Python, III deo:
PyLab i SymPy

Šta je PyLab?

- environment koji cine
  - NumPy
  - SciPy
  - Matplotlib
  - IPython
- kako se instalira pylab?
- Ubuntu: Software Center ili Synaptic
- win: `http://www.enthought.com/`
- . . . Canopy

IDE? Spyder!

- potrebno?
- kako kome, kako za šta . . .
- IPython meni sasvim dobar . . .
- . . . ima i Qt verziju, IPython Qt console
- . . . mada postoji i Spyder
- `https://pypi.python.org/pypi/spyder`
- ako nema dovucite iz repository . . .
- ima i pod win, ali . . .

NumPy

- `http://www.numpy.org/`
- osnovna biblioteka za numerički zahtevne Python primene, sadrži:
  1. N-dimensional array object
  2. array slicing methods
  3. array reshaping methods

  i module za:
  1. basic linear algebra functions
  2. basic Fourier transforms
  3. advanced random number capabilities
- pokrenete `IDLE`
- `import numpy`
- `dir(numpy)`
- `help(numpy)`
- `del numpy`

NumPy

- obradićemo, sve, naravno, imamo vremena, . . .
- evolutivno nastao, „haotično“ iz Numeric i NumArray
- znanje koje raste
- Travis E. Oliphant & Enthought, biće još kontakata sa njima
- http://www.enthought.com/
- nema načina da se zapamti i nije „zauvek“
- potreban nov način učenja i snalaženja
- izbeći reinventing
- masovno korišćenje help-a i primera
- Matlab-Python-R

SciPy

- http://www.scipy.org/
- scientific lib za Python, zavisi od NumPy
- nekoliko modula in a single package, kao i NumPy
- moduli za:
  1. statistics
  2. optimization
  3. numerical integration
  4. linear algebra
  5. Fourier transforms
  6. signal processing
  7. image processing
  8. ODE solvers
  9. special functions

SciPy

- i dalje u `IDLE`
- `import scipy`
- `dir(scipy)`
- `help(scipy)`
- `del scipy`
- dobro razmislite pre nego što krenete u reinvent!
- ja ovo ne mogu da zapamtim, sto zapamtim zaboravim
- pomenuh li nov način učenja?
- kako organizovati informacije, previše ih je (i kratko traju)
- “Numerical Recipes”

matplotlib

- http://matplotlib.sourceforge.net/, sjajan sajt
- package sa dugim nizom modula
- jako dobro se vidi struktuiran namespace, package.modul
- Python 2D plotting library (samo 2D?)
- ono što ima sjajno je, a sada ima skoro sve
- galerija i primeri
- uputstvo, 3.0.0, pdf, 2288 strana, 21.09.2018.
- John D. Hunter, video lectures
- sintaksa vrlo liči na Matlab i Octave
- skoro kao gnuplot, ali direktno, bez izlaženja u gnuplot

IPython

- http://ipython.org/
- https://jupyter.org/
- interaktivni Python envirinment, vrlo nalik na Octave, wxMaxima, ...
- autocompletion by tab
- doteruje komande da budu shvaćene
- od mnogo mogućnosti: store, history, logging, ...

PyLab

- sve prethodno zgodno spakovano
- sređen namespace da ne mora puno dot notation
- sintaksa jako liči na Octave/Matlab
- počinjemo, komandna linija, terminal
- `ipython --pylab`

PyLab, IPython, osnovno

```
help()
?
%quickref
a = 3
b = 'string'
lista = [1, 2, 3]
li<tab>
whos
```

PyLab, IPython, store

```
store a
store b
store lista
del a
del b
a
b
store -r
a
b
store -z
store
```

Pylab, matematika, konstante

```
pi
e
j
1j
exp(1j * pi)
math.exp(1j * pi)
help(exp)
help(math.exp)
exp?
math.exp?
e**(1j * pi) + 1
```

Pylab, matrice 1

```
a = array([[1, 2], [3, 4]])
a
a.size
a.shape
a.ndim
a.dtype
a.dtype.name
a.itemsize
a.transpose()
transpose(a)
a.T
```

Pylab, matrice 2

```
det(a)
eig(a)
b = eig(a)
type(b)
len(b)
type(b[0])
type(b[1])
c, d = b
c.size
c.shape
c.ndim
c.dtype.name
c.itemsize
```

Pylab, matrice 3, inverzija i množenje

```
aa = inv(a)
aa
aa.dtype.name
aa * a
a * aa
dot(a, aa)
dot(aa, a)
x = arange(10)
x
print x
dot(x, x)
```

Sve operacije su **elementwise**!!!
Velika razlika u odnosu na **Octave**!!!

