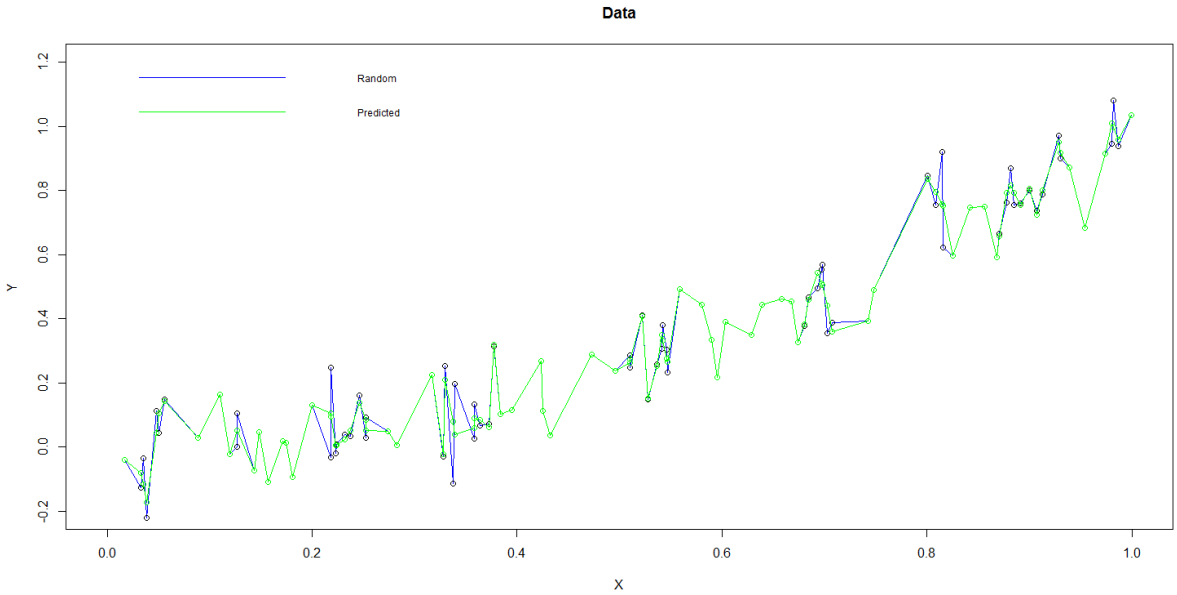
Statistics 202A

Anurag Pande – 604749647

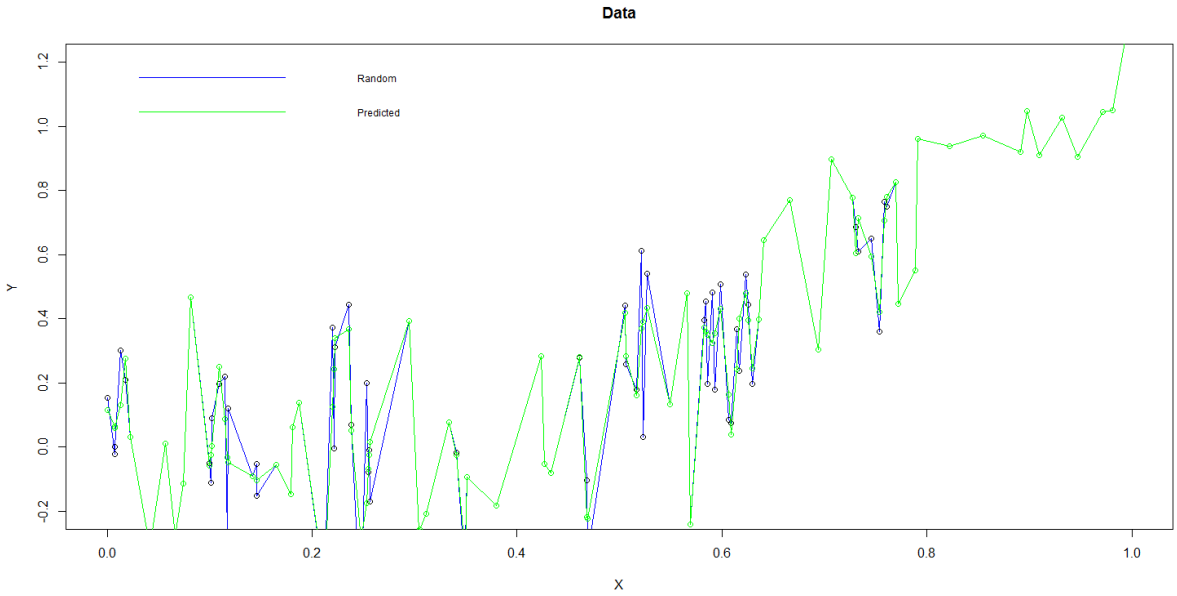
Input data is taken as a randomized set of 100 values in the range of 0 to 1 as the X co-ordinate, and Y co-ordinate as $(X^2 + randomvalue_{normalized} * sigma)$

Regularization can be observed as how close the predicted co-ordinates are to the predicted values. The value of lambda changes the regularization, fitting the predicted values differently to the actual dataset.

