Wprowadzenie do GNU Octave

Janusz Szwabiński

Czym jest GNU Octave?

- środowisko (skryptowy język programowania + interpreter + GUI) do obliczeń numerycznych
- wolny[1] odpowiednik komercyjnego programu Matlab
- aktywnie rozwijany od 1992 roku
- posiada stabilne wersje na większości współczesnych systemów operacyjnych
- więcej na stronie `http://www.gnu.org/software/octave/`

[1] Patrz `http://www.debian.org/intro/free`

Czym GNU Octave nie jest?

- nie jest systemem do algebry symbolicznej (jak Mathematica, Maxima, Axiom)
- nie zawsze może dać dokładną odpowiedź na postawiony problem

Uwaga!!!

Większość rzeczywistych interesujących problemów matematycznych (zwłaszcza tych inżynierskich) i tak nie ma dokładnych, symbolicznych rozwiązań!!!

Dygresja - symbolicznie kontra numerycznie

Zadanie

Szukamy pierwiastków równania

$$x^2 - p = 0$$

Rozwiązanie symboliczne:

1. $p \geq 0$
   Wzór na różnicę kwadratów:
   $$(x - \sqrt{p})(x + \sqrt{p}) = 0$$
   $$x = \pm\sqrt{p}$$
2. $a < 0$
   $$(x - i\sqrt{-p})(x + i\sqrt{-p}) = 0$$
   $$x = \pm i\sqrt{-p}$$

   gdzie $i = \sqrt{-1}$

Dygresja - symbolicznie kontra numerycznie

Rozwiązanie numeryczne:

1. wiele metod numerycznych: metoda bisekcja, regula falsi, Brenta, siecznych, Newtona
2. każda z nich wymaga podstawienia za p konkretnej wartości!!!
3. niektóre wymagają dodatkowo określenia przedziału izolacji pierwiastka lub podania rozsądnego punktu startowego
4. metoda siecznych ma np. postać

$$\begin{aligned} x_0 &= a \\ x_1 &= b \\ x_{n+1} &= x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})} \end{aligned}$$

gdzie $f(x) = x^2 - p$, (a, b) - przedział izolacji pierwiastka

Dygresja - symbolicznie kontra numerycznie

Rozwiązanie numeryczne (c.d.):
Niech $p = 2, a = 1, b = 2$, czyli szukamy rozwiązania równania

$$x^2 - 2 = 0$$

w przedziale $(1, 2)$. Możemy wyliczyć kolejne przybliżenia metodą siecznych (z dokładnością do 4 miejsc po przecinku):

Krok	x	f(x)
x_2	$1,3333$	$-0,2222$
x_3	$1,4000$	$-0,0400$
x_4	$1,4146$	$0,0012$
x_5	$1,4142$	0
x_6	$1,4142$	0

Możliwości GNU Octave

- obliczanie wartości wyrażeń (w tym wyrażeń zawierających zaawansowane funkcje matematyczne, np. funkcje zespolone)
- znajdowanie wartości sum i iloczynów ciągów liczb o bardzo dużej liczbie elementów
- układy równań liniowych (mogących mieć nawet tysiące niewiadomych)
- regresja liniowa
- równania i układy równań nieliniowych
- obliczanie wartości całek oznaczonych
- równania różniczkowe zwyczajne i cząstkowe
- algebra liniowa (iloczyny wektorów i macierzy, wartości i wektory własne, rozkład macierzy itp.)
- prezentacja rozwiązań w postaci wykresów

Dlaczego nie C++?

- stopień skomplikowania języka przeszkadza w rozwiązywaniu właściwych zagadnień matematycznych
- brak natywnego wsparcia dla niektórych koncepcji matematycznych
- brak natywnego wsparcia dla wizualizacji danych
- jeżeli nawet ostatecznie wydajność obliczeń wymaga implementacji w C++ (lub podobnym języku), algorytmy matematyczne często testowane są w GNU Octave

Dlaczego nie Python?

- Python z modułami `numpy`, `scipy` i `matplotlib` oferuje praktycznie podobną funkcjonalność
- składnia jest również podobna do składni Matlab/Octave
- język jest zwięzły, jednak kod z reguły jest nieco dłuższy niż odpowiedni kod w Octave
- . . . jednak od lat niepisanym standardem środowisk numerycznych jest Matlab, dlatego znajomość jego lub jego darmowego klona może okazać się przydatna na rynku pracy[2]

[2] Na kursie z Pythona będzie wykład poświęcony jego numerycznym możliwościom.

