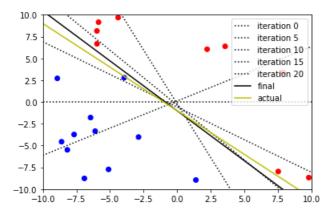
CMPE 257:03 -Homework 1 Learning from data Problem 1.4 and 1.5

- (a) Generate a linearly separated data set of size 20. Plot the examples (xn,yn) as well as target function f on a plane. Be sure to mark the examples from different classes differently and add lables to the plot.
- (b) Run the perceptron learning algorithm on the data set above. Report the number of updates that the algorithm takes before converging. plot the examples {(xn,yn)}, the target function f, and the final hypothesis g in the same figure. Comment on whether f is close to g.

In [26]:

```
import matplotlib.pyplot as plt
import numpy as np
# generate a linearly separated data of size 20-
w_act = np.array([1, 1, 1])
w init = np.array([3, -50, 0])
d = 2
num\_sample = 20
data range = 10
X_train = np.vstack([
    np.ones(num sample),
    np.random.uniform(
       -data_range,
        data range,
        (d, num sample),
    ),
])
y train = evaluate h(w act, X train)
x axis = X train[1, :]
y axis = X train[2, :]
lables = ['r' if y > 0 else 'b' for y in y train]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data range, data range])
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
def evaluate h(w, X):
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
    assert len(w.shape) == 1
    assert len(X.shape) == 2
    assert w.shape[0] == X.shape[0]
    return np.sign(w @ X)
# Run perceptron learning algorithm for the training data set
def run_perceptron(w_init, X_train, y_train, iteration_callback=None):
    w = w init.copy()
    n = 0
    while True:
        y = \text{evaluate } h(w, X \text{ train})
        if iteration_callback:
           iteration_callback(n, w)
        correct = y == y train
        if np.all(correct):
# Print the total number of iterations that let to the generation of final hypothesis g
           print('Total number of iterations: ',n)
            return w
            i = np.argmax(~correct) # indice of first misclassified point
            w = w + y_train[i]*X_train[:, i]
        n = n + 1
def plot hypothesis(x axis, w, *plot args, **plot kwargs):
  m = -w[1]/w[2] if w[2] != 0 else 0
```

```
b = -w[0]/w[2] if w[2] != 0 else 0
    y = xis = m*x = axis + b
    plt.plot(x_axis, y_axis, *plot_args, **plot_kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot_hypothesis(x_hypothesis, w, 'k:', label=label1)
w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data range, data range)
plt.show()
4
```



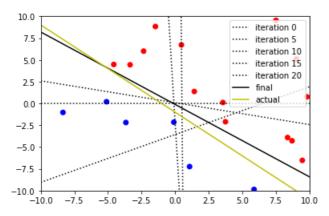
In the above case, we can see that it took 24 iterations for the generation of final hypothesis g. As we can see that the final hyperplane represented by the black line is separating the points perfectly and there are no misclassified points. Hence, f is close to g.

(c) Repeat the step with a different set of randonly generated data of size 20 and compare the results with (b)

In [27]:

```
import matplotlib.pyplot as plt
import numpy as np
# generate a linearly separated data of size 20-
w = np.array([1, 1, 1])
w_{init} = np.array([3, -50, 0])
num sample = 20
data_range = 10
X train = np.vstack([
   np.ones(num sample),
    np.random.uniform(
        -data_range,
        data range,
        (d, num_sample),
    ),
])
y_train = evaluate_h(w_act, X_train)
x axis = X train[1, :]
y_axis = X_train[2, :]
lables = ['r' if y > 0 else 'b' for y in y_train]
plt.scatter(x axis, y axis, c=lables)
x hypothesis = np.array([-data range, data range])
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
```

```
qer evaluate_n(w, A):
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
    assert len(w.shape) == 1
    assert len(X.shape) == 2
    assert w.shape[0] == X.shape[0]
    return np.sign(w @ X)
# Run perceptron learning algorithm for the training data set
def run perceptron(w init, X train, y train, iteration callback=None):
    w = w init.copy()
    n = 0
    while True:
       y = evaluate h(w, X train)
        if iteration callback:
            iteration_callback(n, w)
        correct = y == y_train
        if np.all(correct):
\# Print the total number of iterations that let to the generation of final hypothesis g
            print('Total number of iterations: ',n)
        else:
            i = np.argmax(~correct) # indice of first misclassified point
            w = w + y train[i]*X train[:, i]
        n = n + 1
def plot_hypothesis(x_axis, w, *plot_args, **plot_kwargs):
   m = -w[1]/w[2] if w[2] != 0 else 0
    b = -w[0]/w[2] if w[2] != 0 else 0
    y_axis = m*x_axis + b
    plt.plot(x axis, y axis, *plot args, **plot kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
        label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data range, data range)
plt.ylim(-data range, data range)
plt.show()
4
```

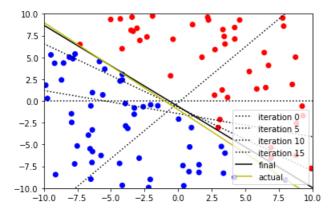


In the above plot, we can see that with a new set of data, we are able to generate a good approximation of final hypothesis and it takes 22 iterations to classify the points correctly in this case.

(d) Repeat the step with a different set of randonly generated data of size 100 and compare the results with (b)

```
import matplotlib.pyplot as plt
import numpy as np
# generate a linearly separated data of size 100-
w = np.array([1, 1, 1])
w init = np.array([3, -50, 0])
d = 2
num sample = 100
data range = 10
X train = np.vstack([
   np.ones(num_sample),
    np.random.uniform(
       -data range,
       data_range,
        (d, num sample),
])
y train = evaluate h(w act, X train)
x_axis = X_train[1, :]
y axis = X train[2, :]
lables = ['r' if y > 0 else 'b' for y in y_train]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
def evaluate_h(w, X):
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
   assert len(w.shape) == 1
    assert len(X.shape) == 2
   assert w.shape[0] == X.shape[0]
   return np.sign(w @ X)
# Run perceptron learning algorithm for the training data set
def run perceptron(w init, X train, y train, iteration callback=None):
   w = w_{init.copy()}
   n = 0
    while True:
       y = evaluate_h(w, X_train)
        if iteration callback:
           iteration callback(n, w)
        correct = y == y_train
        if np.all(correct):
# Print the total number of iterations that let to the generation of final hypothesis g
            print('Total number of iterations: ',n)
            return w
        else:
           i = np.argmax(~correct) # indice of first misclassified point
           w = w + y train[i]*X train[:, i]
        n = n + 1
def plot_hypothesis(x_axis, w, *plot_args, **plot_kwargs):
   m = -w[1]/w[2] if w[2] != 0 else 0
   b = -w[0]/w[2] if w[2] != 0 else 0
    y = m*x = m*x = b
   plt.plot(x axis, y axis, *plot args, **plot kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
   if n % 5 == 0:
        label1 = 'iteration {}'.format(n)
        plot_hypothesis(x_hypothesis, w, 'k:', label=label1)
w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot hypothesis(x hypothesis, w final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.xlim(-data range, data range)
```

```
plt.ylim(-data_range, data_range)
plt.show()
4
```



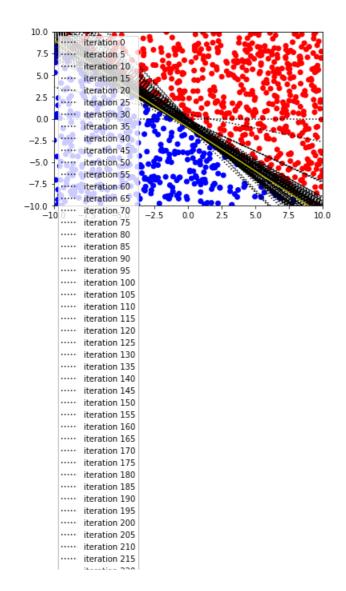
In the above plot, we can see that with a new set of 100 data, we are able to generate a good approximation of final hypothesis and it takes 19 iterations to classify the points correctly in this case.

(e)Repeat the step with a different set of randonly generated data of size 1000 and compare the results with (b)

In [32]:

```
import matplotlib.pyplot as plt
import numpy as np
# generate a linearly separated data of size 1000-
w = np.array([1, 1, 1])
w_{init} = np.array([3, -50, 0])
d = 2
num sample = 1000
data_range = 10
X train = np.vstack([
   np.ones(num sample),
    np.random.uniform(
        -data range,
        data range,
        (d, num sample),
    ),
])
y_train = evaluate_h(w_act, X_train)
x axis = X train[1, :]
y axis = X train[2, :]
lables = ['r' if y > 0 else 'b' for y in y_train]
plt.scatter(x_axis, y_axis, c=lables)
x hypothesis = np.array([-data range, data range])
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
def evaluate_h(w, X):
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
    assert len(w.shape) == 1
    assert len(X.shape) == 2
    assert w.shape[0] == X.shape[0]
   return np.sign(w @ X)
# Run perceptron learning algorithm for the training data set
def run perceptron(w init, X train, y train, iteration callback=None):
    w = w init.copy()
    n = 0
    while True:
```

```
y = evaluate h(w, x train)
        if iteration callback:
            iteration callback(n, w)
        correct = y == y train
        if np.all(correct):
# Print the total number of iterations that let to the generation of final hypothesis g
            print('Total number of iterations: ',n)
            return w
        else:
           i = np.argmax(~correct) # indice of first misclassified point
           w = w + y train[i]*X train[:, i]
        n = n + 1
def plot_hypothesis(x_axis, w, *plot_args, **plot_kwargs):
   m = -w[1]/w[2] if w[2] != 0 else 0
    b = -w[0]/w[2] if w[2] != 0 else 0
    y = xis = m*x = axis + b
    plt.plot(x_axis, y_axis, *plot_args, **plot_kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
   if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
w final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data range, data range)
plt.ylim(-data range, data range)
plt.show()
4
```



····· iteration 220 ····· iteration 225 ····· iteration 230 ····· iteration 235 ····· iteration 240 ····· iteration 245 ····· iteration 250 ····· iteration 255 ····· iteration 260 ····· iteration 265 ····· iteration 270 ····· iteration 275 ····· iteration 280 ····· iteration 285 ····· iteration 290 ····· iteration 295 ····· iteration 300 ····· iteration 305 ····· iteration 310 ····· iteration 315 ····· iteration 320 ····· iteration 325 ····· iteration 330 ····· iteration 335 ····· iteration 340 ····· iteration 345 ····· iteration 355 ····· iteration 360 ····· iteration 365 ····· iteration 370 ····· iteration 375 ····· iteration 380 ····· iteration 385 ····· iteration 390 ····· iteration 395 ····· iteration 400 ····· iteration 405 ····· iteration 410 ····· iteration 415 ····· iteration 420 ····· iteration 425 ····· iteration 430 ····· iteration 435 ····· iteration 440 ····· iteration 445 ····· iteration 450 ····· iteration 455 ····· iteration 460 ····· iteration 465 ····· iteration 470 ····· iteration 475 ····· iteration 480 ····· iteration 485 ····· iteration 490 ····· iteration 495 ····· iteration 500 ····· iteration 505 ····· iteration 510 ····· iteration 515 ····· iteration 520 ····· iteration 525 ····· iteration 530 ····· iteration 535 ····· iteration 545 ····· iteration 550 final

— actual

In the above plot, we can see that with a new set of 100 data, we are able to generate a good approximation of final hypothesis and it takes 554 iterations to classify the points correctly in this case.