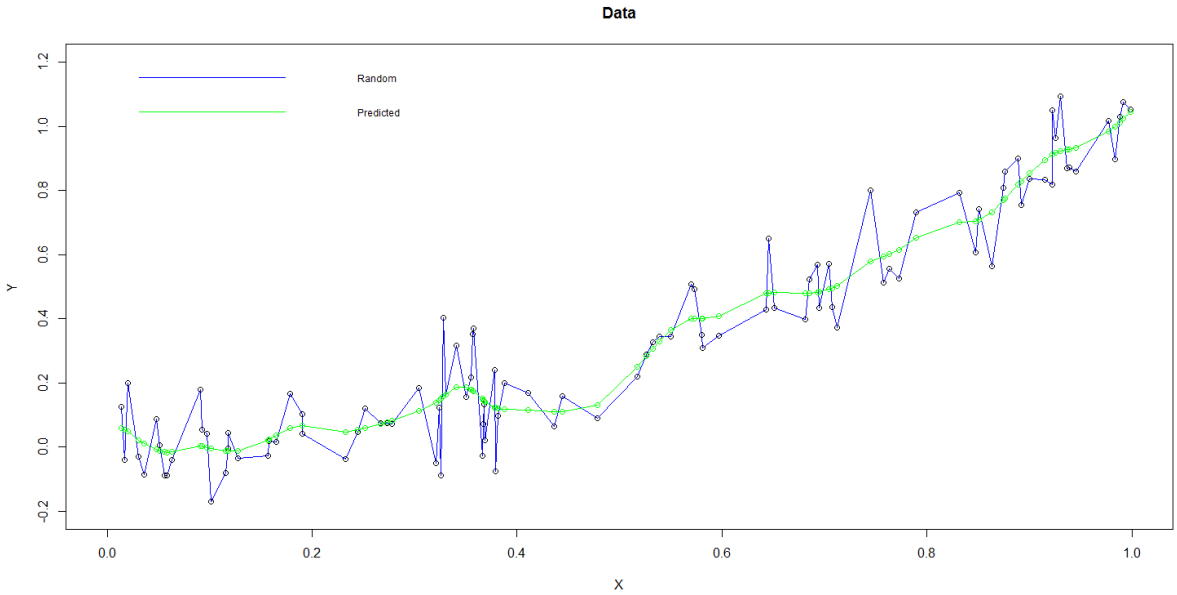
We can see that values of lambda > 1 do not produce quite accurate results, while values of lambda < 0.01 overfit the model too much.



