In [7]:
```
X_rescaled=(X-X.mean(axis=0))/(X.std(axis=0)) #this scales it so that t
he mean is 1 and std is 1
```

In [6]:
```
def RBF(X_train,X_predict,gamma=0.1):
    K=np.zeros((X_train.shape[0],X_predict.shape[0]))
    for i in range(K.shape[0]):
        for j in range(K.shape[1]):
            K[i,j]=np.exp(-gamma*(np.linalg.norm(X_predict[j]-X_train
[i]))**2)
    return K
     #define an rbf kernel yourself

K_10th=RBF(X_10th,X_10th)#then train it with the x every 10 data
```

In [ ]:
```
y_10th=y[::10]
#doing the classic linear regression using linear algebra
#create the model on the training data
A=np.dot(K_10th.T,K_10th)
b=np.dot(K_10th.T,y_10th)
w=np.linalg.solve(A,b)

#now test it with the testing set
K_all=RBF(X_10th,X_rescaled)
yhat=np.dot(w,K_all)
MAE=np.mean(np.abs(y-yhat))
print('MAE = {}'.format(MAE))
```