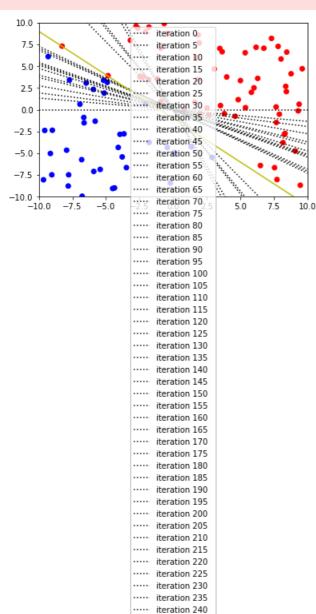
Problem 1.5 - The perceptron learning algorithm works like this: in each iteration t, pick a randon (x(t), y(t)) and compute the 'signal' s(t) = wT(t) x(t). if y(t).s(t) <=0, update w by w(t+1) <- w(t) + y(t). x(t);

One may say that the algorithm does not take the closeness between s(t) and y(t) into account. so if y(t). $s(t) \le 1$, update w by : $w(t+1) \le w(t) + n (y(t) - s(t)).x(t)$

(a) Generate training set of 100 and test set of 1000. run the algorithm by taking n =100 on training set until 1000 updates. plot training set and target function. Report the error on test set

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
# generate a linearly separated training data of size 100 and test data of size 1000 -
w act = np.array([1, 1, 1])
w init = np.array([3, -50, 0])
d = 2
num sample = 100
data range = 10
num_sample_test =1000
error = 0
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
def evaluate h(w, X):
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
    assert len(w.shape) == 1
   assert len(X.shape) == 2
    assert w.shape[0] == X.shape[0]
   return np.sign(w @ X)
X_train = np.vstack([
   np.ones(num_sample),
    np.random.uniform(
        -data_range,
        data_range,
        (d, num_sample),
    ),
])
X test = np.vstack([
   np.ones(num sample test),
   np.random.uniform(
        -data range,
        data_range,
        (d, num sample test),
])
y_train = evaluate_h(w_act, X_train)
y_test = evaluate_h(w_act, X_test)
x axis = X train[1, :]
y_axis = X_train[2, :]
lables = ['r' if y > 0 else 'b' for y in y train]
plt.scatter(x axis, y axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Run perceptron learning algorithm for the training data set
def run perceptron(w init, X train, y train, iteration callback=None):
   w = w_init.copy()
   n = 0
    while True:
        y = evaluate_h(w, X_train)
        if iteration callback:
           iteration callback(n, w)
        correct = y == y_train
# Calculate error after 1000 updates
        if n == 1000:
            return w
        else:
\# calculate weight considering the closeness between s and y\_train
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X train[:,i]
            d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 100 * (y_train[i] - s) * X_train[:, i]
                error = (y_train[i] - s)
       n = n + 1
```

```
def plot_hypothesis(x_axis, w, *plot_args, **plot_kwargs):
    m = -w[1]/w[2] if w[2] != 0 else 0
    b = -w[0]/w[2] if w[2] != 0 else 0
    y = m*x = m*x = b
    plt.plot(x axis, y axis, *plot args, **plot kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot_hypothesis(x_hypothesis, w, 'k:', label=label1)
w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot hypothesis(x hypothesis, w act, 'y', label='actual')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data range, data range)
plt.show()
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:71: RuntimeWarning: overflow
encountered in multiply
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:23: RuntimeWarning: invalid value
encountered in sign
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:78: RuntimeWarning: invalid value
encountered in double scalars
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:71: RuntimeWarning: invalid value
encountered in add
```



		iteration	245
		iteration	
		iteration	255
	•	iteration	260
		iteration	265
		iteration iteration	270 275
		iteration	280
		iteration	285
	• • •	iteration	290
	•	iteration	295
		iteration	300
		iteration iteration	305 310
		iteration	315
		iteration	320
		iteration	325
		iteration	330
		iteration	335
	•	iteration	340
		iteration iteration	345
		iteration	350 355
		iteration	360
		iteration	365
	•	iteration	370
	•	iteration	375
	•	iteration	380
		iteration	385
		iteration	390
		iteration iteration	395
		iteration	
		iteration	
		iteration	
		iteration	420
		iteration	425
	•	iteration	
	•	iteration	
		iteration	
		iteration iteration	
		iteration	475
		iteration	480
		iteration	
		iteration	
		iteration iteration	
		iteration	520
		iteration	525
		iteration	
		iteration	
		iteration	
1		itoration.	
		iteration iteration	545
		iteration iteration iteration	545 550
		iteration	545 550 555
		iteration iteration	545 550 555 560 565
		iteration iteration iteration iteration iteration	545 550 555 560 565 570
		iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575
		iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580
		iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585
		iteration iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585 590
		iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585 590
		iteration iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585 590 595 600
		iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585 590 595 600 605
		iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration iteration	545 550 555 560 565 570 575 580 585 590 595 600 605 610
		iteration iterat	545 550 555 560 565 570 575 580 585 590 605 600 605 610 615 620
		iteration iterat	545 550 555 560 565 570 575 580 585 590 600 605 610 615 620 625
		iteration iterat	545 550 555 560 565 570 575 580 585 590 605 610 615 620 625 630
		iteration iterat	545 550 555 560 565 570 575 580 585 590 600 605 610 620 625 630 635
		iteration iterat	545 550 555 560 565 570 575 580 595 600 605 610 625 630 635 640
		iteration iterat	545 550 555 560 565 570 575 580 585 590 605 610 615 620 625 630 635 640 645
		iteration iterat	545 550 555 560 565 570 575 580 585 590 605 610 615 620 625 630 645 640 645 650
		iteration iterat	545 550 555 560 565 570 575 580 595 600 605 610 625 630 640 645 650 655
		iteration iterat	545 550 555 560 565 570 575 580 595 600 605 610 625 630 640 645 650 655 660
		iteration iterat	545 550 555 560 565 570 575 580 595 600 605 610 625 630 645 640 645 650 665 660 665 660 665 660 665 660
		iteration iterat	545 550 555 560 565 570 585 590 600 605 610 625 630 640 645 650 665 660 665 670 675

```
ILCI GLIVIT VVV
····· iteration 685
····· iteration 690
····· iteration 695
····· iteration 700
····· iteration 710
····· iteration 715
····· iteration 720
····· iteration 725
····· iteration 730
····· iteration 735
····· iteration 740
····· iteration 745
····· iteration 750
····· iteration 755
····· iteration 760
····· iteration 765
····· iteration 770
····· iteration 775
····· iteration 780
····· iteration 785
····· iteration 790
····· iteration 795
····· iteration 800
····· iteration 805
····· iteration 810
····· iteration 815
····· iteration 820
····· iteration 825
····· iteration 830
····· iteration 835
····· iteration 840
····· iteration 845
····· iteration 850
····· iteration 855
····· iteration 860
····· iteration 865
····· iteration 870
····· iteration 875
····· iteration 880
····· iteration 885
····· iteration 890
····· iteration 895
····· iteration 900
····· iteration 905
····· iteration 910
····· iteration 915
····· iteration 920
····· iteration 925
····· iteration 930
····· iteration 935
····· iteration 940
····· iteration 945
····· iteration 950
····· iteration 955
····· iteration 960
····· iteration 965
····· iteration 975
····· iteration 980
····· iteration 985
····· iteration 990
····· iteration 995
····· iteration 1000
  — final
```

The plot above is the plot for training data set for n = 100.

actual

In [111]:

```
# Run perceptron learning algorithm for the test data set
def run_perceptron(w_init, X_train, y_train, iteration_callback=None):
    w = w_init.copy()
    n = 0
    while True:
        y = evaluate_h(w, X_train)
        if iteration_callback:
            iteration_callback(n, w)
        correct = y == y_train
        if np.all(correct):
# Print the total number of iterations that let to the generation of final hypothesis g
```

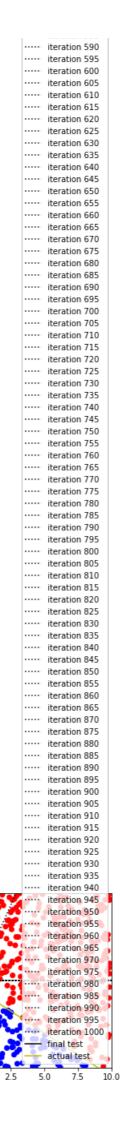
```
print('Total number of iterations: ',n)
             return w
        if n == 1000:
            error = np.absolute(error)
            print("Error on test data for n = 100: ",error[0]/1000)
            return w
        else:
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X_train[:,i]
            d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 100 * (y_train[i] - s) * X_train[:, i]
                error = (y train[i] - s)
       n = n + 1
x axis = X test[1, :]
y axis = X test[2, :]
lables = ['r' if y > 0 else 'b' for y in y test]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
        label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
# Run Perceptron learning algorith for test data set
w_final = run_perceptron(w_init, X_test, y_test, plot_iteration)
plot hypothesis(x hypothesis, w final, 'k', label='final test')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual test')
plt.legend()
plt.xlim(-data range, data range)
plt.ylim(-data_range, data_range)
plt.show()
4
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:27: RuntimeWarning: overflow
encountered in multiply
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:27: RuntimeWarning: invalid value
encountered in add
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:23: RuntimeWarning: invalid value
encountered in sign
```

Error on test data for n = 100: 8.424947340550924e+166

····· iteration 0 ···· iteration 5 ····· iteration 10 ····· iteration 15 ····· iteration 20 ····· iteration 25 ····· iteration 30 ····· iteration 35 ····· iteration 40 ····· iteration 45 ····· iteration 50 ····· iteration 55 ····· iteration 60 ····· iteration 65 ····· iteration 70 ····· iteration 75 ····· iteration 80 ····· iteration 85 ····· iteration 90 ····· iteration 95 ····· iteration 100 ····· iteration 105 ····· iteration 110 ····· iteration 115 ····· iteration 120 ····· iteration 125 ····· iteration 130 ····· iteration 135 ····· iteration 140 ····· iteration 145 ····· iteration 150

	itemation	150
	iteration	155 160
	iteration	165
	iteration	170
	iteration	175
	iteration	180
	iteration	185
	iteration	190
	iteration	195
	iteration	200
	iteration	205
	iteration	210
	iteration	215
	iteration	220
	iteration iteration	225 230
	iteration	235
	iteration	240
	iteration	245
	iteration	250
	iteration	255
	iteration	260
	iteration	265
	iteration	270
	iteration	275
	iteration	280
	iteration	285
	iteration	290
	iteration	295 300
	iteration iteration	305
	iteration	310
	iteration	315
	iteration	320
	iteration	325
	iteration	330
	iteration	335
	iteration	340
	iteration	345
	iteration	350
	iteration	355
	iteration	360 365
	iteration	370
	iteration	375
	iteration	380
	iteration	385
	iteration	390
	iteration	395
	iteration	
	iteration	
	iteration	
	iteration iteration	
	iteration	445
	iteration	450
	iteration	
	iteration	
	iteration	
	iteration iteration	
	iteration	495
	iteration	500
	iteration	505
	iteration	
	iteration iteration	
	iteration	
	iteration iteration	
1	TOTAL PROPERTY.	

···· iteration 585



7.5

5.0

2.5

0.0

-2.5

-5.0

-7.5

-10.0

-10.0

-5.0

-2.5

0.0

The above plot is the output of test data for the previously trained data set with n = 100. The error rate for this case is 8.42e + 166 which is equivalent to 10.3

(b) The previous problem is now solved for n=1

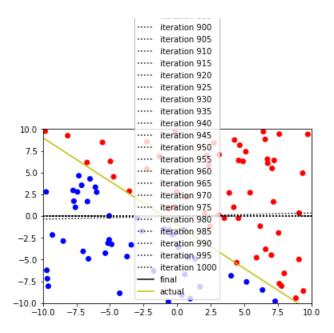
```
In [121]:
```

```
y train = evaluate h(w act, X train)
y test = evaluate h(w act, X test)
x axis = X_train[1, :]
y axis = X_train[2, :]
lables = ['r' if y > 0 else 'b' for y in y train]
plt.scatter(x axis, y axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Run perceptron learning algorithm for the training data set
def run_perceptron(w_init, X_train, y_train, iteration_callback=None):
   w = w_{init.copy()}
    n = 0
    while True:
        y = evaluate h(w, X train)
        if iteration callback:
            iteration_callback(n, w)
        correct = y == y_train
# Calculate error after 1000 updates
       if n == 1000:
            return w
        else:
# calculate weight considering the closeness between s and y train
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X_train[:,i]
            d = y_train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 1 * (y_train[i] - s) * X_train[:, i]
                error = (y_train[i] - s)
        n = n + 1
def plot hypothesis(x axis, w, *plot args, **plot kwargs):
    m = -w[1]/w[2] if w[2] != 0 else 0
    b = -w[0]/w[2] if w[2] != 0 else 0
    y axis = m*x axis + b
    plt.plot(x_axis, y_axis, *plot_args, **plot_kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
w final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot hypothesis(x hypothesis, w final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data range, data range)
plt.ylim(-data_range, data_range)
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:28: RuntimeWarning: overflow
encountered in multiply
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:28: RuntimeWarning: invalid value
encountered in add
C:\Users\kashi\Anaconda3\lib\site-packages\ipykernel launcher.py:23: RuntimeWarning: invalid value
encountered in sign
```