Pylab, gde je dot, tu je cross

```
i = array([1, 0, 0])
j = array([0, 1, 0])
print dot(i, j), dot(j, i)
print cross(i, j)
print cross(j, i)
cross?
source(cross)
```

Poznato od nekud? Rekoh da već ima, . . .

Teško je naći potpuno nov primer . . .

Reinventing problem! Google pomaže puno!

Pylab, rang!!!

```
a = array([[1, 1], [1, 1]])
rank(a)    # ???
help(rank)    # sad jasno?
linalg.matrix_rank(a)
```

Pazite!!!
Nažalost, lično iskustvo, ne tuđe!

Pylab, inicijalizacija, neke posebne matrice

```
I = eye(3)
print I
nule = zeros(4)
print nule
zeros?
zeros((4, 2))
zeros(4, 2)    # pazite!
zeros((4, 2), dtype = int)
zeros((4, 2), dtype = complex)
ones([3, 4])
empty([6, 4])
empty?
```

Pylab, još elementwise operacija

```
a + 1
a * a
a**3
a / 2
a / 2.
```

Pazite na /, velika razlika u odnosu na Octave!!!

Pylab, reshape

```
help(reshape)
print a
a.reshape(1, 4)
a.reshape(4, 1)
reshape(a, (1, 4))
reshape(a, (4, 1))
a.reshape(2, 3)
a.reshape(4)
a.reshape((4, ))
reshape(a, 4)
reshape(a, (4, ))
```

Pylab, in place array operations

```
xx = arange(20)
print xx
xx += 1
print xx
xx = xx.reshape(4, 5)
print xx
xx -= 1
print xx
xx *= 2
print xx
```

Pylab, transpose, revisited

```
help(transpose)
xx = arange(5) + 1
print xx
xx.shape
xx.ndim
xx = xx.transpose()
xx.shape
print xx
xx = xx.reshape(1, 5)
xx.ndim
print xx
xx.shape
xx = xx.transpose()
xx.shape
print xx
```

Pylab, eig, revisited

```
a = eye(2)
print a
b = eig(a)
type(b)
len(b)
c, d = b
print c
print d
a[0, 1] = 1
print a
b = eig(a)
c, d = b
print c
print d
```

Pylab, plot 1

```
x = arange(10) + 0.5 # nikako 1/2
y = x**2
plot(x, y)
plot(x, y, ’r+’)
xlabel(’x’)
ylabel(’y’)
title(’y = x**2’)
```

pretty matlabeće, ali nema `hold on`
save kako hocete, odaberete format
u IDLE za ovo je bilo potrebno ion() (zaglavljivanje!)
pogledati `show()` i `ion()` help

`close()` ili `close(’all’)`

Pylab, plot 2

```
close(’all’)
phi = linspace(0, 2 * pi, 1000)
x = 2 * cos(phi)
y = 2 * sin(phi)
plot(x, y)
axis([-3, 3, -3, 3])
axis(’equal’)
axis([-3, 3, -3, 3])
axis([-4, 4, -3, 3])
title(’kruznica’)

axis?
```

Pylab, plot 3

```
close(’all’)

help(figure)

figure(figsize = (6, 6))
plot(x, y)
axis(’equal’)
axis([-3, 3, -3, 3])
title(’kruznica’)
```

Pazite na redosled `’equal’` i `[-3, 3, -3, 3]`!!!