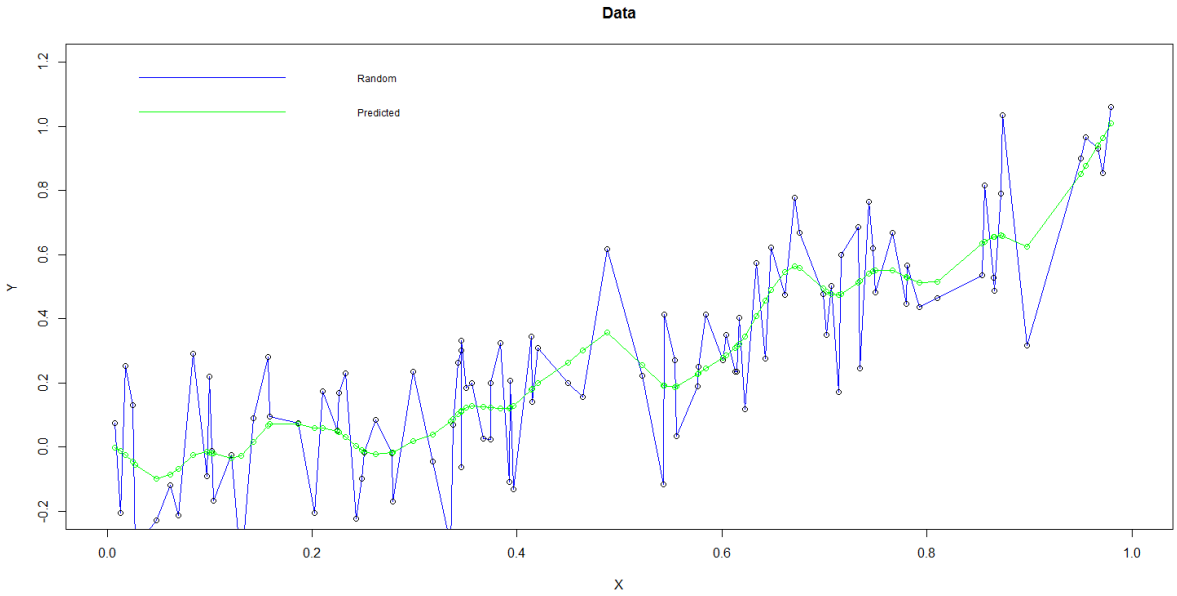
Sigma = 0.1, Lambda = 0.0



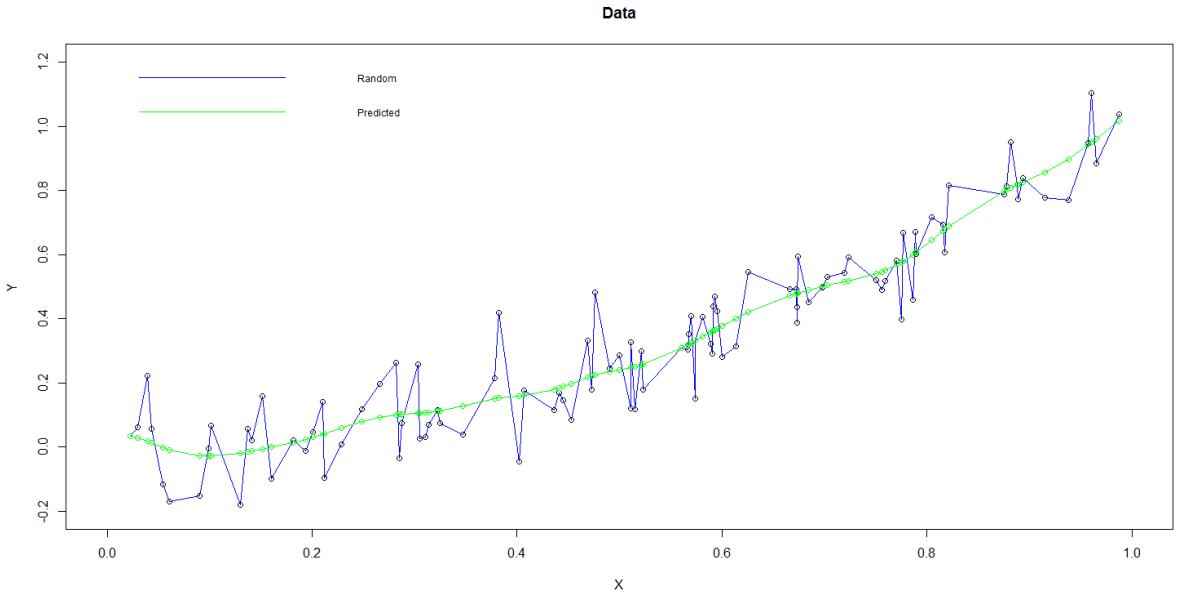
Sigma = 0.2, Lambda = 0.0



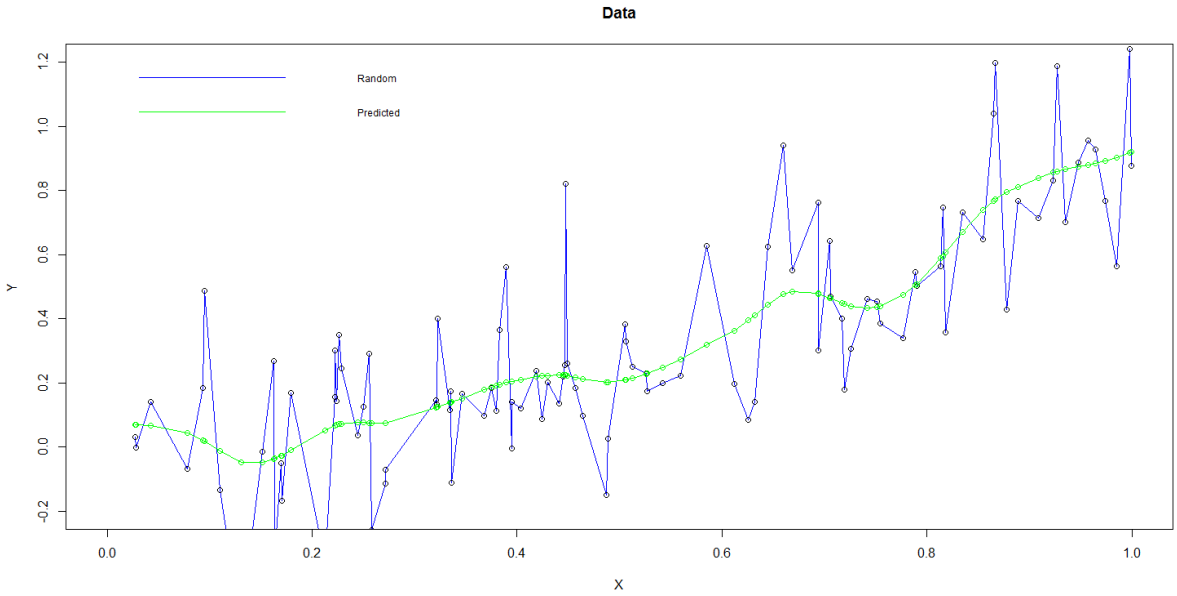
Sigma = 0.1, Lambda = 0.001



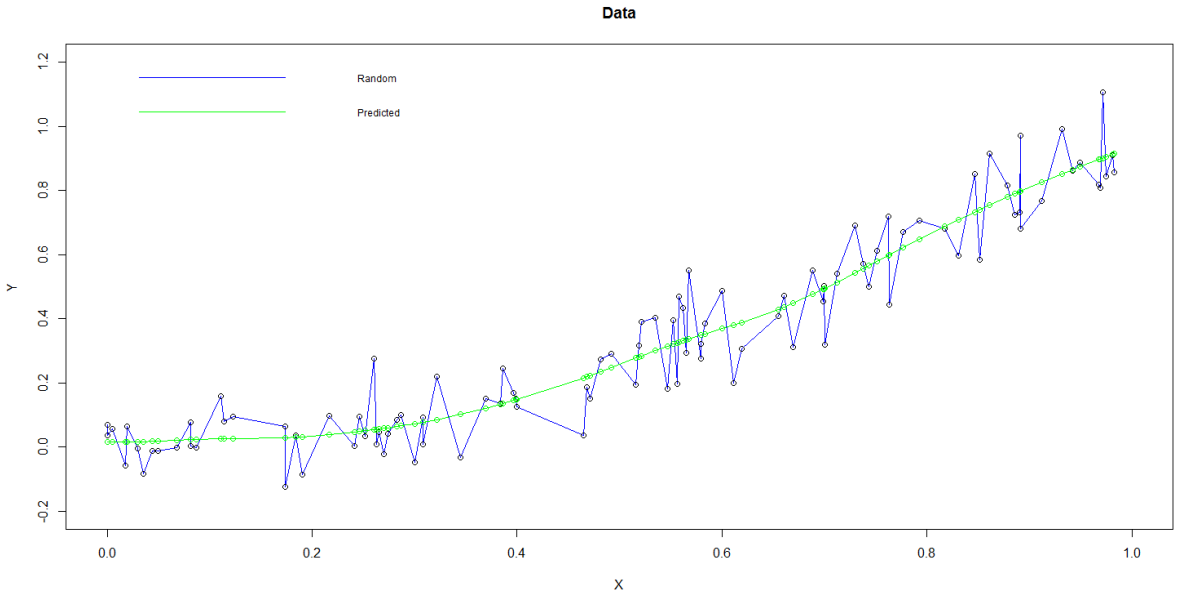
Sigma = 0.2, Lambda = 0.001



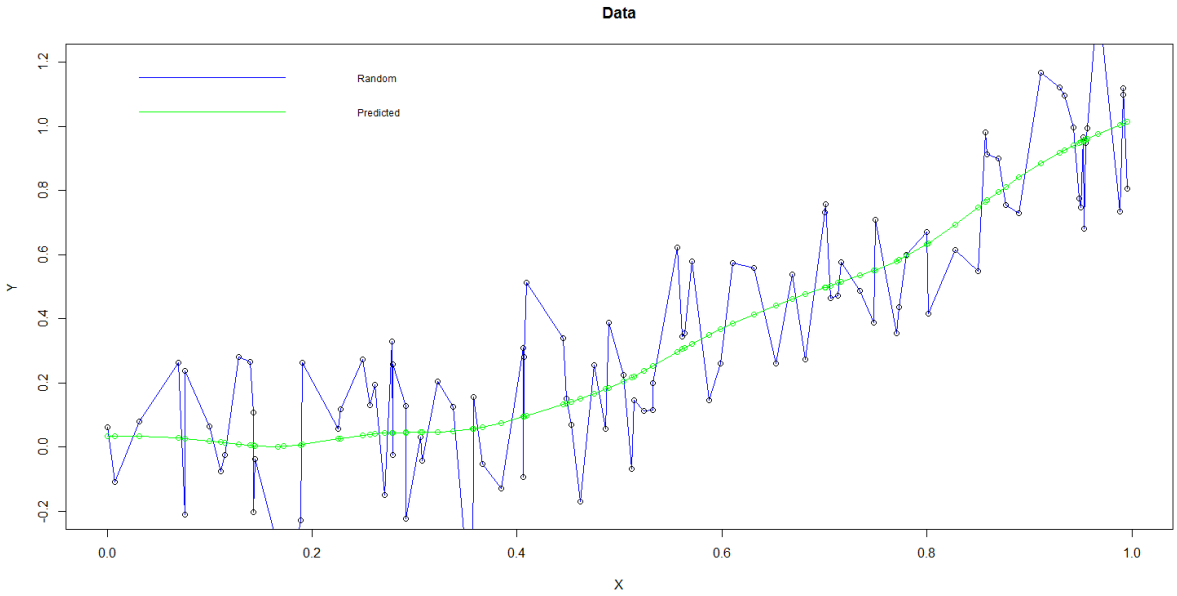
Sigma = 0.1, Lambda = 0.01



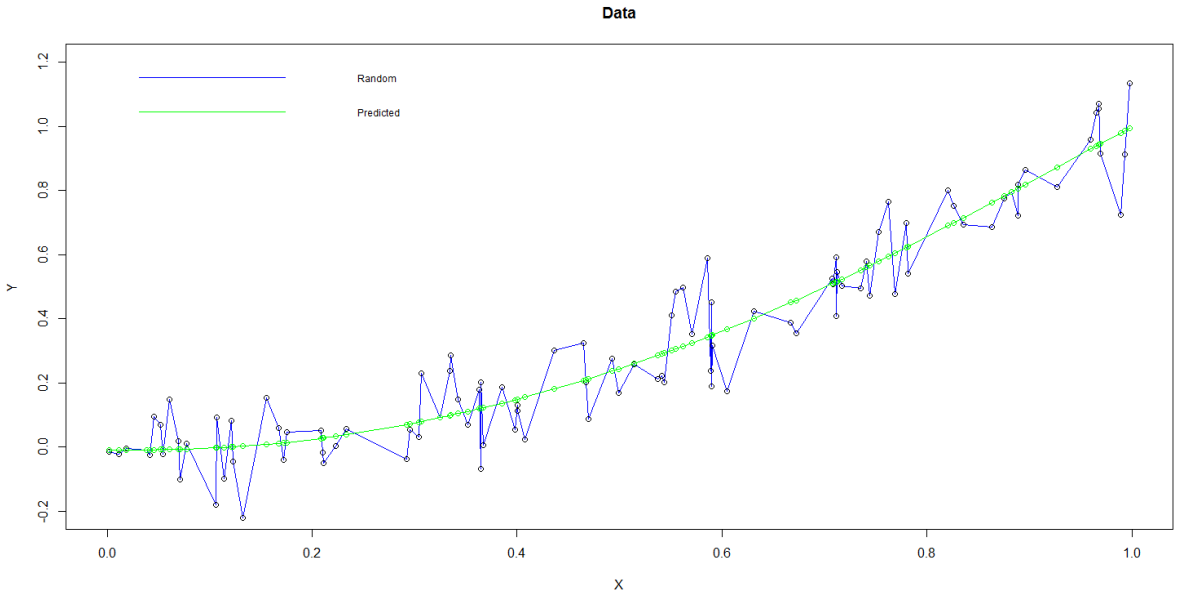
Sigma = 0.2, Lambda = 0.01