```
iteration 0
iteration 5
iteration 10
iteration 15
iteration 20
```

 iteration	25
 iteration	30
 iteration	35
 iteration	40
 iteration	45
 iteration	50
 iteration	55
 iteration	60
 iteration	65
 iteration	70
 iteration	75
 iteration iteration	80 85
 iteration	90
 iteration	95
 iteration	100
 iteration	105
 iteration	110
 iteration	115
 iteration	120
 iteration	125
 iteration	130
 iteration	135
 iteration	140
 iteration	145
 iteration	150
 iteration	155
 iteration	160
 iteration	165
 iteration	170
 iteration	175
 iteration iteration	180 185
 iteration	190
 iteration	195
 iteration	200
 iteration	205
 iteration	210
 iteration	215
 iteration	220
 iteration	225
 iteration	230
 iteration	235
 iteration	240
 iteration	245
 iteration	250
 iteration	255
 iteration	260
 iteration	265
 iteration	
 iteration	300
 iteration	305
 iteration	310
 iteration	315
 iteration	320
 iteration	
 iteration	
 iteration	
 iteration iteration	
 iteration	395
 iteration	400
 iteration	405
 iteration	
 iteration iteration	
 iteration	
 iteration	

 icciacioni	400
 iteration iteration	
 iteration	
 iteration	
 iteration	
 iteration	490
 iteration	
 iteration	
 iteration iteration	
 iteration	
 iteration	
 iteration	525
 iteration	530
 iteration	
 iteration	
 iteration iteration	
 iteration	
 iteration	
 iteration	565
 iteration	570
 iteration	
 iteration	
 iteration iteration	
 iteration	615
 iteration	
 iteration	
 iteration	
 iteration	
 iteration iteration	
 iteration	670
 iteration	675
 iteration	
 iteration	
 iteration	
 iteration iteration	
 iteration	
 iteration	
 iteration	715
 iteration	720
 iteration	725
 iteration	
 iteration	
 iteration	
 iteration iteration	
 iteration	
 iteration iteration	
 iteration	04U
iteration	845
 iteration iteration	
 iteration	850
	850 855
 iteration iteration	850 855 860
 iteration iteration iteration	850 855 860 865
 iteration iteration iteration iteration	850 855 860 865 870
 iteration iteration iteration iteration iteration iteration iteration	850 855 860 865 870 875 880
 iteration iteration iteration iteration iteration iteration iteration iteration	850 855 860 865 870 875 880 885
 iteration iteration iteration iteration iteration iteration iteration	850 855 860 865 870 875 880 885 890



The plot above takes into account n =1 for training data

In [141]:

```
# Run perceptron learning algorithm for the test data set
def run perceptron(w init, X train, y train, iteration callback=None):
    w = w init.copy()
    while True:
        y = evaluate h(w, X train)
        if iteration callback:
            iteration callback(n, w)
        correct = y == y_train
         if np.all(correct):
# Print the total number of iterations that let to the generation of final hypothesis g
             print('Total number of iterations: ',n)
             return w
        if n == 1000:
            error = np.absolute(error)
            print("Error on test data for n = 1: ",error[0]/1000)
            return w
        else:
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X_train[:,i]
            d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 1 * (y_train[i] - s) * X_train[:, i]
                 error = (y_train[i] - s)
        n = n + 1
x_axis = X_test[1, :]
y_axis = X_test[2, :]
lables = ['r' if y > 0 else 'b' for y in y_test]
plt.scatter(x_axis, y_axis, c=lables)
x hypothesis = np.array([-data range, data range])
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
        label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
# Run Perceptron learning algorith for test data set
w_final = run_perceptron(w_init, X_test, y_test, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final test')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual test')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data_range, data_range)
```

Error on test data for n = 1: 0.0

iteration 0 ····· iteration 5 iteration 10 iteration 15 iteration 20 iteration 25 iteration 30 iteration 35 iteration 40 iteration 45 iteration 50 iteration 55 iteration 60 iteration 65 iteration 70 iteration 75 iteration 80 iteration 85 iteration 90 iteration 95 iteration 100 iteration 105 iteration 110 iteration 115 iteration 120 iteration 125 iteration 130 iteration 135 iteration 140 iteration 145 iteration 150 iteration 155 iteration 160 iteration 165 iteration 170 iteration 175 iteration 180 iteration 185 iteration 190 iteration 195 iteration 200 iteration 205 iteration 210 iteration 215 iteration 220 iteration 225 iteration 230 iteration 235 iteration 240 iteration 245 iteration 250 iteration 255 iteration 260 iteration 265 iteration 270 iteration 275 iteration 280 iteration 285 iteration 290 iteration 295 iteration 300 iteration 305 iteration 310 iteration 315 iteration 320 iteration 325 iteration 330 iteration 335 iteration 340 iteration 345 iteration 350 iteration 355 iteration 360 iteration 365 ····· iteration 370 iteration 375 iteration 380 ····· iteration 385 iteration 390 ····· iteration 395 ····· iteration 400

iteration 405
 iteration 410
 iteration 415
 iteration 420
 iteration 425
 iteration 430
 iteration 435
 iteration 440
 iteration 445
 iteration 450
 iteration 455
 iteration 460
 iteration 465
 iteration 470
 iteration 475
 iteration 480
 iteration 485
 iteration 490
 iteration 495
 iteration 500
 iteration 505
 iteration 510
 iteration 515
 iteration 520
 iteration 525
 iteration 530
 iteration 535
 iteration 540
 iteration 545
 iteration 545
 iteration 560
 iteration 565
 iteration 570
 iteration 575
 iteration 580
 iteration 585
 iteration 590
 iteration 595
 iteration 600
 iteration 605
 iteration 610
 iteration 615
 iteration 620
 iteration 625
 iteration 630
 iteration 635
 iteration 640
 iteration 645
 iteration 650
 iteration 655
 iteration 655
iteration 660
 iteration 660 iteration 665
 iteration 660 iteration 665 iteration 670
 iteration 660 iteration 665 iteration 670 iteration 675
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 685
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 685 iteration 690
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 695
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 695 iteration 700
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 685 iteration 690 iteration 700 iteration 700
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 690 iteration 700 iteration 705 iteration 710
 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 695 iteration 700 iteration 705 iteration 705 iteration 710 iteration 715
iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 695 iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 720
iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 685 iteration 695 iteration 700 iteration 705 iteration 715 iteration 715 iteration 725 iteration 720 iteration 720 iteration 725
iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 695 iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 720
iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 685 iteration 695 iteration 700 iteration 705 iteration 715 iteration 715 iteration 725 iteration 720 iteration 720 iteration 725
iteration 660 iteration 665 iteration 670 iteration 680 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 715 iteration 720 iteration 720 iteration 720 iteration 730
iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 700 iteration 705 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725 iteration 730 iteration 733
iteration 660 iteration 670 iteration 675 iteration 675 iteration 685 iteration 690 iteration 700 iteration 705 iteration 710 iteration 710 iteration 725 iteration 725 iteration 725 iteration 730 iteration 735 iteration 740
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 715 iteration 725 iteration 725 iteration 730 iteration 730 iteration 730 iteration 730 iteration 730 iteration 740 iteration 740
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 690 iteration 690 iteration 700 iteration 705 iteration 715 iteration 720 iteration 720 iteration 735 iteration 735 iteration 735 iteration 740 iteration 745 iteration 745
iteration 660 iteration 675 iteration 675 iteration 680 iteration 680 iteration 690 iteration 695 iteration 700 iteration 705 iteration 715 iteration 720 iteration 725 iteration 735 iteration 740 iteration 745 iteration 745 iteration 755 iteration 750 iteration 755 iteration 760
iteration 660 iteration 675 iteration 670 iteration 680 iteration 680 iteration 695 iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 725 iteration 730 iteration 730 iteration 745 iteration 745 iteration 750 iteration 755 iteration 750 iteration 750 iteration 760 iteration 760
iteration 660 iteration 670 iteration 670 iteration 680 iteration 680 iteration 690 iteration 695 iteration 700 iteration 701 iteration 710 iteration 720 iteration 720 iteration 730 iteration 730 iteration 745 iteration 745 iteration 745 iteration 745 iteration 755 iteration 760 iteration 760 iteration 760 iteration 770
iteration 660 iteration 670 iteration 670 iteration 675 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 720 iteration 725 iteration 730 iteration 730 iteration 740 iteration 755 iteration 755 iteration 750 iteration 755 iteration 765 iteration 765 iteration 765 iteration 7670 iteration 7770 iteration 7770 iteration 7771
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 725 iteration 730 iteration 730 iteration 740 iteration 750 iteration 750 iteration 750 iteration 750 iteration 760 iteration 765 iteration 765 iteration 765 iteration 7760 iteration 7770 iteration 7780 iteration 7780
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 720 iteration 720 iteration 735 iteration 730 iteration 740 iteration 745 iteration 750 iteration 750 iteration 750 iteration 765 iteration 765 iteration 770 iteration 775 iteration 777 iteration 778 iteration 7780 iteration 7780 iteration 780 iteration 780 iteration 787
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 690 iteration 700 iteration 710 iteration 720 iteration 720 iteration 735 iteration 735 iteration 740 iteration 755 iteration 760 iteration 775 iteration 7760 iteration 775 iteration 770 iteration 775 iteration 7785 iteration 780 iteration 780 iteration 780 iteration 7790
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 695 iteration 695 iteration 695 iteration 700 iteration 705 iteration 715 iteration 720 iteration 735 iteration 735 iteration 740 iteration 755 iteration 755 iteration 760 iteration 775 iteration 780 iteration 780 iteration 780 iteration 780 iteration 790 iteration 795
iteration 660 iteration 675 iteration 680 iteration 680 iteration 680 iteration 695 iteration 695 iteration 695 iteration 700 iteration 705 iteration 715 iteration 720 iteration 735 iteration 735 iteration 740 iteration 745 iteration 750 iteration 760 iteration 775 iteration 776 iteration 775 iteration 785 iteration 770 iteration 785 iteration 770 iteration 785 iteration 780 iteration 780 iteration 780 iteration 780 iteration 780 iteration 800
iteration 660 iteration 675 iteration 680 iteration 680 iteration 680 iteration 695 iteration 695 iteration 700 iteration 700 iteration 715 iteration 725 iteration 735 iteration 735 iteration 740 iteration 745 iteration 750 iteration 750 iteration 760 iteration 770 iteration 770 iteration 770 iteration 770 iteration 770 iteration 785 iteration 770 iteration 785 iteration 780 iteration 780 iteration 780 iteration 780 iteration 790 iteration 800 it
iteration 660 iteration 670 iteration 675 iteration 675 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 720 iteration 725 iteration 730 iteration 730 iteration 740 iteration 745 iteration 755 iteration 755 iteration 760 iteration 775 iteration 780 iteration 780 iteration 780 iteration 780 iteration 780 iteration 800 iteration 805 iteration 805 iteration 805 iteration 805 iteration 810
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 680 iteration 690 iteration 700 iteration 700 iteration 705 iteration 725 iteration 725 iteration 730 iteration 730 iteration 740 iteration 745 iteration 750 iteration 750 iteration 750 iteration 750 iteration 760 iteration 765 iteration 775 iteration 775 iteration 775 iteration 775 iteration 775 iteration 785 iteration 780 iteration 785 iteration 795 iteration 795 iteration 800 iteration 805 iteration 805 iteration 805 iteration 805 iteration 805 iteration 810 iteration 815
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 725 iteration 730 iteration 740 iteration 740 iteration 750 iteration 750 iteration 750 iteration 750 iteration 760 iteration 765 iteration 770 iteration 775 iteration 770 iteration 775 iteration 770 iteration 775 iteration 785 iteration 785 iteration 790 iteration 790 iteration 800 iteration 800 iteration 801 iteration 815 iteration 815 iteration 812
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 720 iteration 735 iteration 740 iteration 745 iteration 750 iteration 765 iteration 765 iteration 765 iteration 770 iteration 775 iteration 780 iteration 800 iteration 800 iteration 805 iteration 810 iteration 810 iteration 810 iteration 815 iteration 810 iteration 812 iteration 820 iteration 820 iteration 825
iteration 660 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 690 iteration 700 iteration 700 iteration 710 iteration 725 iteration 725 iteration 730 iteration 740 iteration 740 iteration 750 iteration 750 iteration 750 iteration 750 iteration 760 iteration 765 iteration 770 iteration 775 iteration 770 iteration 775 iteration 770 iteration 775 iteration 785 iteration 785 iteration 790 iteration 790 iteration 800 iteration 800 iteration 801 iteration 815 iteration 815 iteration 812

····· iteration 855 ····· iteration 860 ····· iteration 865 ····· iteration 870 ····· iteration 875 ····· iteration 880 ····· iteration 885 ····· iteration 890 ····· iteration 895 ····· iteration 900 ····· iteration 905 ····· iteration 910 ····· iteration 915 ····· iteration 920 ····· iteration 925 ····· iteration 930 ····· iteration 935 ····· iteration 940 10.0 ···· iteration 945 iteration 950 iteration 955 iteration 960 5.0 iteration 965 iteration 970 iteration 975 0.0 iteration 980 iteration 985 -2.5 iteration 995 -5.0iteration 1000 final test -7.5actual test -10.0 -10.05.0 7.5

In the above plot the test data is plotted for n =1 and error is measured. The error rate in this case is 0

····· iteration 840 ····· iteration 845 iteration 850

(c) The problem is now run for n = 0.01

In [122]:

```
y_train = evaluate_h(w_act, X_train)
y_test = evaluate_h(w_act, X_test)
x axis = X train[1, :]
y axis = X_train[2, :]
lables = ['r' if y > 0 else 'b' for y in y_train]
plt.scatter(x axis, y axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Run perceptron learning algorithm for the training data set
def run perceptron(w init, X train, y train, iteration callback=None):
   w = w init.copy()
   n = 0
    while True:
        y = evaluate_h(w, X_train)
        if iteration callback:
           iteration callback(n, w)
        correct = y == y_train
# Calculate error after 1000 updates
        if n == 1000:
            return w
        else:
# calculate weight considering the closeness between s and y train
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X train[:,i]
            d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 0.01 * (y_train[i] - s) * X_train[:, i]
                error = (y_train[i] - s)
        n = n + 1
```

```
def plot_hypothesis(x_axis, w, *plot_args, **plot_kwargs):
    m = -w[1]/w[2] if w[2] != 0 else 0
   b = -w[0]/w[2] if w[2] != 0 else 0
    y = x = m \times x = x = b
    plt.plot(x axis, y axis, *plot args, **plot kwargs)
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot_hypothesis(x_hypothesis, w, 'k:', label=label1)
w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data range, data range)
plt.show()
```

```
····· iteration 0
····· iteration 5
····· iteration 10
····· iteration 15
····· iteration 20
····· iteration 25
····· iteration 30
····· iteration 35
····· iteration 40
····· iteration 45
····· iteration 50
····· iteration 55
····· iteration 60
····· iteration 65
····· iteration 70
····· iteration 75
····· iteration 80
····· iteration 85
····· iteration 90
····· iteration 95
····· iteration 100
····· iteration 105
····· iteration 110
····· iteration 115
····· iteration 120
····· iteration 125
····· iteration 130
····· iteration 135
····· iteration 140
····· iteration 145
····· iteration 150
····· iteration 155
····· iteration 160
····· iteration 165
····· iteration 170
····· iteration 175
····· iteration 180
····· iteration 185
····· iteration 190
····· iteration 195
····· iteration 200
····· iteration 205
····· iteration 210
····· iteration 215
····· iteration 220
····· iteration 225
····· iteration 230
····· iteration 235
····· iteration 240
····· iteration 245
····· iteration 250
····· iteration 255
····· iteration 260
····· iteration 265
····· iteration 270
····· iteration 275
····· iteration 280
····· iteration 285
····· iteration 290
····· iteration 295
····· iteration 300
····· iteration 305
····· iteration 310
```

	iteration	315
	iteration	320
	iteration	325
	iteration	330
	iteration iteration	335 340
	iteration	345
	iteration	350
	iteration	355
	iteration	360
	iteration iteration	365 370
	iteration	375
	iteration	380
	iteration	385
	iteration	390
	iteration iteration	395 400
	iteration	405
	iteration	410
	iteration	415
	iteration	420
	iteration iteration	425 430
	iteration	435
	iteration	440
	iteration	445
	iteration	450
	iteration iteration	455 460
		465
	iteration	470
	iteration	475
	iteration	480
	iteration iteration	485 490
	iteration	495
	iteration	500
	iteration	505
	iteration	510
	iteration iteration	515 520
	iteration	525
	iteration	530
	iteration	535
	iteration iteration	540 545
	iteration	550
	iteration	555
	iteration	560
	iteration	565
	iteration iteration	570 575
	iteration	580
	iteration	585
	iteration	590
	iteration	595
	iteration iteration	600 605
	iteration	610
	iteration	615
	iteration	620
	iteration iteration	625 630
	iteration	635
	iteration	640
	iteration	645
	iteration iteration	650 655
	iteration	660
	iteration	665
	iteration	670
	iteration	675
	iteration iteration	680 685
	iteration	690
	iteration	695
	iteration	700
	iteration	705 710
	iteration iteration	715
	iteration	720
	iteration	725
	iteration	730
	iteration iteration	735 740
	iteration	745
1	itaration	750

```
····· iteration 755
         ····· iteration 760
         ····· iteration 770
              iteration 775
         .....
         ····· iteration 780
         ····· iteration 785
              iteration 790
         ····· iteration 795
         .....
              iteration 800
         ····· iteration 805
         ····· iteration 810
              iteration 815
         ····· iteration 820
         ····· iteration 825
         ····· iteration 830
         ····· iteration 835
              iteration 840
         ····· iteration 845
         ····· iteration 850
         ····· iteration 855
         ····· iteration 860
              iteration 865
         ····· iteration 870
         ····· iteration 875
         .....
              iteration 880
              iteration 885
              iteration 890
         ····· iteration 895
         .....
              iteration 900
         .....
              iteration 905
         ····· iteration 910
              iteration 915
         ····· iteration 920
         ····· iteration 925
         ····· iteration 930
         ····· iteration 935
         ···· iteration 940
 10.0 iteration 945
        ····· iteration 950
              iteration 955
       ..... iteration 960
         ····· iteration 965
       ..... iteration 970
        ···· iteration 975
        ····· iteration 980
        iteration 985
         ..... iteration 990
         ···· iteration 995
              iteration 1000
              final
              actual
-10.0 -
     -10.0
            -7.5 -5.0
```

····· iteration /50

The plot above takes into account n =1 for training data

In [132]:

7.5

5.0

2.5

0.0

-5.0

-7.5

```
\# Run perceptron learning algorithm for the test data set
def run perceptron(w init, X train, y train, iteration callback=None):
   w = w_{init.copy}()
   n = 0
    while True:
        y = \text{evaluate } h(w, X \text{ train})
        if iteration callback:
            iteration callback(n, w)
        correct = y == y_train
         if np.all(correct):
 Print the total number of iterations that let to the generation of final hypothesis g
#
            print('Total number of iterations: ',n)
             return w
        if n == 1000:
            error = np.absolute(error)
            print("Error on test data for n = 0.01: ",error[0]/1000)
            return w
        else:
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X train[:,i]
```

```
d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
               w = w + 0.01 * (y_train[i] - s) * X_train[:, i]
                error = (y train[i] - s)
       n = n + 1
x axis = X test[1, :]
y_axis = X_test[2, :]
lables = ['r' if y > 0 else 'b' for y in y_test]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
        label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
# Run Perceptron learning algorith for test data set
w_final = run_perceptron(w_init, X_test, y_test, plot_iteration)
plot hypothesis(x hypothesis, w final, 'k', label='final test')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual test')
plt.legend()
plt.xlim(-data range, data range)
plt.ylim(-data_range, data_range)
plt.show()
4
```

Error on test data for n = 0.01: 0.0009270516925850184

```
····· iteration 0
···· iteration 5
····· iteration 10
····· iteration 15
····· iteration 20
····· iteration 25
····· iteration 30
····· iteration 35
····· iteration 40
····· iteration 45
····· iteration 50
····· iteration 55
····· iteration 60
····· iteration 65
····· iteration 70
····· iteration 75
····· iteration 80
····· iteration 85
····· iteration 90
····· iteration 95
····· iteration 100
····· iteration 105
····· iteration 110
····· iteration 115
····· iteration 120
····· iteration 125
····· iteration 130
····· iteration 135
····· iteration 140
····· iteration 145
····· iteration 150
····· iteration 155
····· iteration 160
····· iteration 165
····· iteration 170
····· iteration 175
····· iteration 180
····· iteration 185
····· iteration 190
····· iteration 195
····· iteration 200
····· iteration 205
····· iteration 210
····· iteration 215
····· iteration 220
····· iteration 225
····· iteration 230
····· iteration 235
····· iteration 240
····· iteration 245
····· iteration 250
..... itaration 255
```

	iceration	200
	iteration	
	iteration iteration	265 270
	iteration	275
	iteration	280
	iteration	285
	iteration	290
	iteration	295
	iteration	300
	iteration iteration	305 310
	iteration	315
	iteration	320
	iteration	325
	iteration	330
	iteration	335
	iteration iteration	340 345
	iteration	350
	iteration	355
	iteration	360
	iteration	365
	iteration	370
	iteration iteration	
	iteration	
	iteration	
	iteration	395
	iteration	
	iteration	
	iteration	
	iteration iteration	
	iteration	
	iteration	430
	iteration	
	iteration	
	iteration	
	iteration iteration	
	iteration	
	iteration	
	iteration	470
	iteration	475
	iteration	
	iteration iteration	
	iteration	
	iteration	500
	iteration	505
	iteration	510
	iteration	515
	iteration iteration	520 525
	iteration	530
	iteration	535
	iteration	540
	iteration	545
	iteration	550
	iteration iteration	555 560
	iteration	565
	iteration	570
	iteration	575
	iteration	580
	iteration iteration	585 590
	iteration	
	iteration	
	iteration	605
	iteration	
	iteration	
	iteration iteration	
	iteration	
	iteration	
	iteration	640
	iteration	
	iteration	
	iteration	
	iteration iteration	
	iteration	
1	iteration	OBO

····· iteration 695 iteration 700 ····· iteration 705 iteration 710 ····· iteration 715 ····· iteration 720 iteration 725 ····· iteration 730 ····· iteration 735 ····· iteration 740 ····· iteration 745 iteration 750 ····· iteration 755 ····· iteration 760 ····· iteration 765 ····· iteration 770 iteration 775 ····· iteration 780 ····· iteration 785 ····· iteration 790 ····· iteration 795 ····· iteration 800 ····· iteration 805 ····· iteration 810 ····· iteration 815 ····· iteration 820 ····· iteration 825 ····· iteration 830 ····· iteration 835 ····· iteration 840 ····· iteration 845 ····· iteration 850 ····· iteration 855 ····· iteration 860 ····· iteration 865 ····· iteration 870 ····· iteration 875 ····· iteration 880 ····· iteration 885 ····· iteration 890 ····· iteration 895 ····· iteration 900 ····· iteration 905 ····· iteration 910 iteration 915 ····· iteration 920 ····· iteration 925 ····· iteration 930 ····· iteration 935 iteration 940 iteration 945 iteration 950 iteration 955 iteration 960 iteration 965 iteration 970 iteration 975 iteration 980 iteration 985 iteration 990 iteration 995 iteration 1000 final test actual test

In the above plot the test data is plotted for n =0.01 and error is measured. The error rate in this case is 0.00092

7.5

5.0

10.0

(d)The problem is now run for n = 0.0001

-5.0

0.0

2.5

In [117]:

10.0

7.5

5.0

0.0

-2.5

-5.0

-7.5

-10.0

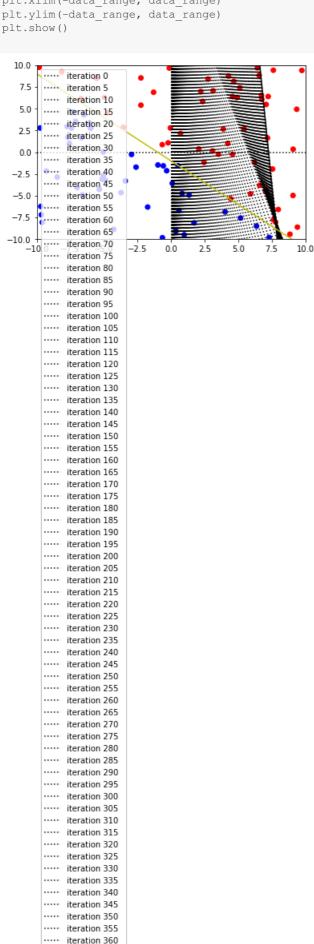
-10.0