Dlaczego nie Python?

expint.m

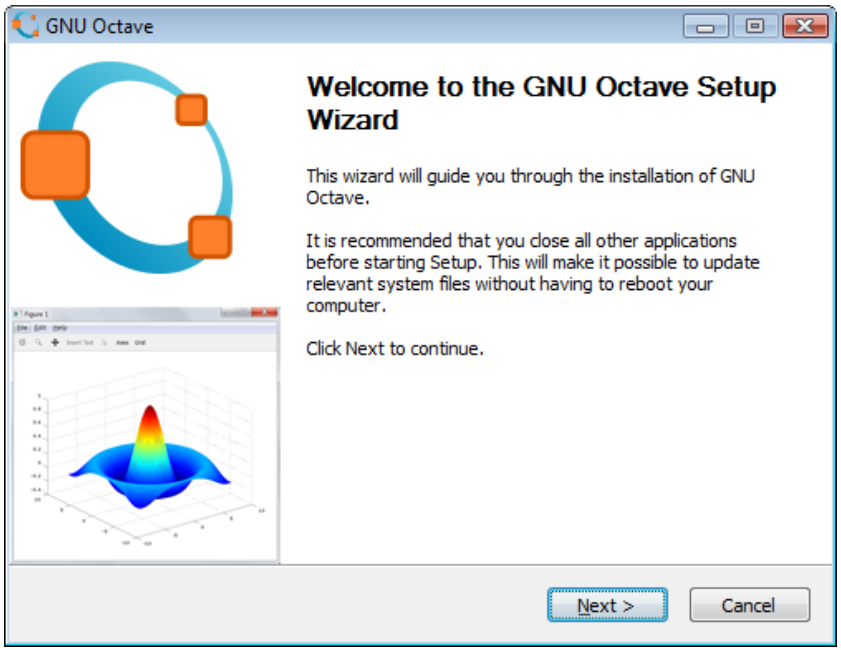
```
% plot the integrand and approximate
% the integral
%    / 1
%    |     exp(-x^2/pi) dx
%    / 0
% by left-hand, right-hand, and
% trapezoid rules

N = 1000;
dx = (1 - 0) / N;
x = linspace(0,1,N+1);
y = exp(- x.^2 / pi);

plot(x,y)
axis([0 1 0 1]), grid

format long
lhand = dx * sum(y(1:end-1))
rhand = dx * sum(y(2:end))
trap  = (dx/2) * sum(y(1:end-1)+y(2:end))
exact = (pi/2) * erf(1/sqrt(pi))
```

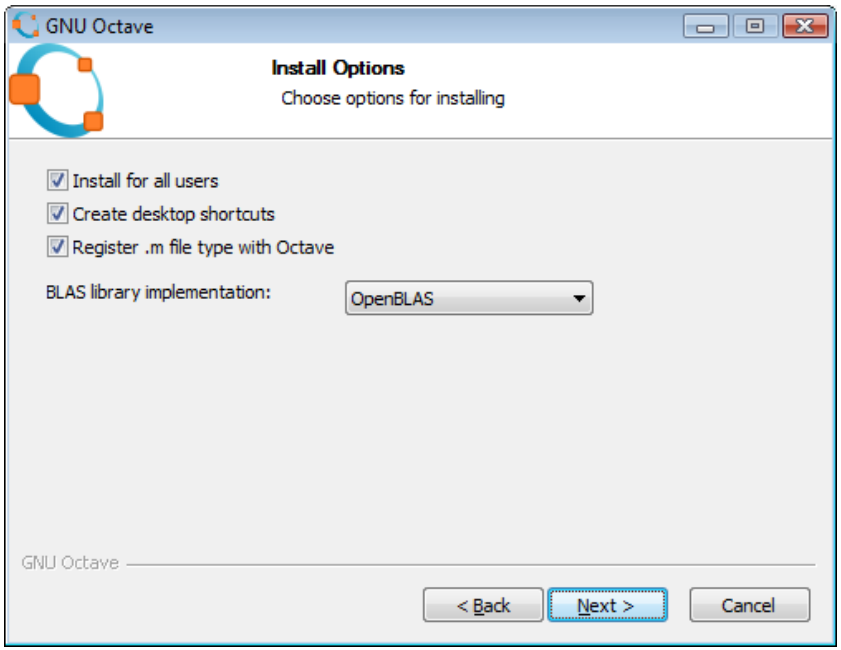
expint.py

```
#!/usr/bin/env python

# plot the integrand and approximate
# the integral
#    / 1
#    |     exp(-x^2/pi) dx
#    / 0
# by left-hand, right-hand, and
# trapezoid rules

from pylab import plot,axis,linspace,sum, \
                  pi,sqrt,exp,show,grid
from scipy.special import erf

N = 1000
dx = (1.0 - 0.0) / N
x = linspace(0.0,1.0,N+1)
y = exp(- x**2 / pi)

plot(x,y)
axis([0.0,1.0,0.0,1.0]); grid(True)

lhand = dx * sum(y[:-1])
print "lhand = %.15f" % lhand
rhand = dx * sum(y[1:])
print "rhand = %.15f" % rhand
trap  = (dx/2) * sum(y[:-1]+y[1:])
print "trap = %.15f" % trap
exact = (pi/2) * erf(1/sqrt(pi))
print "exact = %.15f" % exact
show()  # allow user to close figure
```

