EE266 and MS&E251: Introduction to Julia

Installation

Basic types and operations

How to plot ?

Installation

Installation

- Download julia version (v0.4.5) http://julialang.org/downloads/
- Follow the instructions !
- You can also use Jupyter notebook. Use :

```
Pkg.add("IJulia")
using IJulia
notebook()
```

in julia terminal or directly `jupyter notebook` in your terminal.

Basic types and operations

Basic Types

- integers : Int64, e.g., -135
- real numbers : Float64, e.g., 1.23, 3.77e-7
- to force Julia to interpret an integer as a real number, use 2.0
- booleans : Bool, true or false
- strings : ASCIIString, e.g., "Hello, world!"

Vectors and Matrices

- Initialization : `x = zeros(n)`, `ones(n,m)`
- Random uniform : `rand(n)`
- Random gaussian : `randn(n,m)`
- Other matrix initialization : `diagm(vect)`, Identity matrix `eye(n)`
- Basic functions : `sum(x)`, `length(x)`, `mean(x)`, `var(x)`, `norm(x)`, `maximum(x)`, `minimum(x)` (different from `min(a,b)`),...
- Matrix specific functions : `nrows, ncols = size(A)`; `inv(A)`,...

Differences Matrix/Vector

Be careful with the shape of your vectors, because can be of type vector or matrix :

```
julia> a = [1,2,3]
3-element Array{Int64,1}:
 1
 2
 3

julia> b = [ 1 2 3 ]
1x3 Array{Int64,2}:
 1  2  3

julia> c = [1;2;3]
3-element Array{Int64,1}:
 1
 2
 3
```

Indexing/slicing

- Indexing : `a[1]`
  Warning : begins at 1, not 0 !
- Slicing : `a[2:3]`
  Warning : 2 and 3 are included !
- For matrix : `A[2,:]` takes the second row
- Transpose : `a'`
  Warning : `a'` is a `1x3 ArrayInt64,2`

Matrix operations

- Componentwise operation : `a + 1`, `a * 2`, `a .^2`
- Sum : `a + c`
- Componentwise product : `a .* c`
- Inner product : `b * a`
- Outer product : `a * b`
- Horizontal concatenation : `[a c]`
- Vertical concatenation : `[a ; c]`

Other data containers

- List : Arrays are in fact mutable we can add some elements `push!(a, 4)` Warning cannot push if it is not the good type !
- ... Except for nested list : `z = [1, "hello"]`. The type will be `Any` Warning : only possible since latest versions.
- Dictionnary :

```
dict = Dict("a" => 1, "b" => 2, "c" => 3)
dict["d"]=4 # add the "d" key
```

Loops/If

```
# FOR loop
values = zeros(10)
for i = 1:10
    println("i = ", i)
    values[i] = i^2
end

# WHILE loop
while i <= 5
    println(i)
    i += 1
end

# IF statement
if x < 2
    println("x<2")
elseif x == 2
    println("x=2")
else
    println("x>2")
```

How to plot ?

Packages installation

- I like to use PyPlot : same commands as in Python.
- Can also use Gadfly...
- To install and update PyPlot (or any other package):

```
Pkg.add("PyPlot")
Pkg.update()
```

- You just need to import this library at the begining of your script :
`using PyPlot`

