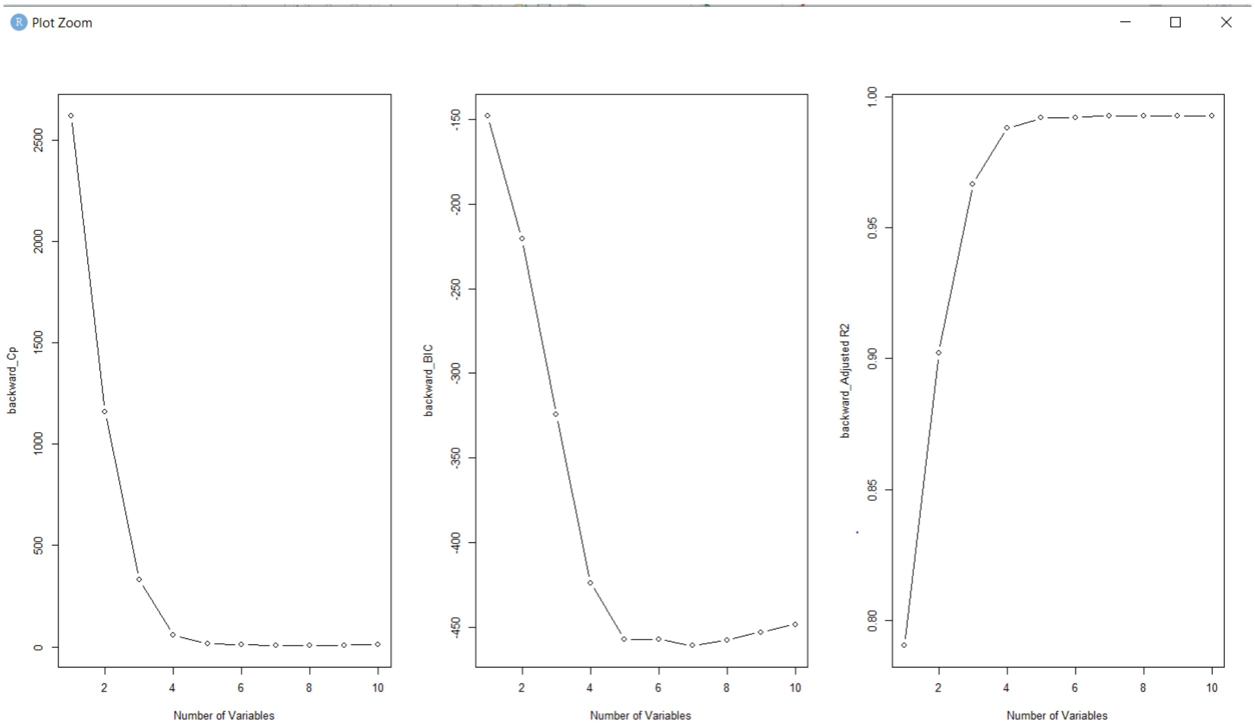
Best Model:

Hence, the model with 3 variables (X, X^2, X^3) is the best choice.

d) Ans:

```
> forward_model = regsubsets(Y ~ ., data = data, nvmax = 10, method = "forward")
> summary.forward = summary(forward_model)
> par(mfrow = c(1, 3))
> plot(summary.forward$cp, xlab = "Number of Variables", ylab = "forward_Cp", type = "b")
> plot(summary.forward$bic, xlab = "Number of Variables", ylab = "forward_BIC", type = "b")
> plot(summary.forward$adjr2, xlab = "Number of Variables", ylab = "forward_Adjusted R2", type =
"b")
> backward_model = regsubsets(Y ~ ., data = data, nvmax = 10, method = "backward")
> summary.backward = summary(backward_model)
> par(mfrow = c(1, 3))
> plot(summary.backward$cp, xlab = "Number of Variables", ylab = "backward_Cp", type = "b")
> plot(summary.backward$bic, xlab = "Number of Variables", ylab = "backward_BIC", type = "b")
> plot(summary.backward$adjr2, xlab = "Number of Variables", ylab = "backward_Adjusted R2", type =
"b")
> summary.forward$cp
 [1] 1086.104639   85.498864    6.186395    7.383294    6.092477    7.961659    7.600343
 [8]    8.419645   10.349037   11.000000
> summary.backward$cp
 [1] 2618.160625 1159.991107  331.530656   59.990047   14.423327   11.895859    6.128827
 [8]    7.002717    9.000439   11.000000
> coef(forward_model, 3)
(Intercept)          X1          X2          X3
   1.113556    2.059186    2.836236    3.974698
> coef(backward_model,4)
(Intercept)          X1          X2          X5          X7
  1.0821532   4.6766938   2.9848434   1.3672846  -0.1249905
>
```





Forward Stepwise Selection

The results are exactly the same as Best Subset Selection in 8(c). Cp, BIC, and Adjusted R^2 all favor the 3-variable model. Hence, Selected model is 3 variables (X, X^2, X^3) with the same coefficients as above in (c).