```
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
# generate a linearly separated training data of size 100 and test data of size 1000 -
w = np.array([1, 1, 1])
w_{init} = np.array([3, -50, 0])
num sample = 100
```

```
data range = 10
num_sample_test =1000
error = 0
# check if the hypothesis h from the hypothesis set H is following the weight and dimensionality c
ondition
def evaluate h(w, X):
    . . . .
    Evaluate the hypothesis, h \in H, which is described by its set of weights,
    w, against 1 or more points in the input space (stored in a matrix).
    assert len(w.shape) == 1
    assert len(X.shape) == 2
    assert w.shape[0] == X.shape[0]
    return np.sign(w @ X)
X train = np.vstack([
    np.ones(num sample),
    np.random.uniform(
       -data range,
       data range,
       (d, num_sample),
    ),
])
X test = np.vstack([
   np.ones(num_sample_test),
   np.random.uniform(
        -data range,
        data range,
        (d, num sample test),
    ),
])
y train = evaluate h(w act, X train)
y test = evaluate h(w act, X test)
x axis = X train[1, :]
y axis = X train[2, :]
lables = ['r' if y > 0 else 'b' for y in y_train]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Run perceptron learning algorithm for the training data set
def run_perceptron(w_init, X_train, y_train, iteration_callback=None):
    w = w init.copy()
   n = 0
    while True:
        y = \text{evaluate } h(w, X \text{ train})
        if iteration_callback:
           iteration_callback(n, w)
        correct = y == y_train
# Calculate error after 1000 updates
        if n == 1000:
            return w
        else:
# calculate weight considering the closeness between s and y train
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X_train[:,i]
            d = y_train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 0.0001 * (y train[i] - s) * X train[:, i]
                error = (y train[i] - s)
       n = n + 1
def plot hypothesis(x axis, w, *plot args, **plot kwargs):
   m = -w[1]/w[2] if w[2] != 0 else 0
   b = -w[0]/w[2] if w[2] != 0 else 0
    y axis = m*x axis + b
    plt.plot(x_axis, y_axis, *plot_args, **plot_kwargs)
# Plot the hypothesis for all the iterations
def plot_iteration(n, w):
  if n % 5 == 0:
```

```
label1 = 'iteration {}'.format(n)
    plot_hypothesis(x_hypothesis, w, 'k:', label=label1)

w_final = run_perceptron(w_init, X_train, y_train, plot_iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data_range, data_range)
plt.show()
```



 iteration 365
 iteration 370
 iteration 375
 iteration 380 iteration 385
 iteration 390
 iteration 395
 iteration 400
 iteration 405
 iteration 410 iteration 415
 iteration 420
 iteration 425
 iteration 430
 iteration 435 iteration 440
 iteration 445
 iteration 450
 iteration 455
 iteration 460
 iteration 465
 iteration 470 iteration 475
 iteration 480
 iteration 485
 iteration 490
 iteration 495
 iteration 500 iteration 505
 iteration 510
 iteration 515
 iteration 520
 iteration 525
 iteration 530
 iteration 535 iteration 540
 iteration 545
 iteration 550
 iteration 555
 iteration 560
 iteration 565 iteration 570
 iteration 575
 iteration 580
 iteration 585
 iteration 590
 iteration 595 iteration 600
 iteration 605
 iteration 610
 iteration 615
 iteration 620
 iteration 625 iteration 630
 iteration 635
 iteration 640
 iteration 645
 iteration 650
 iteration 655 iteration 660
 iteration 665
 iteration 670
 iteration 675
 iteration 680 iteration 685
 iteration 690
 iteration 695
 iteration 700
 iteration 705
 iteration 710 iteration 715
 iteration 715 iteration 720
 iteration 725
 iteration 730
 iteration 735
 iteration 740
 iteration 745 iteration 750
 iteration 755
 iteration 760
 iteration 765
 iteration 770
 iteration 775 iteration 780
 iteration 785
 iteration 790
 iteration 795

```
····· iteration 800
····· iteration 805
····· iteration 810
····· iteration 815
····· iteration 820
····· iteration 825
····· iteration 830
····· iteration 835
····· iteration 840
····· iteration 845
····· iteration 850
····· iteration 855
····· iteration 860
····· iteration 865
····· iteration 870
····· iteration 875
····· iteration 880
····· iteration 885
····· iteration 890
····· iteration 895
····· iteration 900
····· iteration 905
····· iteration 910
····· iteration 915
····· iteration 920
····· iteration 925
····· iteration 930
····· iteration 935
····· iteration 940
····· iteration 945
····· iteration 950
····· iteration 955
····· iteration 960
····· iteration 965
····· iteration 970
····· iteration 975
····· iteration 980
····· iteration 985
····· iteration 990
····· iteration 995
····· iteration 1000
 — final
     actual
```

In []:

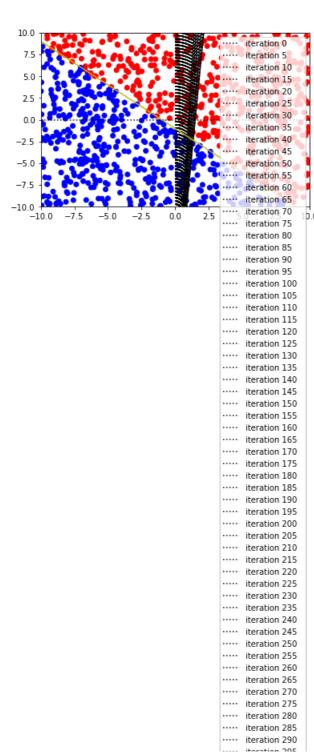
```
The plot above takes into account n =1 for training data
```

In [133]:

```
# Run perceptron learning algorithm for the test data set
def run_perceptron(w_init, X_train, y_train, iteration_callback=None):
   w = w init.copy()
   n = 0
    while True:
       y = \text{evaluate } h(w, X \text{ train})
        if iteration_callback:
           iteration_callback(n, w)
        correct = y == y_train
        if np.all(correct):
#
 Print the total number of iterations that let to the generation of final hypothesis g
            print('Total number of iterations: ',n)
#
             return w
        if n == 1000:
            error = np.absolute(error)
            print("Error on test data for n = 0.0001: ",error[0]/1000)
        else:
            i = np.argmax(~correct) # indice of first misclassified point
            s = w * X_train[:,i]
            d = y train[i] * s
            d = d[0] + d[1] + d[2]
            if d <=1:
                w = w + 0.0001 * (y_train[i] - s) * X_train[:, i]
                error = (y_train[i] - s)
x axis = X test[1. :1
```