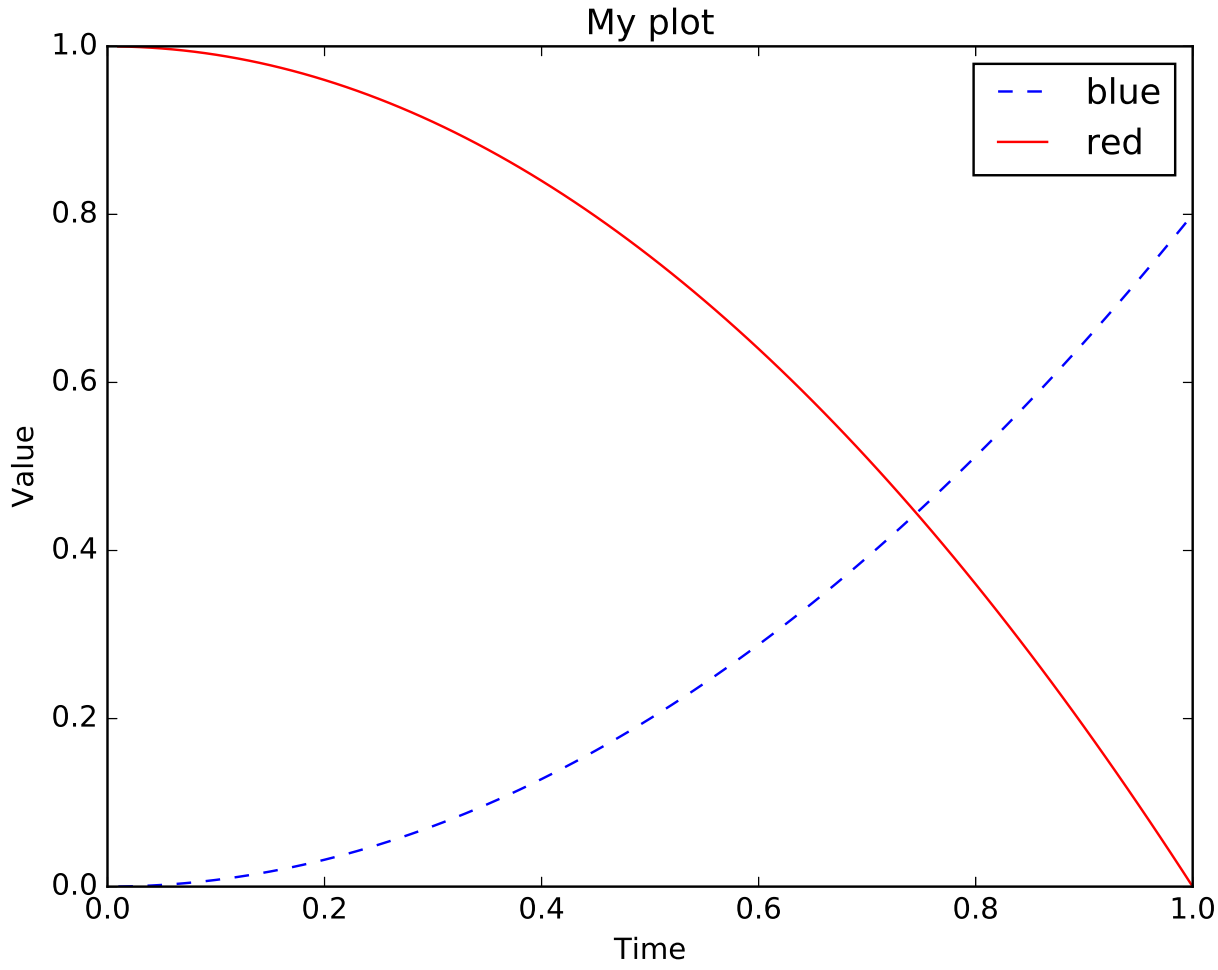
Examples

```
using PyPlot

vect = (1:100)*0.01

figure();
axis([0, 1 , 0, 1]) # or xlim(0,1); ylim(0,1)
plot(vect, 0.8*vect.^2,"b",linestyle="--", label = "blue")
plot(vect, 1-vect.^2,"r", label = "red")
xlabel("Time");
ylabel("Value");
title("My plot")
legend()
savefig("./plot.eps");
show()
```

Plot using PyPlot



- Can use instead of `plot` : `semilogx`, `semilogy`, `loglog`, `bar`, `hist`...

Homework 1: hist and subplot

```
using PyPLot

x = randn(1000); y = rand(1000)
nbins = 50

fig = figure("pyplot_histogram",figsize=(10,10))

subplot(211)
plt[:hist](x,nbins) # Histogram
title("Histogram of a gaussian")

subplot(212)
plt[:hist](y,nbins) # Histogram
title("Histogram of an uniform")

fig[:canvas][:draw]()
suptitle("2x1 Subplot")
```

Homework 1: hist and subplot