Pylab, plot 4, histograms

```
x = rand(10000)
plot(x)
close()
rand?

hist(x, 100)
close()

hist?
```

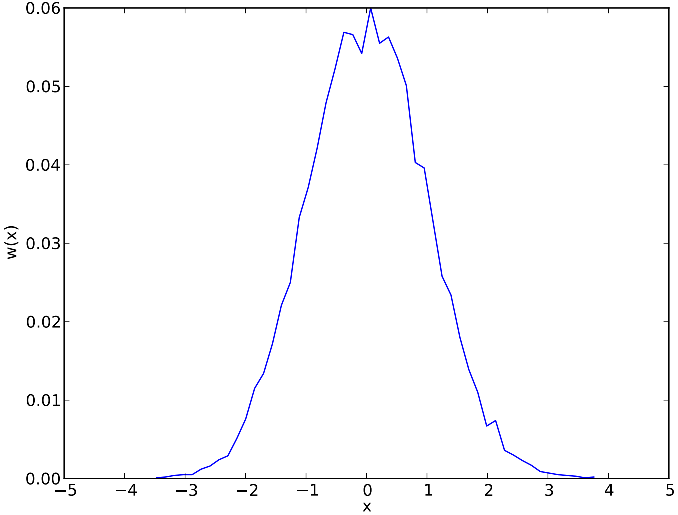
Pylab, plot 5, normal distribution

```
x = randn(10000)
plot(x)
close()

randn?

hist(x, 100)
close()

t = hist(x, 50)
type(t)
len(t)
len(t[0])
len(t[1])
```

Pylab, plot 6

```
y = t[0] / 10000.0   # pazite kod /!
x = t[1]
len(y)
len(x)
xx = (x[0 : len(x)-1] + x[1 : len(x)]) / 2
len(xx)
close()
plot(xx, y)
xlabel(’x’)
ylabel(’w(x)’)
xlim(-5, 5)
xticks(linspace(-5, 5, 11))
sum(y)
help(savefig)
savefig(’slika’) # potrazite slika.png
savefig(’slika.pdf’) # potrazite slika.pdf
```

slika.pdf



Pylab, jedan script, dat.py

```
from pylab import *

deg = linspace(0, 4*360, 4*360 + 1)
wt = radians(deg)

f = exp(- wt / 2 / pi * 0.5) * sin(wt)
fp = exp(- wt / 2 / pi * 0.5)
fm = -fp

dat = array([deg, wt, f, fp, fm]).transpose()

np.save(’deg.npy’, deg)
np.save(’f.npy’, f)
np.save(’fp.npy’, fp)

np.savetxt(’dat.txt’, dat, fmt=’%.4f’)
```

Pylab, run, #1

na komandnoj liniji:

```
python dat.py
more dat.txt
less dat.txt
ls *.npy
ipython --pylab
```

Pylab, run, #2 i #3

inside IPython:

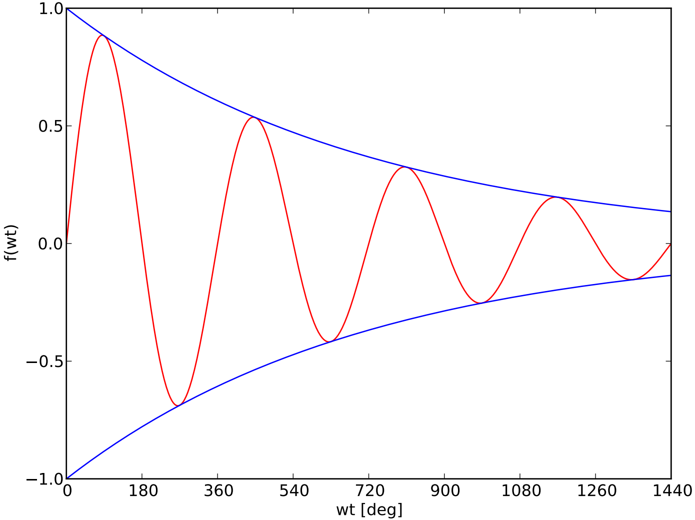
```
run dat.py
execfile('dat.py')
```

`execfile` radi i kod IDLE

Pylab, after the run

```
np.save?
np.savetxt?
np.load?
deg = np.load('deg.npy')
f = np.load('f.npy')
fp = np.load('fp.npy')
fm = -fp
plot(deg, f, 'r')
plot(deg, fp, 'b')
plot(deg, fm, 'b')
xlim(0, 360 * 4)
xticks(arange(0, 360 * 4 + 1, 180))
xlabel('wt [deg]')
ylabel('f(wt)')
savefig('datslik')
close()
```