```
y_axis = X_test[2, :]
lables = ['r' if y > 0 else 'b' for y in y test]
plt.scatter(x_axis, y_axis, c=lables)
x_hypothesis = np.array([-data_range, data_range])
# Plot the hypothesis for all the iterations
def plot iteration(n, w):
    if n % 5 == 0:
       label1 = 'iteration {}'.format(n)
        plot hypothesis(x hypothesis, w, 'k:', label=label1)
# Run Perceptron learning algorith for test data set
w final = run perceptron(w init, X test, y test, plot iteration)
plot_hypothesis(x_hypothesis, w_final, 'k', label='final test')
plot_hypothesis(x_hypothesis, w_act, 'y', label='actual test')
plt.legend()
plt.xlim(-data_range, data_range)
plt.ylim(-data range, data range)
plt.show()
4
```

Error on test data for n = 0.0001: 0.0036196935435885438



iteration 300 iteration 305 iteration 310 iteration 310 iteration 320 iteration 320 iteration 333 iteration 335 iteration 340 iteration 345 iteration 345 iteration 365 iteration 365 iteration 365 iteration 370 iteration 370 iteration 370 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 391 iteration 400 iteration 401 iteration 402 iteration 411 iteration 412 iteration 420 iteration 420 iteration 420 iteration 420 iteration 435 iteration 440 iteration 445 iteration 445 iteration 455 iteration 460 iteration 460 iteration 475 iteration 480 iteration 470 iteration 480 iteration 490 iteration 490 iteration 505 iteration 500 iteration 600 it		
iteration 305 iteration 310 iteration 320 iteration 320 iteration 330 iteration 335 iteration 340 iteration 345 iteration 355 iteration 355 iteration 365 iteration 370 iteration 370 iteration 375 iteration 380 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 400 iteration 400 iteration 401 iteration 401 iteration 415 iteration 416 iteration 420 iteration 430 iteration 430 iteration 430 iteration 445 iteration 445 iteration 446 iteration 456 iteration 470 iteration 470 iteration 480 iteration 480 iteration 490 iteration 490 iteration 500 iteration 600 it	 петацоп	290
iteration 310 iteration 320 iteration 320 iteration 325 iteration 335 iteration 340 iteration 345 iteration 345 iteration 355 iteration 360 iteration 360 iteration 375 iteration 375 iteration 375 iteration 375 iteration 380 iteration 380 iteration 380 iteration 380 iteration 380 iteration 390 iteration 400 iteration 400 iteration 400 iteration 410 iteration 415 iteration 420 iteration 425 iteration 426 iteration 433 iteration 440 iteration 445 iteration 446 iteration 446 iteration 455 iteration 460 iteration 470 iteration 470 iteration 485 iteration 480 iteration 480 iteration 490 iteration 500 iteration 600 it	 iteration	300
iteration 315 iteration 320 iteration 325 iteration 330 iteration 330 iteration 340 iteration 345 iteration 350 iteration 350 iteration 360 iteration 365 iteration 365 iteration 370 iteration 380 iteration 380 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 400 iteration 401 iteration 415 iteration 415 iteration 425 iteration 426 iteration 430 iteration 433 iteration 440 iteration 440 iteration 440 iteration 440 iteration 440 iteration 455 iteration 450 iteration 450 iteration 460 iteration 470 iteration 470 iteration 480 iteration 480 iteration 490 iteration 500 it	 iteration	305
iteration 320 iteration 325 iteration 335 iteration 335 iteration 345 iteration 350 iteration 355 iteration 360 iteration 365 iteration 370 iteration 370 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 390 iteration 400 iteration 401 iteration 410 iteration 410 iteration 420 iteration 420 iteration 420 iteration 430 iteration 431 iteration 433 iteration 440 iteration 445 iteration 455 iteration 455 iteration 456 iteration 466 iteration 477 iteration 480 iteration 480 iteration 480 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 600 it	 iteration	
iteration 325 iteration 330 iteration 340 iteration 340 iteration 350 iteration 350 iteration 360 iteration 365 iteration 365 iteration 370 iteration 370 iteration 385 iteration 380 iteration 380 iteration 390 iteration 390 iteration 390 iteration 400 iteration 400 iteration 401 iteration 410 iteration 411 iteration 412 iteration 420 iteration 420 iteration 430 iteration 431 iteration 433 iteration 440 iteration 445 iteration 446 iteration 450 iteration 470 iteration 480 iteration 480 iteration 480 iteration 480 iteration 480 iteration 495 iteration 500 iteration 500 iteration 500 iteration 500 iteration 520 iteration 520 iteration 520 iteration 530 iteration 535 iteration 540 iteration 545 iteration 565 iteration 565 iteration 560 iteration 570 iteration 565 iteration 570 iteration 565 iteration 600 it	 	
iteration 330 iteration 345 iteration 345 iteration 355 iteration 360 iteration 365 iteration 365 iteration 370 iteration 370 iteration 375 iteration 380 iteration 380 iteration 380 iteration 390 iteration 400 iteration 400 iteration 401 iteration 405 iteration 410 iteration 411 iteration 425 iteration 426 iteration 433 iteration 433 iteration 440 iteration 445 iteration 446 iteration 450 iteration 450 iteration 450 iteration 460 iteration 470 iteration 470 iteration 470 iteration 480 iteration 480 iteration 485 iteration 480 iteration 490 iteration 490 iteration 500 iteration 600 it		
iteration 335 iteration 340 iteration 350 iteration 350 iteration 360 iteration 360 iteration 370 iteration 375 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 400 iteration 400 iteration 410 iteration 411 iteration 412 iteration 415 iteration 420 iteration 430 iteration 430 iteration 430 iteration 430 iteration 440 iteration 445 iteration 450 iteration 460 iteration 475 iteration 470 iteration 475 iteration 470 iteration 480 iteration 470 iteration 480 iteration 480 iteration 500 iteration 520 iteration 520 iteration 520 iteration 530 iteration 540 iteration 540 iteration 555 iteration 560 iteration 660 iteration 660 iteration 600 it		
iteration 340 iteration 355 iteration 355 iteration 365 iteration 370 iteration 375 iteration 375 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 400 iteration 400 iteration 415 iteration 415 iteration 425 iteration 426 iteration 430 iteration 430 iteration 430 iteration 445 iteration 445 iteration 446 iteration 455 iteration 460 iteration 470 iteration 470 iteration 480 iteration 480 iteration 470 iteration 470 iteration 480 iteration 500 iteration 500 iteration 500 iteration 501 iteration 501 iteration 520 iteration 620 iteration 620 iteration 620 iteration 620 iteration 630 iteration 640 iteration 640 iteration 665 iteration 667 iteration 668 iteration 670 it		
iteration 345 iteration 350 iteration 365 iteration 366 iteration 375 iteration 375 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 490 iteration 405 iteration 415 iteration 415 iteration 420 iteration 425 iteration 430 iteration 430 iteration 430 iteration 445 iteration 445 iteration 455 iteration 460 iteration 470 iteration 480 iteration 480 iteration 480 iteration 470 iteration 480 iteration 490 iteration 500 iteration 500 iteration 500 iteration 500 iteration 500 iteration 500 iteration 520 it		
iteration 350 iteration 365 iteration 365 iteration 375 iteration 375 iteration 380 iteration 380 iteration 385 iteration 380 iteration 385 iteration 390 iteration 395 iteration 400 iteration 401 iteration 415 iteration 415 iteration 425 iteration 425 iteration 436 iteration 436 iteration 440 iteration 440 iteration 446 iteration 455 iteration 460 iteration 465 iteration 477 iteration 485 iteration 480 iteration 480 iteration 490 iteration 490 iteration 500 iteration 500 iteration 500 iteration 501 iteration 500 iteration 501 iteration 503 iteration 525 iteration 520 iteration 525 iteration 520 iteration 520 iteration 520 iteration 525 iteration 520 iteration 630 iteration 640 it		
iteration 355 iteration 360 iteration 370 iteration 370 iteration 380 iteration 380 iteration 380 iteration 390 iteration 390 iteration 395 iteration 400 iteration 405 iteration 410 iteration 415 iteration 420 iteration 420 iteration 425 iteration 430 iteration 435 iteration 440 iteration 445 iteration 455 iteration 455 iteration 460 iteration 470 iteration 480 iteration 500 iteration 500 iteration 500 iteration 501 iteration 500 iteration 600 it		
iteration 360 iteration 370 iteration 370 iteration 370 iteration 385 iteration 385 iteration 395 iteration 395 iteration 400 iteration 405 iteration 410 iteration 410 iteration 420 iteration 425 iteration 433 iteration 435 iteration 445 iteration 445 iteration 455 iteration 455 iteration 460 iteration 477 iteration 480 iteration 495 iteration 500 iteration 600 it		
iteration 365 iteration 370 iteration 375 iteration 385 iteration 390 iteration 395 iteration 400 iteration 405 iteration 410 iteration 411 iteration 425 iteration 425 iteration 435 iteration 436 iteration 446 iteration 446 iteration 455 iteration 456 iteration 456 iteration 465 iteration 470 iteration 480 iteration 481 iteration 482 iteration 483 iteration 465 iteration 470 iteration 470 iteration 481 iteration 482 iteration 483 iteration 505 iteration 500 iteration 500 iteration 500 iteration 501 iteration 501 iteration 502 iteration 503 iteration 520 iteration 520 iteration 520 iteration 520 iteration 520 iteration 520 iteration 535 iteration 540 iteration 555 iteration 555 iteration 565 iteration 565 iteration 560 iteration 570 iteration 565 iteration 570 iteration 560 iteration 605 iteration 607 iteration 608 iteration 609 iteration 609 iteration 609 it	 	
iteration 375 iteration 380 iteration 390 iteration 390 iteration 400 iteration 410 iteration 415 iteration 425 iteration 425 iteration 435 iteration 435 iteration 445 iteration 445 iteration 445 iteration 455 iteration 466 iteration 477 iteration 478 iteration 477 iteration 480 iteration 478 iteration 481 iteration 479 iteration 485 iteration 486 iteration 500 iteration 500 iteration 500 iteration 501 iteration 501 iteration 502 iteration 503 iteration 503 iteration 504 iteration 540 iteration 540 iteration 555 iteration 555 iteration 555 iteration 556 iteration 557 iteration 558 iteration 560 iteration 570 iteration 580 iteration 590 iteration 590 iteration 590 iteration 555 iteration 560 iteration 555 iteration 560 iteration 560 iteration 560 iteration 560 iteration 560 iteration 560 iteration 665 iteration 600 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 700		365
iteration 380 iteration 390 iteration 390 iteration 390 iteration 490 iteration 405 iteration 410 iteration 420 iteration 425 iteration 430 iteration 430 iteration 434 iteration 445 iteration 455 iteration 460 iteration 465 iteration 470 iteration 480 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 500 iteration 500 iteration 500 iteration 510 iteration 520 iteration 620 iteration 630 iteration 630 iteration 630 iteration 630 iteration 640 iteration 645 iteration 645 iteration 665 iteration 665 iteration 667 iteration 668 iteration 669 iteration 670 iteration 700 it	 iteration	370
iteration 385 iteration 390 iteration 395 iteration 400 iteration 410 iteration 412 iteration 422 iteration 423 iteration 433 iteration 443 iteration 444 iteration 455 iteration 456 iteration 460 iteration 470 iteration 470 iteration 485 iteration 480 iteration 480 iteration 470 iteration 470 iteration 481 iteration 482 iteration 500 iteration 500 iteration 501 iteration 501 iteration 502 iteration 510 iteration 520 iteration 530 iteration 520 iteration 530 iteration 520 iteration 530 iteration 540 iteration 555 iteration 560 iteration 565 iteration 565 iteration 560 iteration 570 iteration 580 iteration 580 iteration 580 iteration 580 iteration 595 iteration 600 iteration 605 iteration 605 iteration 605 iteration 606 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 609 iteration 675 iteration 609 iteration 675 iteration 670 iteration 705	 iteration	375
iteration 390 iteration 400 iteration 405 iteration 410 iteration 415 iteration 425 iteration 425 iteration 425 iteration 436 iteration 440 iteration 440 iteration 445 iteration 455 iteration 455 iteration 465 iteration 477 iteration 480 iteration 480 iteration 480 iteration 480 iteration 490 iteration 505 iteration 505 iteration 500 iteration 505 iteration 525 iteration 525 iteration 526 iteration 526 iteration 530 iteration 530 iteration 540 iteration 550 iteration 560 iteration 600 iteration 600 iteration 601 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 609 iteration 609 iteration 609 iteration 700	 iteration	380
iteration 395 iteration 400 iteration 405 iteration 410 iteration 420 iteration 420 iteration 423 iteration 433 iteration 444 iteration 445 iteration 455 iteration 455 iteration 460 iteration 477 iteration 480 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 500 iteration 501 iteration 502 iteration 503 iteration 520 iteration 520 iteration 530 iteration 550 iteration 550 iteration 550 iteration 550 iteration 560 iteration 560 iteration 560 iteration 560 iteration 570 iteration 580 iteration 590 iteration 600 iteration 601 iteration 602 iteration 603 iteration 603 iteration 604 iteration 635 iteration 635 iteration 635 iteration 640 iteration 645 iteration 645 iteration 655 iteration 660 iteration 665 iteration 660 iteration 675 iteration 685 iteration 690 iteration 700	 iteration	385
iteration 400 iteration 405 iteration 415 iteration 425 iteration 425 iteration 436 iteration 437 iteration 445 iteration 445 iteration 450 iteration 450 iteration 465 iteration 477 iteration 477 iteration 478 iteration 479 iteration 488 iteration 489 iteration 499 iteration 500 iteration 500 iteration 501 iteration 501 iteration 502 iteration 503 iteration 520 iteration 520 iteration 535 iteration 535 iteration 536 iteration 540 iteration 545 iteration 555 iteration 565 iteration 570 iteration 565 iteration 570 iteration 575 iteration 570 iteration 570 iteration 570 iteration 570 iteration 570 iteration 600 iteration 600 iteration 600 iteration 601 iteration 602 iteration 603 iteration 6040 iteration 605 iteration 645 iteration 645 iteration 645 iteration 645 iteration 665 iteration 667 iteration 675 iteration 705	 iteration	390
