[WS13/14] Mathematics for Robotics and Control Assignment 003 - Matrix Decomposition

In [40]:
```
import IPython.core.display
import sys
if not "win" in sys.platform and not "linux" in sys.platform:
    %pylab
else:
    %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

Function and Module List

Each week, the assignment sheet will contain a list of Python modules and functions you are supposed to know. These modules and functions will help you to solve the assignments and many of them are required in later assignments. You are expected to familiarize yourself with each module / function listed here and to know when and how to apply them. Remember this is especially important in the exam!

Modules:

- numpy.linalg

Functions:

numpy.linalg:

- qr, svd

Assignment 3.1 [L1]

Take the following matrix and obtain it's eigenvalues:

$$A_0 = \begin{bmatrix} 1.35264537e+00 & -2.83012728e-03 & 3.75980444e-01 \\ -2.83012728e-03 & 1.00491214e-02 & -2.40593554e-03 \\ 3.75980444e-01 & -2.40593554e-03 & 4.51412148e+00 \end{bmatrix}$$

Now obtain the QR decomposition of A_0 and implement the following algorithm:

1. $A_{k+1} = Q_k^{-1} \cdot A_k \cdot Q_k$

...and run it for at least 1500 iterations of k. What do you notice about the resulting matrix A_{1500} ?

In [41]:
```
# Solution 3.1
# ...
sys.setrecursionlimit(1600)
A = array([1.35264537e+00, -2.83012728e-03, 3.75980444e-01, -2.83012728e-03,
 1.00491214e-02, -2.40593554e-03, 3.75980444e-01, -2.40593554e-03, 4.5141214
8e+00]).reshape(3,3)
w,v = eig(A)
print w, "\n"
def qr_decomposition(a, limit, current):
    if current == limit:
        return a
    q, r = qr(a)
    return qr_decomposition((inv(q).dot(a)).dot(q), limit, current + 1)
print qr_decomposition(A, 1500, 0)

##Eigen values of A are the same as diagonal of the final Matrix.
```

```
[ 4.55822169  1.30855172  0.01004256]