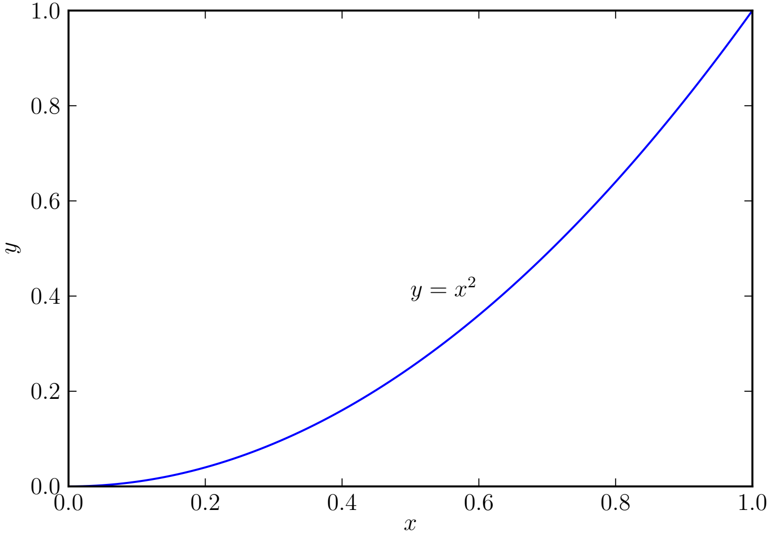
datslik.pdf



Pylab, LaTeX, estetika

```
help(rc)

close(’all’)
x = linspace(0, 1, 101)
y = x**2
rc(’text’, usetex = True)
rc(’font’, family = ’serif’)
figure(0, figsize = (6, 4))
plot(x, y)
xlabel(r’$x$’)
ylabel(r’$y$’)
text(0.5, 0.4, r’$y = x^2$’)
savefig(’kvadrat.pdf’, bbox_inches = ’tight’)
```

kvadrat.pdf



Pylab, LaTeX, „standardna“ „preambula“

```
rc(’text’, usetex = True)
rc(’font’, family = ’serif’)
rc(’font’, size = 12)
rcParams[’text.latex.preamble’]=[r’\usepackage{amsmath}’]
```

Pylab, linear algebra, 0th part

hoću da rešim sistem jednačina:

$$x + y = 3$$
$$x - y = 1$$

```
a = array([[1, 1], [1, -1]])
print a
b = array([[3], [1]])
print b
x = solve(a, b)
print x
b = array([3, 1])
print b
x = solve(a, b)
print x
```

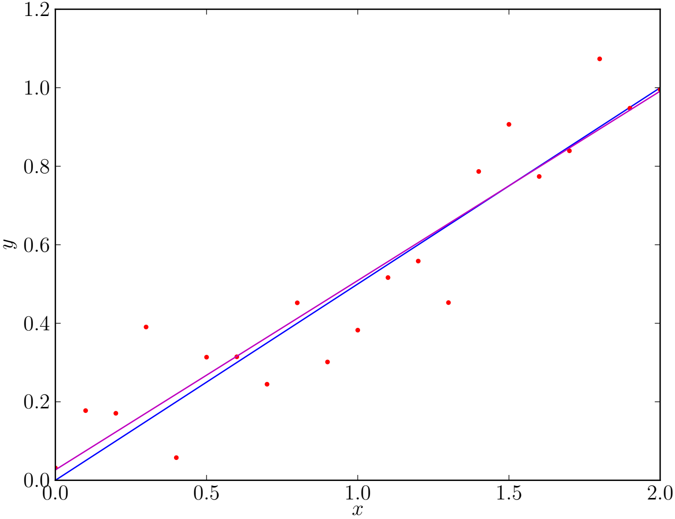
Pylab, linear algebra, 1st part

```
randn?
linalg.lstsq?

x = linspace(0, 2, 201)
y = x / 2

xe = linspace(0, 2, 21)
ye = xe / 2 + randn(21) * 0.1

A = array([xe, ones(len(xe))]).transpose()
t = linalg.lstsq(A, ye)
print t
type(t)
len(t)
a = t[0][0]
b = t[0][1]
```

Pylab, linear algebra, 2nd part

```
close('all')

rc('text', usetex=True)
rc('font', family='serif', size='16')

plot(x, y, 'b')
plot(xe, ye, 'r.')

yfit = a * x + b
plot(x, yfit, 'm')
xlabel(r'$x$')
ylabel(r'$y$')
savefig('fitovanje.pdf')
```

fitovanje.pdf



Šta je SymPy?