iteration 405 iteration 410 iteration 420 iteration 420 iteration 430 iteration 433 iteration 445 iteration 445 iteration 445 iteration 455 iteration 460 iteration 475 iteration 476 iteration 477 iteration 485 iteration 488 iteration 488 iteration 490 iteration 500 iteration 500 iteration 500 iteration 501 iteration 515 iteration 520 iteration 520 iteration 530 iteration 533 iteration 533 iteration 540 iteration 540 iteration 555 iteration 555 iteration 556 iteration 570 iteration 580 iteration 580 iteration 580 iteration 570 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 660 iteration 600 iteration 600 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 609 iteration 609 iteration 609 iteration 609 iteration 705	 iteration	395
iteration 410 iteration 420 iteration 425 iteration 435 iteration 445 iteration 445 iteration 445 iteration 445 iteration 455 iteration 460 iteration 475 iteration 475 iteration 475 iteration 475 iteration 480 iteration 485 iteration 486 iteration 485 iteration 485 iteration 500 iteration 500 iteration 500 iteration 501 iteration 501 iteration 515 iteration 520 iteration 520 iteration 525 iteration 530 iteration 533 iteration 540 iteration 540 iteration 555 iteration 555 iteration 555 iteration 556 iteration 570 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 595 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 609 iteration 609 iteration 609 iteration 609 iteration 700 iteration 700 iteration 700 iteration 710 iteration 720		
iteration 415 iteration 420 iteration 430 iteration 430 iteration 430 iteration 440 iteration 445 iteration 450 iteration 455 iteration 460 iteration 470 iteration 470 iteration 470 iteration 480 iteration 480 iteration 480 iteration 480 iteration 480 iteration 500 iteration 500 iteration 500 iteration 501 iteration 501 iteration 515 iteration 520 iteration 520 iteration 520 iteration 520 iteration 520 iteration 530 iteration 520 iteration 530 iteration 540 iteration 540 iteration 555 iteration 560 iteration 555 iteration 560 iteration 560 iteration 570 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 590 iteration 605 iteration 605 iteration 605 iteration 600 iteration 605 iteration 606 iteration 607 iteration 608 iteration 609 iteration 609 iteration 609 iteration 609 iteration 705		
iteration 420 iteration 430 iteration 430 iteration 430 iteration 440 iteration 445 iteration 455 iteration 455 iteration 465 iteration 465 iteration 470 iteration 470 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 500 iteration 501 iteration 510 iteration 510 iteration 520 iteration 520 iteration 520 iteration 530 iteration 520 iteration 533 iteration 540 iteration 540 iteration 555 iteration 555 iteration 560 iteration 570 iteration 560 iteration 570 iteration 580 iteration 595 iteration 600 iteration 605 iteration 605 iteration 605 iteration 605 iteration 605 iteration 605 iteration 620 iteration 635 iteration 640 iteration 645 iteration 645 iteration 655 iteration 665 iteration 665 iteration 665 iteration 665 iteration 665 iteration 665 iteration 670 iteration 675 iteration 670 iteration 675 iteration 670 iteration 670 iteration 670 iteration 705		
iteration 425 iteration 430 iteration 435 iteration 440 iteration 445 iteration 455 iteration 465 iteration 465 iteration 477 iteration 477 iteration 485 iteration 480 iteration 485 iteration 490 iteration 500 iteration 500 iteration 501 iteration 501 iteration 501 iteration 502 iteration 525 iteration 525 iteration 526 iteration 530 iteration 530 iteration 530 iteration 535 iteration 540 iteration 550 iteration 550 iteration 550 iteration 550 iteration 550 iteration 550 iteration 565 iteration 570 iteration 570 iteration 575 iteration 580 iteration 585 iteration 580 iteration 595 iteration 595 iteration 600 iteration 601 iteration 611 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 636 iteration 635 iteration 640 iteration 645 iteration 655 iteration 660 iteration 675 iteration 660 iteration 675 iteration 670 iteration 670 iteration 670 iteration 670 iteration 700 iteration 700 iteration 710 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720		
iteration 430 iteration 445 iteration 445 iteration 455 iteration 465 iteration 466 iteration 470 iteration 477 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 500 iteration 501 iteration 505 iteration 515 iteration 525 iteration 526 iteration 530 iteration 530 iteration 530 iteration 530 iteration 540 iteration 540 iteration 550 iteration 565 iteration 565 iteration 570 iteration 570 iteration 570 iteration 580 iteration 570 iteration 580 iteration 590 iteration 590 iteration 600 iteration 601 iteration 615 iteration 615 iteration 620 iteration 635 iteration 635 iteration 635 iteration 635 iteration 640 iteration 645 iteration 645 iteration 665 iteration 660 iteration 675 iteration 685 iteration 690 iteration 690 iteration 700		
iteration 435 iteration 440 iteration 445 iteration 455 iteration 465 iteration 465 iteration 470 iteration 470 iteration 488 iteration 488 iteration 495 iteration 495 iteration 500 iteration 500 iteration 501 iteration 501 iteration 502 iteration 515 iteration 520 iteration 520 iteration 520 iteration 535 iteration 535 iteration 535 iteration 540 iteration 545 iteration 555 iteration 555 iteration 565 iteration 565 iteration 570 iteration 570 iteration 570 iteration 575 iteration 580 iteration 580 iteration 590 iteration 590 iteration 605 iteration 600 iteration 600 iteration 601 iteration 605 iteration 635 iteration 630 iteration 635 iteration 635 iteration 640 iteration 645 iteration 645 iteration 665 iteration 667 iteration 669 iteration 675 iteration 675 iteration 680 iteration 690 iteration 700 iteration 700 iteration 700 iteration 700 iteration 700 iteration 700 iteration 720		
iteration 440 iteration 445 iteration 450 iteration 450 iteration 460 iteration 470 iteration 477 iteration 480 iteration 485 iteration 485 iteration 490 iteration 500 iteration 500 iteration 510 iteration 511 iteration 520 iteration 520 iteration 520 iteration 520 iteration 535 iteration 535 iteration 536 iteration 536 iteration 540 iteration 555 iteration 555 iteration 555 iteration 565 iteration 570 iteration 570 iteration 580 iteration 585 iteration 580 iteration 590 iteration 590 iteration 600 it		
iteration 445 iteration 455 iteration 460 iteration 460 iteration 477 iteration 477 iteration 485 iteration 485 iteration 485 iteration 485 iteration 490 iteration 495 iteration 500 iteration 500 iteration 515 iteration 520 iteration 520 iteration 525 iteration 525 iteration 530 iteration 533 iteration 540 iteration 540 iteration 540 iteration 555 iteration 555 iteration 560 iteration 565 iteration 570 iteration 577 iteration 580 iteration 660 iteration 600 iteration 600 iteration 600 iteration 625 iteration 625 iteration 625 iteration 625 iteration 625 iteration 625 iteration 630 iteration 640 iteration 645 iteration 655 iteration 665 iteration 669 iteration 670 iteration 690 iteration 700 iteration 700 iteration 700 iteration 700 iteration 700 iteration 700 iteration 720 iteration 720 iteration 720 iteration 720		
iteration 450 iteration 455 iteration 460 iteration 467 iteration 475 iteration 475 iteration 480 iteration 485 iteration 480 iteration 485 iteration 490 iteration 500 iteration 501 iteration 515 iteration 520 iteration 520 iteration 523 iteration 523 iteration 530 iteration 533 iteration 540 iteration 540 iteration 555 iteration 555 iteration 555 iteration 560 iteration 570 iteration 570 iteration 570 iteration 580 iteration 580 iteration 580 iteration 590 iteration 605 iteration 605 iteration 605 iteration 605 iteration 620 iteration 620 iteration 625 iteration 620 iteration 635 iteration 620 iteration 635 iteration 620 iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 701 iteration 702 iteration 705 iteration 705 iteration 706 iteration 707 iteration 705 iteration 707		
iteration 455 iteration 460 iteration 470 iteration 470 iteration 470 iteration 480 iteration 480 iteration 485 iteration 490 iteration 500 iteration 500 iteration 510 iteration 520 iteration 520 iteration 520 iteration 530 iteration 533 iteration 540 iteration 545 iteration 555 iteration 555 iteration 560 iteration 560 iteration 570 iteration 570 iteration 580 iteration 595 iteration 605 iteration 605 iteration 605 iteration 605 iteration 605 iteration 605 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 670 iteration 675 iteration 670 iteration 690 iteration 690 iteration 705		
iteration 460 iteration 470 iteration 475 iteration 475 iteration 485 iteration 485 iteration 490 iteration 495 iteration 500 iteration 501 iteration 510 iteration 520 iteration 520 iteration 525 iteration 530 iteration 533 iteration 535 iteration 540 iteration 545 iteration 555 iteration 555 iteration 560 iteration 570 iteration 570 iteration 570 iteration 580 iteration 588 iteration 588 iteration 595 iteration 595 iteration 600 iteration 601 iteration 602 iteration 603 iteration 620 iteration 635 iteration 635 iteration 635 iteration 630 iteration 635 iteration 630 iteration 645 iteration 645 iteration 655 iteration 660 iteration 675 iteration 675 iteration 670 iteration 705		
iteration 470 iteration 480 iteration 480 iteration 480 iteration 490 iteration 500 iteration 500 iteration 501 iteration 510 iteration 515 iteration 520 iteration 530 iteration 530 iteration 535 iteration 540 iteration 545 iteration 556 iteration 556 iteration 565 iteration 570 iteration 570 iteration 570 iteration 575 iteration 580 iteration 570 iteration 580 iteration 580 iteration 580 iteration 590 iteration 590 iteration 600 iteration 601 iteration 601 iteration 615 iteration 620 iteration 635 iteration 635 iteration 635 iteration 635 iteration 635 iteration 635 iteration 640 iteration 645 iteration 665 iteration 660 iteration 665 iteration 660 iteration 660 iteration 660 iteration 660 iteration 675 iteration 670 iteration 680 iteration 690 iteration 700 iteration 700 iteration 710 iteration 720		
iteration 475 iteration 480 iteration 485 iteration 495 iteration 500 iteration 501 iteration 510 iteration 520 iteration 520 iteration 520 iteration 535 iteration 535 iteration 540 iteration 545 iteration 555 iteration 555 iteration 565 iteration 570 iteration 575 iteration 600 iteration 600 iteration 600 iteration 601 iteration 615 iteration 620 iteration 630 iteration 630 iteration 630 iteration 630 iteration 640 iteration 645 iteration 655 iteration 660 iteration 665 iteration 660 iteration 665 iteration 660 iteration 665 iteration 660 iteration 675 iteration 670 iteration 680 iteration 690 iteration 700 iteration 700 iteration 710 iteration 715 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720	 iteration	465
iteration 480 iteration 490 iteration 495 iteration 595 iteration 510 iteration 515 iteration 516 iteration 520 iteration 525 iteration 530 iteration 533 iteration 545 iteration 545 iteration 556 iteration 556 iteration 557 iteration 565 iteration 570 iteration 570 iteration 570 iteration 585 iteration 585 iteration 580 iteration 580 iteration 610 iteration 600 iteration 600 iteration 610 iteration 611 iteration 625 iteration 625 iteration 625 iteration 625 iteration 625 iteration 630 iteration 630 iteration 640 iteration 650 iteration 650 iteration 650 iteration 650 iteration 665 iteration 665 iteration 665 iteration 660 iteration 670 iteration 670 iteration 670 iteration 700 iteration 710 iteration 710 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720	 iteration	470
iteration 485 iteration 490 iteration 500 iteration 500 iteration 500 iteration 515 iteration 525 iteration 525 iteration 530 iteration 535 iteration 540 iteration 540 iteration 555 iteration 555 iteration 560 iteration 565 iteration 570 iteration 577 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 590 iteration 600 iteration 600 iteration 605 iteration 605 iteration 605 iteration 625 iteration 625 iteration 625 iteration 625 iteration 630 iteration 640 iteration 655 iteration 655 iteration 665 iteration 670 iteration 680 iteration 690 iteration 705	 iteration	475
iteration 490 iteration 495 iteration 500 iteration 505 iteration 515 iteration 520 iteration 523 iteration 530 iteration 535 iteration 540 iteration 545 iteration 555 iteration 556 iteration 560 iteration 570 iteration 577 iteration 580 iteration 580 iteration 580 iteration 580 iteration 595 iteration 590 iteration 605 iteration 605 iteration 605 iteration 605 iteration 605 iteration 620 iteration 620 iteration 620 iteration 620 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 640 iteration 655 iteration 665 iteration 669 iteration 670 iteration 690 iteration 700 iteration 705 iteration 707 iteration 705 iteration 705 iteration 707 iteration 705	 iteration	480
iteration 495 iteration 500 iteration 510 iteration 510 iteration 520 iteration 520 iteration 523 iteration 533 iteration 535 iteration 540 iteration 545 iteration 555 iteration 560 iteration 565 iteration 570 iteration 577 iteration 588 iteration 588 iteration 588 iteration 599 iteration 599 iteration 600 iteration 601 iteration 611 iteration 620 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 635 iteration 635 iteration 640 iteration 655 iteration 660 iteration 665 iteration 660 iteration 675 iteration 675 iteration 670 iteration 680 iteration 670 iteration 690 iteration 700 iteration 705 iteration 705 iteration 705 iteration 705 iteration 710 iteration 720 iteration 725	 iteration	485
iteration 500 iteration 510 iteration 515 iteration 525 iteration 530 iteration 533 iteration 535 iteration 535 iteration 540 iteration 545 iteration 550 iteration 550 iteration 560 iteration 570 iteration 577 iteration 578 iteration 578 iteration 578 iteration 588 iteration 589 iteration 599 iteration 600 iteration 601 iteration 611 iteration 612 iteration 620 iteration 635 iteration 630 iteration 635 iteration 635 iteration 635 iteration 640 iteration 655 iteration 660 iteration 665 iteration 660 iteration 675 iteration 670 iteration 670 iteration 685 iteration 690 iteration 700 iteration 705 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 720 iteration 720 iteration 720 iteration 725		
iteration 505 iteration 510 iteration 520 iteration 520 iteration 530 iteration 535 iteration 540 iteration 545 iteration 545 iteration 555 iteration 555 iteration 565 iteration 560 iteration 570 iteration 577 iteration 578 iteration 579 iteration 580 iteration 579 iteration 580 iteration 590 iteration 600 iteration 601 iteration 601 iteration 615 iteration 615 iteration 620 iteration 630 iteration 630 iteration 635 iteration 635 iteration 635 iteration 645 iteration 655 iteration 665 iteration 660 iteration 660 iteration 660 iteration 660 iteration 675 iteration 675 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 700 iteration 715 iteration 715 iteration 720 iteration 725		