Backward Stepwise Selection

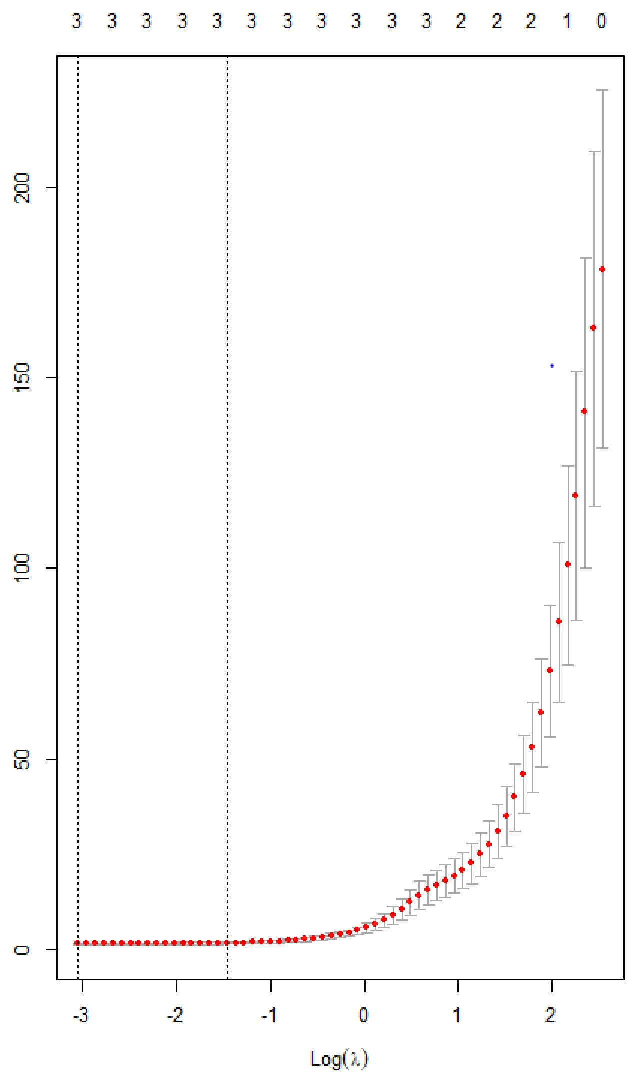
Cp is slightly lower for 6–7 variables, but differences are small. BIC still prefers smaller models (around 3 variables). And adjusted R^2 increases slightly for larger models but plateaus after 3 variables. Hence, backward may include extra variables, like X5 or X7, but does not significantly improve fit.

e) Ans:

Using the optimal λ (left dotted line), Lasso selected 3 predictors out of the 10 possible. The minimum error (left line) is at a $\log(\lambda)$ around -3. At this λ, the model keeps 3 non-zero predictors. Hence Lasso identifies the cubic relationship and removes noise variables.