- paket za simboličko računanje koristeći Python
- cilj: isto što i wxMaxima
- ideja: Python sintaksa, poznata
- moduli, funkcije za simboličko računanje
- `http://sympy.org/en/index.html`
- `https://github.com/sympy/sympy/releases`
- `http://live.sympy.org/`
- uputstvo 1.3, 2044 strane, 14.09.2018; čitate stranu po stranu?
- Ubuntu: Software Center ili Synaptic
- volite Mathematica sintaksu?
- **odličan pregled i tutorial**
- za one koji vole matematiku `http://www.sagemath.org/`

SymPy, počinjemo

- komandna linija, isympy
- stari poznanik, IPython, customized again
- pogledajte šta radi na početku, dekleracije `symbols` . . .

SymPy, polinomi

```
p1 = x**2 + 2 * x + 1
p1
print p1
pprint(p1)
p2 = x + 1
p2
p3 = x**2 - 4
p3
p1 * p2 * p3
_.expand()
```

SymPy, expand i apart

```
p1 / p2
_.expand()
p1 / p2
_.apart()
apart(p1 / p2, x)
help(expand)
help(apart)
f = 1 / (x**2 * (x + 1))
apart(f, x)
together(_, x)
_.expand()
_.factor()
_.subs(x, 3)
```

SymPy, opet problemi sa deljenjem

```
1 / 3
Rational(1, 3)
S('1/3')
S('1 / 2')
S('1/2') - S('1/3')
```

SymPy, konstante i izračunavanje

```
p2 = pi**2
p2
p2.evalf()
p2.evalf(100)
p2.n()
p2.n(200)
N(p2)
N(p2, 300)
exp(1)
e
E
E.evalf()
I**2
exp(I * pi) + 1
E**(I * pi) + 1
```

SymPy, Ramanujan

```
r = E**(pi * sqrt(163))
r
r.n(20)
r.n(21)
r.n(22)
r.n(23)
r.n(24)
r.n(25)
r.n(28)
r.n(30)
r.n(35)
r.n(500)
```

SymPy, realni brojevi

```
exp(I * x).expand(complex = True)
xr = Symbol('xr', real = True)
exp(I * xr).expand(complex = True)
```

SymPy, linearna algebra

```
e1 = 2 * x + y - 4
e2 = x - y + 1
e1
e2
solve((e1, e2), (x, y))

e3 = x - y + 4
e2
e3
solve((e2, e3), (x, y))

e4 = 3 * x - 3 * y + 12
e3
e4
solve((e3, e4), (x, y))
```

SymPy, linearna algebra, opet isto . . .

```
e1 = Eq(2 * x + y, 4)
e1
e2 = Eq(x - y, -1)
e1
e2
solve((e1, e2), (x, y))

e3 = Eq(x - y, -4)
e2
e3
solve((e2, e3), (x, y))

e4 = Eq(3 * x - 3 * y, -12)
e3
e4
solve((e3, e4), (x, y))
```

SymPy, matrice, samo osnovno

```
A = Matrix([[x, 1], [1, y]])
A
print A
pprint(A)
A**2
A**(-1)
B = A.inv()
B
A**(-1) - B
A * B
simplify(A * B)
```

SymPy, ne baš linearna algebra

```
solve(p1, x)
solve(p2, x)
solve(p3, x)
solve(x**4 - 1, x)
solve(Eq(x**4, 1), x)
```

SymPy, limesi

```
f = sin(14 * x) / x
g = 1 / x
limit(f, x, 0)
limit(g, x, 0)
limit(g, x, 0, dir = ’+’)
limit(g, x, 0, dir = ’-’)
limit?
```

SymPy, izvodi

```
print p1
pprint(p1)
diff(p1, x)
diff(p1, x, 2)
diff(p1 * p2 * p3, x)
_.expand()
diff(p1 * p2 * p3, x, 14)
diff(p1*exp(3*x), x)
_.expand()
```