[[  4.55822169e+000  -3.86695267e-016  -5.40758374e-016]
 [ -4.94065646e-324   1.30855172e+000   3.94169997e-016]
 [ -1.48219694e-323   4.94065646e-324   1.00425639e-002]]
```

Assignment 3.1 took me minutes.

Assignment 3.2 L1

Watch the following video and write a summary about half a DINA4 page long.

In [42]:
```
IPython.display.YouTubeVideo("OSGv2VnC0go")
```

Out[42]:

Summary here...

Assignment 3.2 took me minutes.

- Index manipulation is discouraged, use python's looping idioms.
- for in python is equivalent of for-each.
- range() and xrange can be used to loop over a list. range() produces a list, which can be disastrous for big values, xrange should be used for large numbers it produces numbers one a time. for i in xrange(600000): print i
- Looping over a collection: for color in colors: print color
- reversed() can be used for looping backward.
- enumerate() can be used to keep track of indices.
- For looping over two collections zip and izip can be used, latter should be preferred. zip uses too much memory, izip creates one tuple at a time:
- Looping in sorted order (reverse=true sorts in reverse order), (key=len defines the criteria for sorting):
- Iterator iter() takes 2 arguments the function and the sentinel value.
- for-else clause shoulde be used to return default value when loop has reached its limit.
- Looping over dictionaries returns key. iteritems() can be used to return key and value.
- dict and izip can be used to build dictionaries.
- get() and defualtdict() can be used for counting with dictionaries.
- defaultdict() can be used to group.
- popitem() is threadsafe.
- ChainMap() for linking dicitionaries.
- keyword arguments, named tuples, unpack improve readability- join() should be used join strings.
- Decorators seperate business logic from adiminstrative logic.
- Context() saves old values and restores it when exiting context.
- with() should be used when opening files.
- ignored() can be use to ignore exceptions.
- List comprehensions are one liners and better for looping.---

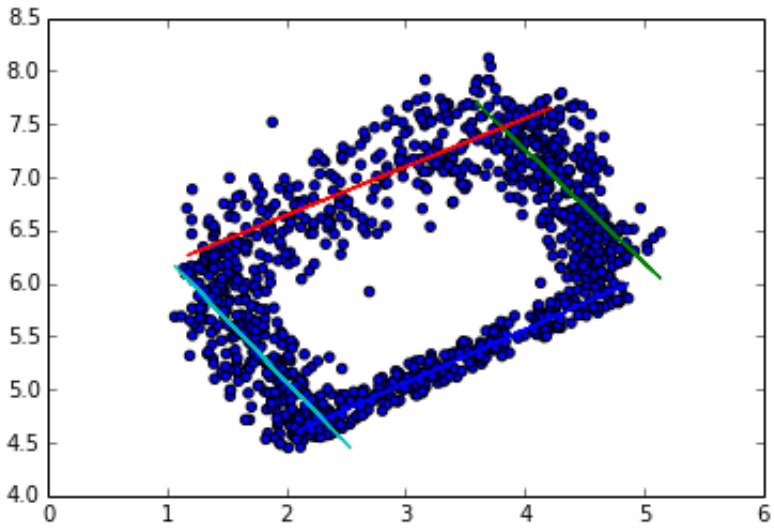
Assignment 3.3 L3

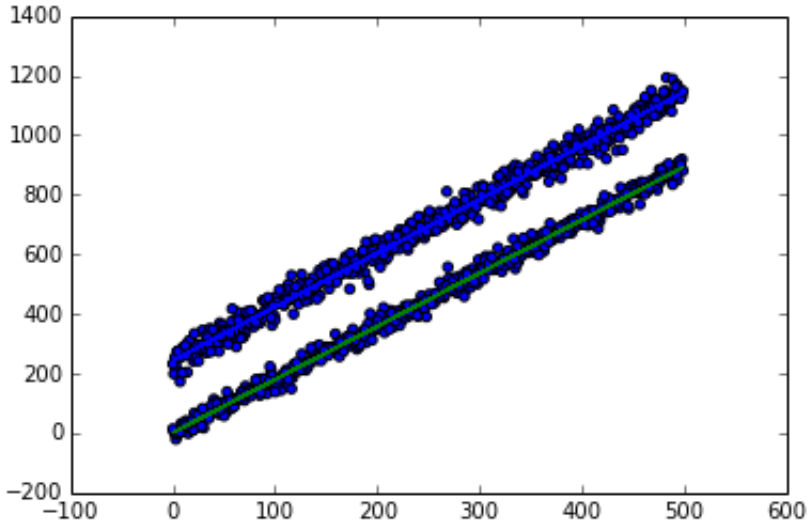
Read this article about least squares fitting (http://mathworld.wolfram.com/LeastSquaresFitting.html). Read this paper about linear least squares and matrix decompositions (http://classes.soe.ucsc.edu/cmps290c/Spring04/paps/lls.pdf). **For all of the following fitting tasks, use a singular value decomposition to fit a shape to each of the given point clouds. Show your results by plotting bot the original points and the fitted shape. Determine what kind of shape to use by inspecting the data points given to you. For some tasks, you will have to use more than one shape (i.e. fit two parallel or orthogonal lines).**

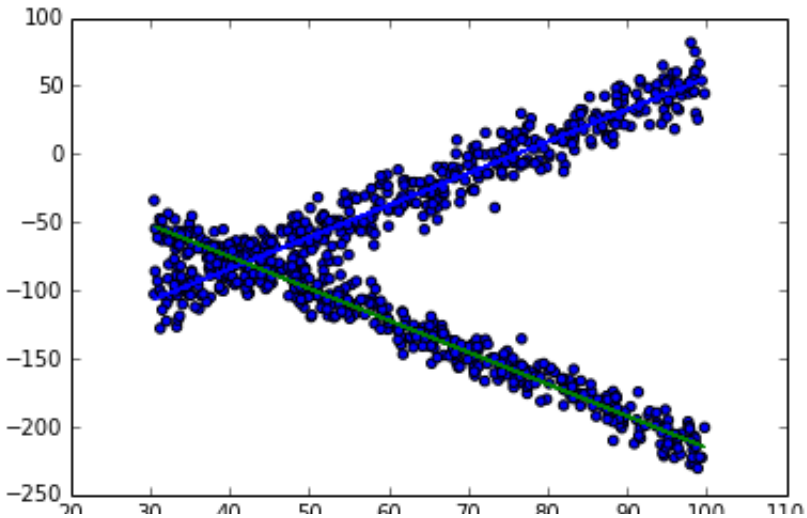
In [44]:

```
# Solution 3.3
# ... Using new data.
import numpy.linalg
from scipy.cluster.vq import kmeans, vq

def get_y_values(c, m, x):
    y = []
    for value in x:
        #calculate y for each value of x
        y.append(m * value + c)
    return y

def fit_line(fig):
    x = fig[:, 0]
    y = fig[:, 1]
    shape_first_column = shape(fig)[0]
    matrix_shape = (shape_first_column, 1)
    x = x.reshape(matrix_shape)
    y = y.reshape(matrix_shape)

    #form the 'A' Matrix by hstacking x to a vector of 1s
    A = hstack((ones((len(x), 1)), x))

    #calculate the svd of 'A'
    u, s, v = svd(A)

    #since we are inversing A tranpose the matrices
    u, v = u.T, v.T

    #vstack sigma 's' with zero matrix, and then inverse it
    s = pinv(vstack((diag(s), zeros((shape_first_column - 2, 2)))))

    #calculate the final result
    result = ((v.dot(s)).dot(u)).dot(y)

    #set the constant and slope
    c, m = result[0], result[1]

    #calculate y values using the given x coordinate, slope and constant
    final_y = get_y_values(c, m, x)

    return x, y, final_y

def plot_data(data):
    dimensions = shape(data)
    for dimension in data:
```

```
        x, y, final_y = fit_line(dimension)
        scatter(x, y)
        plot(x, final_y)
    show()

plot_data(load("shape0001.npy"))
plot_data(load("shape002.npy"))
plot_data(load("shape003.npy"))
```







20 30 40 50 60 70 80 90 100 110

Assignment 3.3 took me minutes.

Use this button to create a .txt file containing the time in minutes you spent working on the assignments. Make sure to include your name in the textbox below. The file will be created in the current directory.

Student's name:

Save timings