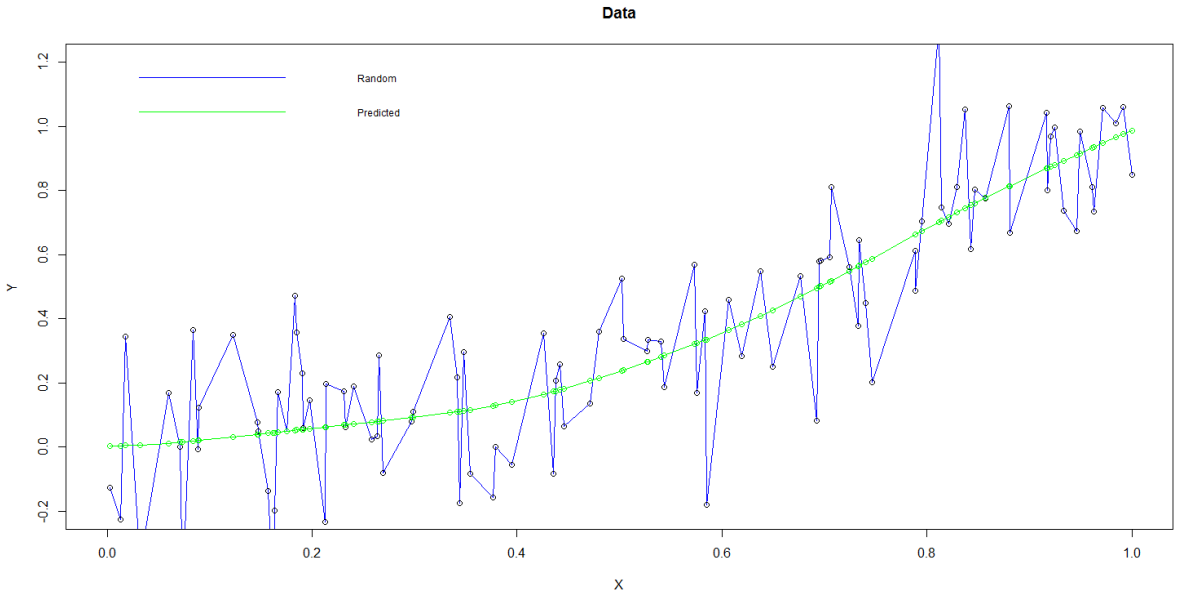
Sigma = 0.1, Lambda = 0.1



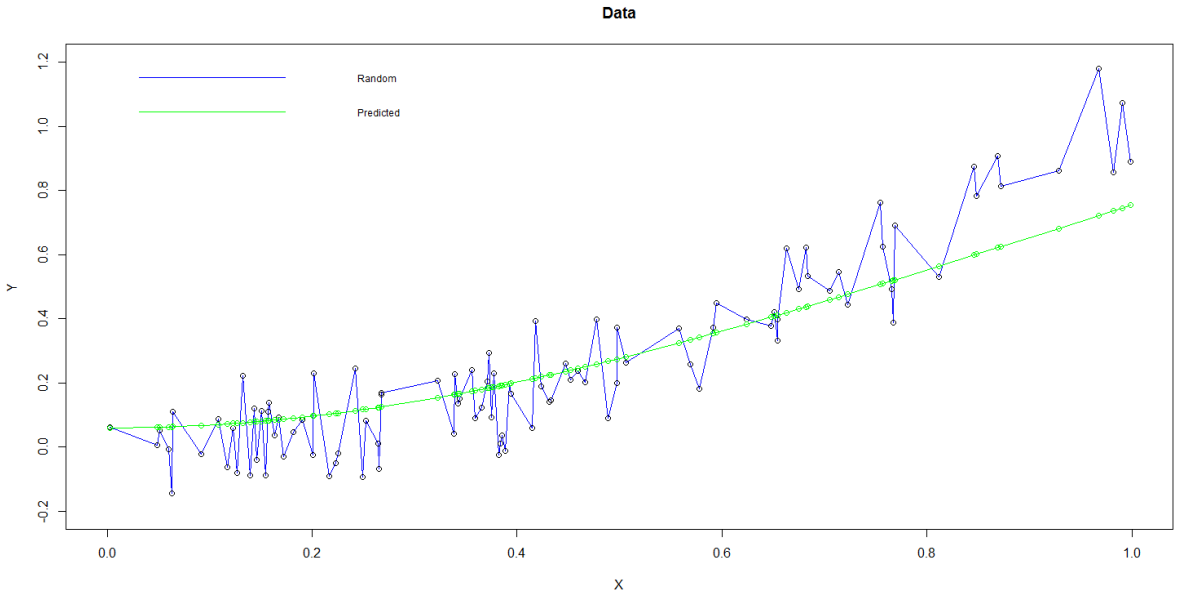
Sigma = 0.2, Lambda = 0.1



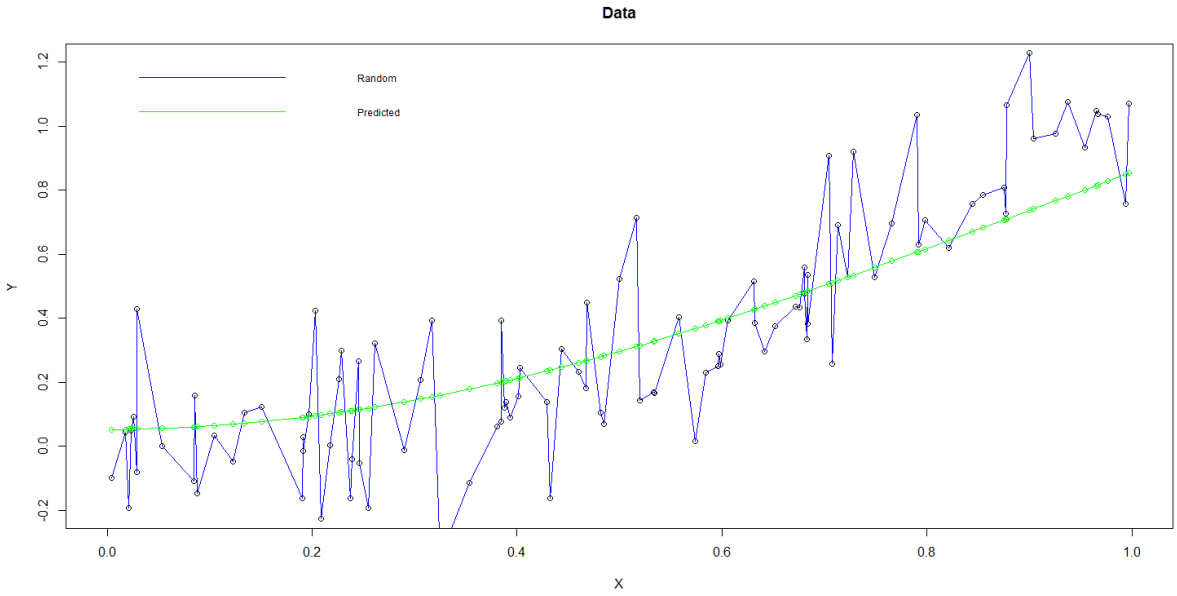
Sigma = 0.1, Lambda = 1



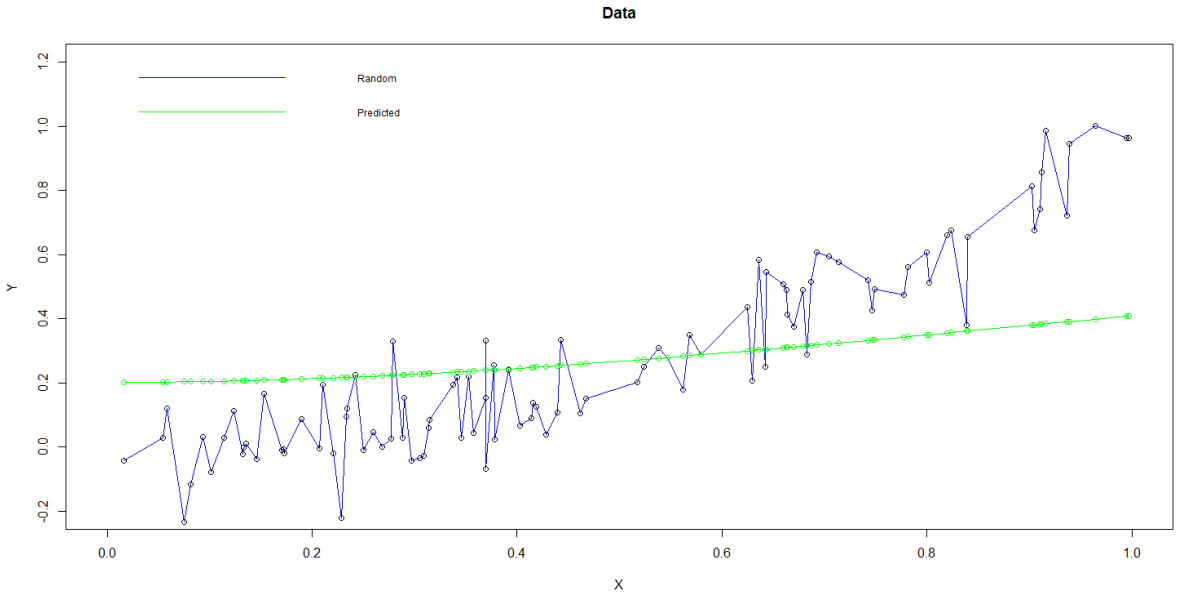
Sigma = 0.2, Lambda = 1



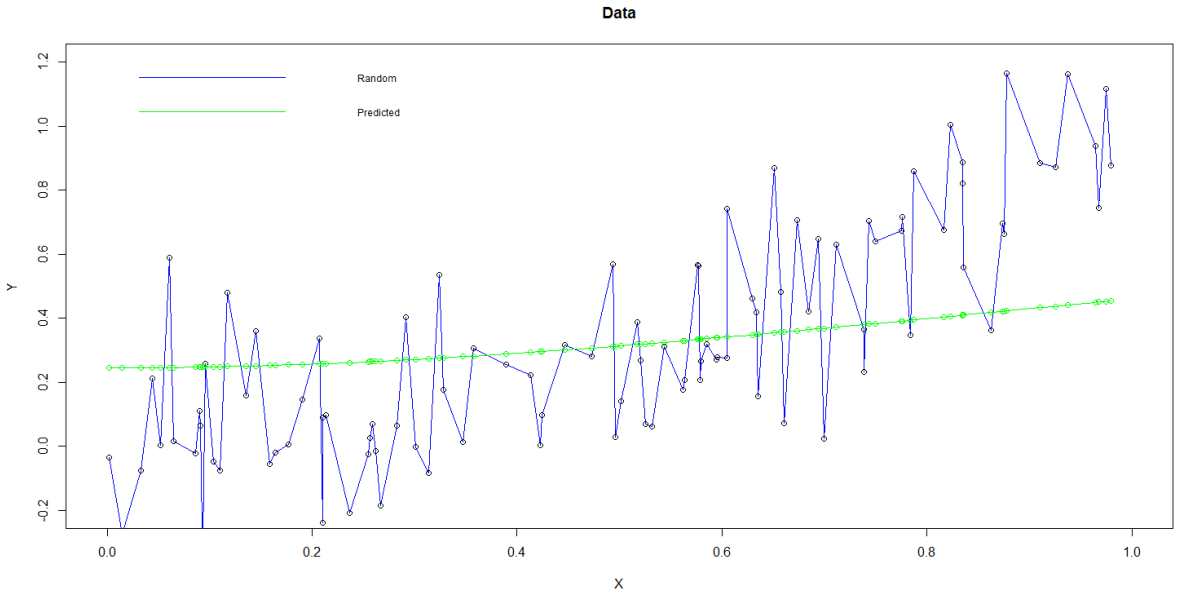
Sigma = 0.1, Lambda = 100



Sigma = 0.2, Lambda = 100



Sigma = 0.1, Lambda = 1000



Sigma = 0.2, Lambda = 1000