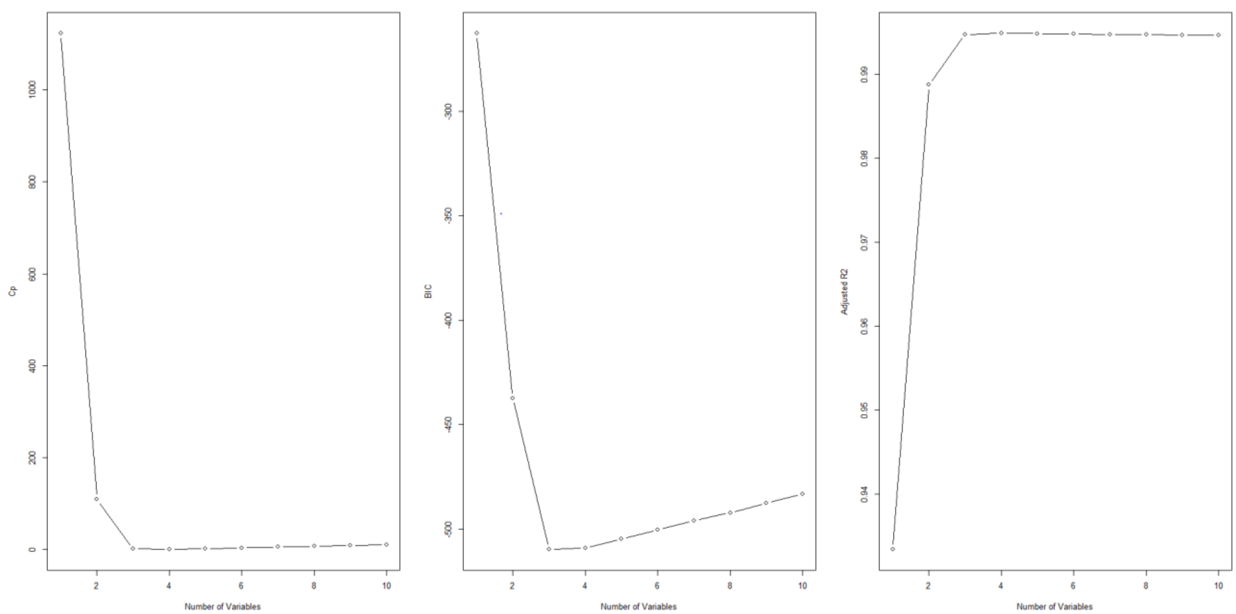
```
> x.mat = model.matrix(Y ~ poly(X, 10, raw = TRUE))[,-1]
> y.vec = Y
> set.seed(1)
> cv.lasso = cv.glmnet(x.mat, y.vec, alpha = 1)
> plot(cv.lasso)
> best.lambda = cv.lasso$lambda.min
> best.lambda
[1] 0.04757254
> lasso.fit = glmnet(x.mat, y.vec, alpha = 1, lambda = best.lambda)
> coef(lasso.fit)
11 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)                 1.172886
poly(X, 10, raw = TRUE)1  2.089356
poly(X, 10, raw = TRUE)2  2.768111
poly(X, 10, raw = TRUE)3  3.940270
poly(X, 10, raw = TRUE)4  .
poly(X, 10, raw = TRUE)5  .
poly(X, 10, raw = TRUE)6  .
poly(X, 10, raw = TRUE)7  .
poly(X, 10, raw = TRUE)8  .
poly(X, 10, raw = TRUE)9  .
poly(X, 10, raw = TRUE)10 .
>
```

Plot Zoom



f) Ans:

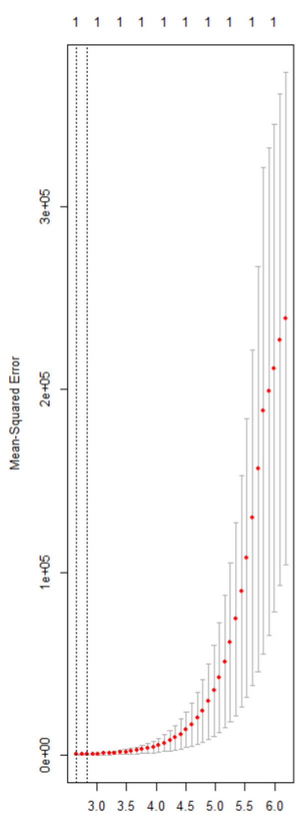
```
> β7=8
> Y_new = β0 + β7*X^7 + Epsilon
> Y_new
  [1]  7.672349e-02  1.042172e+00 -2.186978e+00  2.115116e+02  3.487894e-01  7.650159e-01  1.769004e+00  2.866976e+00
  [9]  1.552024e+00  2.680194e+00  1.447461e+02  5.493029e-01  2.146583e+00 -2.090363e+03  1.903036e+01  6.071921e-01
 [17]  6.800071e-01  6.058738e+00  3.509355e+00  1.031162e+00  4.922207e+00  3.775448e+00  7.854207e-01 -9.856202e+02
 [25]  1.180985e+00  1.712666e+00  9.264178e-01 -1.181253e+02  2.726265e-01  6.935494e-01  6.943704e+01  4.111045e-01
 [33]  1.542024e+00 -5.183941e-01 -7.381404e+01 -5.534083e-01  6.871716e-01  4.717201e-01  1.594016e+01  2.149422e+00
 [41] -9.143855e-01  2.176047e+00 -2.588587e-02  6.689762e-01 -7.041491e-01 -4.606489e-01  3.094016e+00  2.284252e+00
 [49] -2.863023e-01  2.657716e+00  1.462866e+00  7.241157e-01  6.862313e-01 -1.867608e+01  9.879296e+01  9.557099e+02
 [57]  1.992825e+00 -1.044521e+01 -2.285727e-01  2.869284e+00  3.687950e+03  7.613529e-01  2.652618e+00  1.886423e+00
 [65] -6.218468e-01  3.206171e+00 -4.985541e+02  1.157483e+02  8.556163e-01  1.829159e+03  3.351953e+00  3.785770e-01
 [73]  1.710521e+00 -4.041200e+00 -3.826395e+01  9.667029e-01  1.760729e+00  3.075245e+00  2.027393e+00  2.009945e+00
 [81] -3.851762e-01  1.983889e+00  2.641592e+01 -1.529148e+02  1.729625e+00  8.448741e-01  1.474204e+01  2.319903e-01
 [89]  5.773855e-01  7.466638e-02  7.122336e-01  3.140924e+01  2.293300e+01  2.490616e+00  2.024712e+02  8.759019e-02
 [97] -4.176221e+01 -1.786196e-01 -3.163115e+01  5.762963e-01
> best_subset_new = regsubsets(Y ~ ., data , nvmax = 10)
> summary_new = summary(best_subset_new)
> par(mfrow = c(1, 3))
> plot(summary_new$cp, xlab = "Number of Variables", ylab = "Cp", type = "b")
> plot(summary_new$bic, xlab = "Number of Variables", ylab = "BIC", type = "b")
> plot(summary_new$adjr2, xlab = "Number of Variables", ylab = "Adjusted R2", type = "b")
> coef(best_subset_new, 7)
 (Intercept)           X1           X2           X3           X5           X6           X8          X10
 1.074706727  2.376680750  2.928139705  3.574377652  0.078062625 -0.113745920  0.045641386 -0.004563741
> coef(best_subset_new, which.min(summary_new$cp))
(Intercept)          X1          X2          X3          X5
 1.07200775  2.38745596  2.84575641  3.55797426  0.08072292
> coef(best_subset_new, which.min(summary_new$adjr2))
(Intercept)          X3
   3.437156    4.828270
> 
```



BIC (lowest) selected 3 variables (X1, X2, X3). This is wrong, because the true model only contains X7. Cp (lowest) selected 5 variables (X1, X2, X3, X5). Again, this is incorrect. Adjusted R^2 (highest) selected 1 variable (X3) but still wrong.

Hence, best subset selection completely failed to identify X7. It was misled by the strong correlations between X, X2, …, X10 and the noise.

```
> library(glmnet)
> x.mat.new = model.matrix(Y_new ~ poly(X, 10, raw = TRUE))[,-1]
> y.vec.new = Y_new
> set.seed(1)
> cv.lasso.new = cv.glmnet(x.mat.new, y.vec.new, alpha = 1)
> plot(cv.lasso.new)
> best.lambda.new = cv.lasso.new$lambda.min
> best.lambda.new
[1] 14.13562
> lasso.fit.new = glmnet(x.mat.new, y.vec.new, alpha = 1, lambda = best.lambda.new)
> coef(lasso.fit.new)
11 x 1 sparse Matrix of class "dgCMatrix"
                                 s0
(Intercept)              1.947394330
poly(X, 10, raw = TRUE)1  .
poly(X, 10, raw = TRUE)2  .
poly(X, 10, raw = TRUE)3  .
poly(X, 10, raw = TRUE)4  .
poly(X, 10, raw = TRUE)5  .
poly(X, 10, raw = TRUE)6  .
poly(X, 10, raw = TRUE)7  7.750481484
poly(X, 10, raw = TRUE)8  .
poly(X, 10, raw = TRUE)9  0.002855533
poly(X, 10, raw = TRUE)10 .
> 
```



Lasso correctly selected X7 and eliminated all irrelevant predictors. The estimated coefficient of 7.75 is very close to the true value β7=8. Hence, Lasso outperformed best subset selection when the true model has only one predictor (X7).