iteration 510 iteration 520 iteration 525 iteration 535 iteration 536 iteration 540 iteration 545 iteration 545 iteration 550 iteration 550 iteration 555 iteration 565 iteration 575 iteration 577 iteration 580 iteration 580 iteration 580 iteration 580 iteration 690 iteration 600 iteration 610 iteration 610 iteration 611 iteration 620 iteration 625 iteration 620 iteration 630 iteration 630 iteration 640 iteration 645 iteration 650 iteration 655 iteration 650 iteration 650 iteration 650 iteration 665 iteration 665 iteration 660 iteration 665 iteration 660 iteration 665 iteration 660 iteration 670 iteration 680 iteration 690 iteration 700 iteration 700 iteration 700 iteration 710 iteration 710 iteration 710 iteration 720 iteration 720 iteration 720	 	
iteration 515 iteration 520 iteration 530 iteration 530 iteration 530 iteration 540 iteration 545 iteration 555 iteration 556 iteration 565 iteration 570 iteration 570 iteration 570 iteration 580 iteration 580 iteration 580 iteration 690 iteration 600 iteration 601 iteration 610 iteration 610 iteration 625 iteration 625 iteration 625 iteration 630 iteration 630 iteration 640 iteration 645 iteration 650 iteration 650 iteration 650 iteration 650 iteration 650 iteration 665 iteration 665 iteration 665 iteration 660 iteration 660 iteration 665 iteration 660 iteration 660 iteration 660 iteration 660 iteration 660 iteration 670 iteration 680 iteration 690 iteration 700 iteration 700 iteration 700 iteration 701 iteration 710 iteration 710 iteration 720 iteration 720 iteration 720 iteration 720		
iteration 520 iteration 530 iteration 535 iteration 545 iteration 555 iteration 556 iteration 560 iteration 565 iteration 570 iteration 577 iteration 585 iteration 580 iteration 580 iteration 580 iteration 580 iteration 600 iteration 600 iteration 601 iteration 601 iteration 611 iteration 612 iteration 625 iteration 625 iteration 630 iteration 630 iteration 635 iteration 640 iteration 645 iteration 655 iteration 655 iteration 665 iteration 665 iteration 665 iteration 665 iteration 665 iteration 665 iteration 670 iteration 680 iteration 690 iteration 700 iteration 700 iteration 700 iteration 701 iteration 710 iteration 710 iteration 710 iteration 720 iteration 720 iteration 720 iteration 720 iteration 720	 	
iteration 525 iteration 530 iteration 540 iteration 540 iteration 555 iteration 555 iteration 565 iteration 565 iteration 570 iteration 575 iteration 576 iteration 580 iteration 580 iteration 580 iteration 590 iteration 600 iteration 605 iteration 601 iteration 605 iteration 601 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 640 iteration 645 iteration 655 iteration 655 iteration 665 iteration 670 iteration 680 iteration 690 iteration 700 iteration 705 iteration 710 iteration 720 iteration 725		
iteration 530 iteration 540 iteration 545 iteration 555 iteration 560 iteration 565 iteration 565 iteration 570 iteration 570 iteration 575 iteration 588 iteration 588 iteration 598 iteration 599 iteration 600 iteration 601 iteration 611 iteration 612 iteration 620 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 636 iteration 645 iteration 655 iteration 665 iteration 665 iteration 665 iteration 665 iteration 670 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 705 iteration 710 iteration 720 iteration 720		
iteration 535 iteration 540 iteration 550 iteration 550 iteration 565 iteration 565 iteration 577 iteration 577 iteration 578 iteration 578 iteration 588 iteration 589 iteration 599 iteration 600 iteration 601 iteration 611 iteration 615 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 645 iteration 655 iteration 655 iteration 660 iteration 665 iteration 660 iteration 660 iteration 675 iteration 675 iteration 670 iteration 670 iteration 680 iteration 670 iteration 690 iteration 700 iteration 705 iteration 710 iteration 720 iteration 725		
iteration 545 iteration 550 iteration 550 iteration 560 iteration 567 iteration 575 iteration 575 iteration 580 iteration 580 iteration 585 iteration 585 iteration 580 iteration 590 iteration 600 iteration 605 iteration 610 iteration 611 iteration 620 iteration 625 iteration 630 iteration 635 iteration 640 iteration 645 iteration 650 iteration 655 iteration 665 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 700 iteration 705 iteration 705 iteration 705 iteration 705 iteration 715 iteration 720 iteration 720 iteration 720		
iteration 550 iteration 555 iteration 566 iteration 567 iteration 577 iteration 580 iteration 580 iteration 580 iteration 580 iteration 580 iteration 590 iteration 690 iteration 600 iteration 615 iteration 620 iteration 625 iteration 630 iteration 635 iteration 636 iteration 640 iteration 650 iteration 650 iteration 655 iteration 665 iteration 670 iteration 680 iteration 680 iteration 690 iteration 690 iteration 700 iteration 700 iteration 700 iteration 701 iteration 715 iteration 710 iteration 720 iteration 720 iteration 720 iteration 720	 iteration	540
iteration 555 iteration 560 iteration 565 iteration 575 iteration 585 iteration 585 iteration 585 iteration 585 iteration 590 iteration 595 iteration 600 iteration 605 iteration 610 iteration 611 iteration 620 iteration 625 iteration 633 iteration 635 iteration 635 iteration 645 iteration 655 iteration 655 iteration 665 iteration 665 iteration 665 iteration 665 iteration 665 iteration 667 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 705 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725	 iteration	545
iteration 560 iteration 570 iteration 575 iteration 575 iteration 585 iteration 590 iteration 590 iteration 600 iteration 600 iteration 610 iteration 615 iteration 625 iteration 625 iteration 630 iteration 633 iteration 635 iteration 640 iteration 650 iteration 655 iteration 650 iteration 650 iteration 665 iteration 670 iteration 675 iteration 680 iteration 690 iteration 690 iteration 705 iteration 700 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 720 iteration 720 iteration 720	 iteration	550
iteration 565 iteration 570 iteration 575 iteration 580 iteration 580 iteration 595 iteration 695 iteration 605 iteration 615 iteration 620 iteration 620 iteration 635 iteration 635 iteration 635 iteration 645 iteration 645 iteration 655 iteration 655 iteration 665 iteration 665 iteration 665 iteration 665 iteration 675 iteration 670 iteration 680 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 725	 iteration	555
iteration 570 iteration 575 iteration 585 iteration 595 iteration 605 iteration 605 iteration 615 iteration 620 iteration 620 iteration 635 iteration 635 iteration 645 iteration 645 iteration 655 iteration 655 iteration 660 iteration 660 iteration 675 iteration 675 iteration 670 iteration 685 iteration 670 iteration 685 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 725	 iteration	
iteration 575 iteration 580 iteration 580 iteration 595 iteration 605 iteration 605 iteration 610 iteration 620 iteration 620 iteration 635 iteration 635 iteration 645 iteration 645 iteration 655 iteration 665 iteration 665 iteration 660 iteration 675 iteration 675 iteration 670 iteration 685 iteration 685 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 725		
iteration 580 iteration 595 iteration 595 iteration 600 iteration 610 iteration 615 iteration 625 iteration 635 iteration 635 iteration 645 iteration 645 iteration 645 iteration 655 iteration 655 iteration 660 iteration 660 iteration 675 iteration 675 iteration 675 iteration 670 iteration 685 iteration 690 iteration 705 iteration 705 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 725		
iteration 585 iteration 590 iteration 595 iteration 600 iteration 610 iteration 610 iteration 620 iteration 625 iteration 630 iteration 635 iteration 645 iteration 650 iteration 655 iteration 665 iteration 665 iteration 665 iteration 667 iteration 675 iteration 680 iteration 688 iteration 690 iteration 690 iteration 690 iteration 700 iteration 700 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 720 iteration 725		
iteration 590 iteration 595 iteration 600 iteration 610 iteration 610 iteration 620 iteration 625 iteration 635 iteration 635 iteration 640 iteration 645 iteration 650 iteration 665 iteration 660 iteration 667 iteration 670 iteration 680 iteration 690 iteration 690 iteration 700 iteration 700 iteration 705 iteration 705 iteration 715 iteration 715 iteration 715 iteration 725		
iteration 595 iteration 600 iteration 605 iteration 610 iteration 615 iteration 625 iteration 630 iteration 635 iteration 635 iteration 640 iteration 645 iteration 650 iteration 650 iteration 665 iteration 665 iteration 670 iteration 680 iteration 685 iteration 685 iteration 670 iteration 690 iteration 690 iteration 705 iteration 700 iteration 705 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 600 iteration 610 iteration 615 iteration 625 iteration 630 iteration 630 iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 665 iteration 667 iteration 675 iteration 675 iteration 680 iteration 680 iteration 690 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 605 iteration 615 iteration 620 iteration 620 iteration 635 iteration 635 iteration 645 iteration 645 iteration 655 iteration 655 iteration 660 iteration 675 iteration 675 iteration 675 iteration 675 iteration 675 iteration 685 iteration 685 iteration 685 iteration 685 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 610 iteration 620 iteration 625 iteration 635 iteration 635 iteration 640 iteration 645 iteration 645 iteration 655 iteration 655 iteration 660 iteration 675 iteration 677 iteration 685 iteration 685 iteration 685 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 715 iteration 725		
iteration 615 iteration 620 iteration 625 iteration 635 iteration 640 iteration 645 iteration 645 iteration 655 iteration 655 iteration 660 iteration 660 iteration 675 iteration 675 iteration 685 iteration 685 iteration 685 iteration 685 iteration 690 iteration 690 iteration 705 iteration 705 iteration 705 iteration 715 iteration 715 iteration 725		
iteration 625 iteration 630 iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 665 iteration 670 iteration 675 iteration 685 iteration 685 iteration 685 iteration 685 iteration 690 iteration 690 iteration 700 iteration 700 iteration 705 iteration 715 iteration 715 iteration 725		
iteration 630 iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 665 iteration 667 iteration 670 iteration 688 iteration 688 iteration 690 iteration 690 iteration 700 iteration 700 iteration 705 iteration 715 iteration 715 iteration 715 iteration 720 iteration 720	 iteration	620
iteration 635 iteration 640 iteration 645 iteration 655 iteration 665 iteration 665 iteration 667 iteration 670 iteration 685 iteration 688 iteration 689 iteration 690 iteration 700 iteration 700 iteration 705 iteration 715 iteration 715 iteration 720 iteration 725		
iteration 640 iteration 655 iteration 655 iteration 665 iteration 665 iteration 667 iteration 675 iteration 675 iteration 680 iteration 685 iteration 685 iteration 690 iteration 700 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 645 iteration 650 iteration 655 iteration 660 iteration 670 iteration 675 iteration 675 iteration 680 iteration 685 iteration 685 iteration 690 iteration 705 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 650 iteration 665 iteration 665 iteration 665 iteration 670 iteration 675 iteration 680 iteration 680 iteration 685 iteration 695 iteration 700 iteration 705 iteration 715 iteration 715 iteration 720 iteration 725		
iteration 655 iteration 660 iteration 665 iteration 670 iteration 675 iteration 680 iteration 680 iteration 690 iteration 700 iteration 705 iteration 715 iteration 725		
iteration 660 iteration 665 iteration 675 iteration 680 iteration 685 iteration 685 iteration 695 iteration 700 iteration 705 iteration 715 iteration 725		
iteration 665 iteration 670 iteration 675 iteration 685 iteration 690 iteration 700 iteration 705 iteration 715 iteration 715 iteration 720 iteration 720 iteration 720		
iteration 670 iteration 680 iteration 685 iteration 690 iteration 700 iteration 705 iteration 715 iteration 715 iteration 720 iteration 720		
iteration 675 iteration 680 iteration 685 iteration 690 iteration 795 iteration 705 iteration 710 iteration 715 iteration 725		
iteration 680 iteration 685 iteration 690 iteration 705 iteration 710 iteration 715 iteration 720 iteration 720		
iteration 685 iteration 690 iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725		
iteration 690 iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725		
iteration 695 iteration 700 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725		
iteration 700 iteration 705 iteration 710 iteration 715 iteration 720 iteration 725		
iteration 710 iteration 715 iteration 720 iteration 725	 iteration	700
iteration 715 iteration 720 iteration 725	 iteration	705
····· iteration 720		
···· iteration 725		
reration /30		
	 ireration	7.30

····· iteration 735 ····· iteration 740 ····· iteration 745 ····· iteration 750 ····· iteration 755 ····· iteration 760 ····· iteration 765 ····· iteration 770 ····· iteration 775 ····· iteration 780 ····· iteration 785 ····· iteration 790 ····· iteration 795 ····· iteration 800 ····· iteration 805 ····· iteration 810 ····· iteration 815 ····· iteration 820 ····· iteration 825 ····· iteration 830 ····· iteration 835 ····· iteration 840 ····· iteration 845 ····· iteration 850 ····· iteration 855 ···· iteration 860 ····· iteration 865 ····· iteration 870 ····· iteration 875 ····· iteration 885 ····· iteration 890 ····· iteration 895 ····· iteration 900 ····· iteration 910 ····· iteration 915 ····· iteration 920 ····· iteration 925 ····· iteration 930 ····· iteration 935 ····· iteration 940 ····· iteration 945 ····· iteration 950 ····· iteration 955 ····· iteration 960 ···· iteration 965 ····· iteration 970 ····· iteration 975 ···· iteration 980 ····· iteration 985 ····· iteration 990 ····· iteration 995 ····· iteration 1000 — final test

In the above plot the test data is plotted for n =0.01 and error is measured. The error rate in this case is 0.0036

actual test

Comparing all the 4 scenarios, we see that the error rate for n = 100 is 10.3 n = 1 is 0 n = 0.01 is 0.00092 n = 0.0001 is 0.0036