Divide data into 80% training data and 20% testing data as:

```
X <- runif(n)

train_x <- X[1:80]
test_x <- X[81:100]

train_x <- sort (train_x)
test_x <- sort (test_x)

train_y <- train_x^2 + rnorm(80) * sigma
test_y <- test_x^2 + rnorm(20) * sigma
```

Then, the beta_spline can be got from the training data as:

```
op_train <- mySpline(train_x,train_y,lambda,p)
beta_spl <- op$beta_spline
```

The beta spline is used on the test data as:

```
op_test <- mySplineTest(test_x,test_y,lambda,p,beta_spl)
```

Training error is calculated as:

```
training_error = sum(abs(train_y - op_train$predicted_y))/length(train_y)
```

Testing error is calculated as:

```
testing_error = sum(abs(test_y - op_test$predicted_y))/length(test_y)
```

The test data is calculated via a Spline Test function as follows:

```
mySplineTest <- function(x, Y, lambda, p = 100, beta_spline){
  n = length(x)
  X <- matrix(x, nrow=n)
  for ( k in (1:(p-1))/p)
    X <- cbind(X,(x>k)*(x-k))
  Yhat <- cbind(rep(1,n),X)%*% beta_spline
  output <- list(beta_spline = beta_spline, predicted_y = Yhat)
  return(output)
}
```

For lambda = 1

```
training_error = 0.08270616
testing_error = 0.103775
```

For lambda = 0.1

```
training_error = 0.07752532
testing_error = 0.06756782
```

For lambda = 0.01

```
training_error = 0.07071239
testing_error = 0.0753857
```

For lambda = 0.001

```
training_error = 0.07105358
testing_error = 0.07870062
```

For lambda = 100

```
training_error = 0.1061344
testing_error = 0.1051672
```

For lambda = 1000

```
training_error = 0.2157368
testing_error = 0.2620837
```

Here, for lambda, which is on the y-axis:

1 -> 0.001
2 -> 0.01
3 -> 0.1
4 -> 1
5 -> 100
6 -> 1000