SymPy, integrali

```
integrate(p1, x)
integrate(p1, (x, 1, 2))
integrate(cos(x), x)
integrate(sin(x), (x, 0, pi))
```

SymPy, Taylor

```
sin(x).series(x, 0, 10)
series(sin(x), x, 0, 10)
series?
series(sin(x), x, 1, 10)
series(exp(x), x, 0, 5)
series(exp(x), x, 5, 5)
source(series)
```

SymPy, da obrišemo funkciju

```
f
del f
f = Function('f')
f
```

SymPy, diferencijalne jednačine

trigonometrijske funkcije kao rešenje

```
f(t).diff(t, 2) + 4 * f(t)
dsolve(_, f(t))
f(t).diff(t, 2) + 4 * f(t) - 24 * cos(4*t)
dsolve(_, f(t))
```

i eksponencijalne . . .

```
f(x).diff(x, 2) - 4 * f(x)
dsolve(_, f(x))
Eq(f(x).diff(x, 2) - 4 * f(x), - 4 * exp(- 2 * x))
dsolve(_, f(x))
```

SymPy, diferencijalne jednačine, provera

```
deq = Eq(f(t).diff(t, 2) - 4 * f(t), - 4 * exp(-2*t))
deq
deq.lhs
deq.rhs
sol = dsolve(deq, f(t))
sol
sol.lhs
sol.rhs
s = sol.rhs
s
ver = deq.subs(f(t), s)
ver
ver = ver.doit()
ver
ver.lhs.expand()
ver.rhs
ver.expand()
```

SymPy, diferencijalne jednačine, script, problem

Rešiti diferencijalnu jednačinu

$$\frac{d^2 f(t)}{dt^2} + 2\frac{d f(t)}{dt} + 4 f(t) = 2 sin(t)$$

i proveriti rešenje.

Napisati Python script koji rešava problem, prikazati jednačinu, rešenje i rezultat provere.

SymPy, diferencijalne jednačine, script

```
from __future__ import division
from sympy import *

# cast
y, t = symbols(’y t’)
f = symbols(’f’, cls=Function)

# equation
deq = Eq(f(t).diff(t, 2) + 2 * f(t).diff(t, 1) + 4 * f(t), 2 * sin(t))
print
pprint(deq)

# solution
sol = dsolve(deq, f(t))
print
pprint(sol)

# verification
s = sol.rhs
ver = deq.subs(f(t), s)
ver = ver.doit()
ver = ver.expand()
print
pprint(ver)
```

SymPy, diferencijalne jednačine, script, rešenje

```
                        2
           d           d
4*f(t) + 2*--(f(t)) + ---(f(t)) = 2*sin(t)
           dt           2
                      dt

       /      /  ___  \         /  ___  \\  -t   6*sin(t)   4*cos(t)
f(t) = \C1*sin\\/ 3 *t/ + C2*cos\\/ 3 *t//*e    + -------- - --------
                                                     13         13

True
```

SymPy: :, =, is()

- na kraju, a moglo je i na početku ...

Maxima	vs.	SymPy	objašnjenje
`:`	je	`=`	dodela vrednosti
`is()`	je	`==`	logički operator
`=`	je	`Eq(*, *)`	deklarisanje jednakosti

- očigledno postoji potreba za različitim =

in SymPy:

```
x**4 = 1       # error
x**4 == 1      # poređenje
Eq(x**4, 1)    # jednakost, jednačina
solve(_, x)    # koja može da se reši
```

PSAE, šta nismo uradili, a trebalo je, t $\ll$ (potrebnog)

- **komandna linija**, `http://linuxcommand.org/tlcl.php`
- **regular expressions**, Charles Severance, video, 35', 23''
- detaljnije Code::Blocks ili neki drugi IDE
- **Eclipse** `http://www.eclipse.org/`
- ukratko LibreOffice, više pravila pisanja
- malo više vremena za Octave, ipak je jako dobar program
- makar malo vremena za **SciLab**, Xcos pre svega
- malo vremena za **GIMP**, mada nije problem
- malo vremena za **Inkscape**, ovo je veći problem
- **Qucs** `http://qucs.sourceforge.net/`
- **Ngspice** `http://ngspice.sourceforge.net/`
- **moj izbor:** julia **i** sage
- **ovo je samo početak, mada je za sada . . .**

— KRAJ —