Instalacja GNU Octave - Windows

- pakiet instalacyjny pod adresem `https://ftp.gnu.org/gnu/octave/windows/`



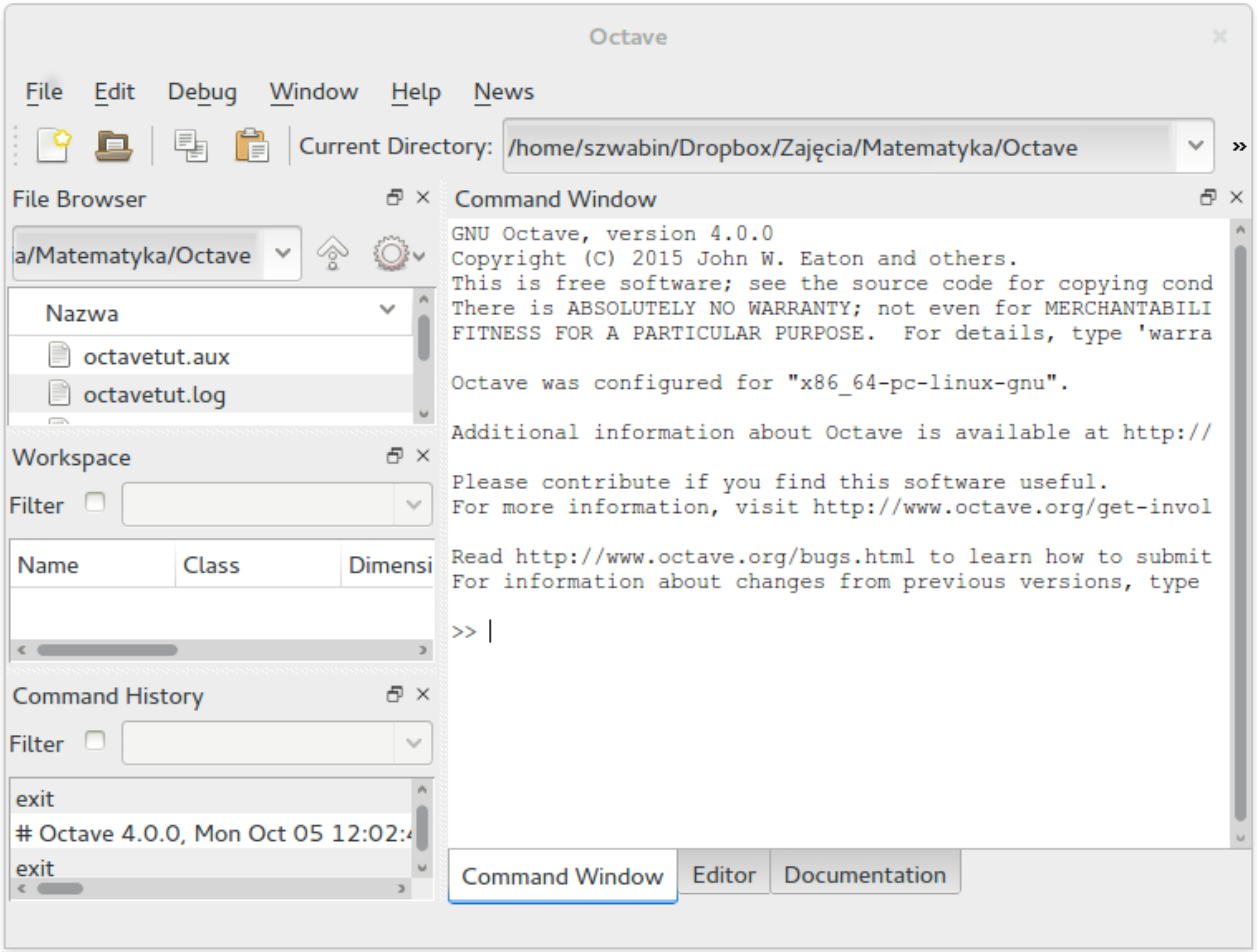


Instalacja GNU Octave - Linux

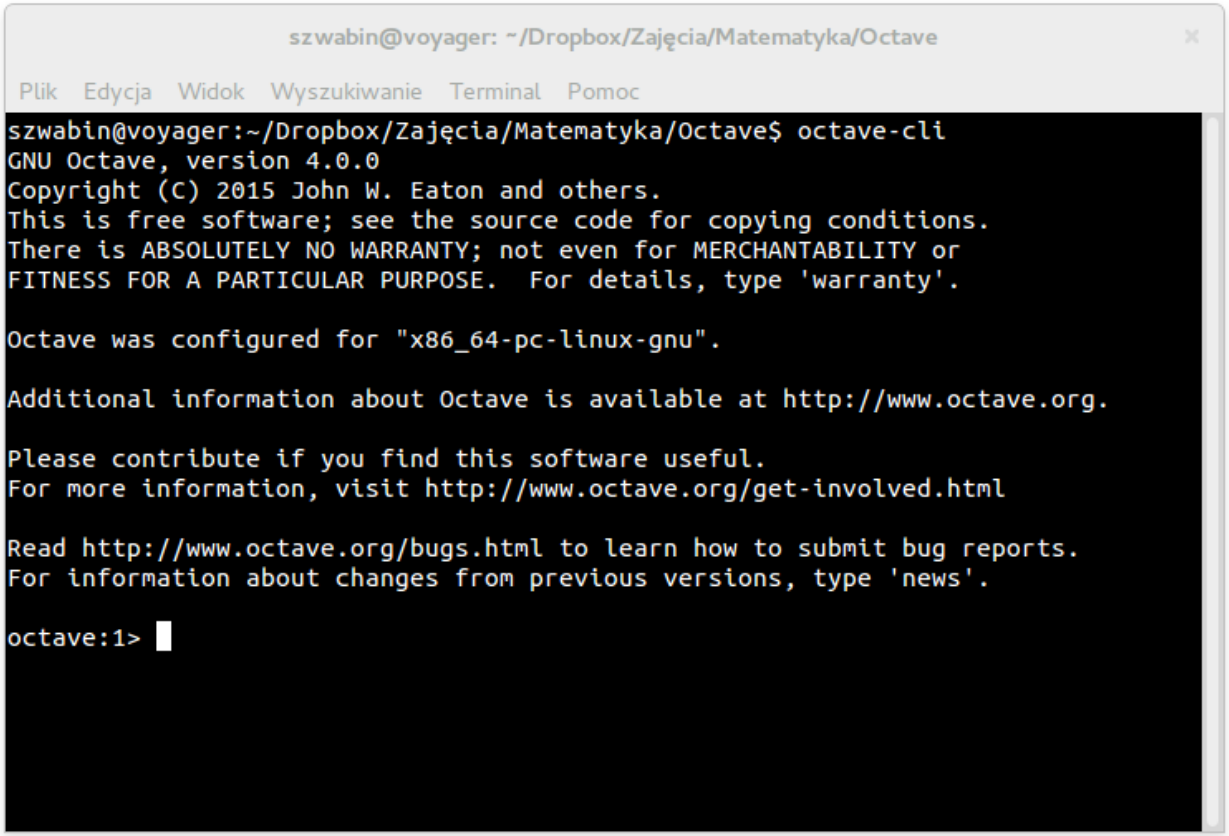
- źródła do pobrania na stronie `http://www.gnu.org/software/octave/download.html`
- wiele dystrybucji oferuje pakiety binarne w swoich repozytoriach
- jeżeli w dystrybucji jest starsza wersja, ciągle możemy znaleźć pakiety binarne w Internecie
- przykład na Ubuntu:

```
apt-add-repository ppa:octave/stable
apt-get update
apt-get install octave
apt-get install octave-info
```

Pierwsze kroki w GNU Octave



Pierwsze kroki w GNU Octave



Pierwsze kroki w GNU Octave

```
octave:1> help sin
'sin' is a built-in function from the file libinterp/corefcn/
    mappers.cc

 -- Mapping Function: sin (X)
    Compute the sine for each element of X in radians.

    See also: asin, sind, sinh.

Additional help for built-in functions and operators is
available in the online version of the manual. Use the command
'doc <topic>' to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help@octave.org
mailing list.
```

Pierwsze kroki w GNU Octave

```
octave:2> doc
File: octave.info, Node: Top, Next: Preface, Up: (dir)

GNU Octave
**********

This manual documents how to run, install and port GNU Octave, as well
as its new features and incompatibilities, and how to report bugs. It
corresponds to GNU Octave version 4.0.0.

* Menu:

* Preface::
* Introduction::              A brief introduction to Octave.
* Getting Started::
* Data Types::
* Numeric Data Types::
* Strings::
* Data Containers::
* Variables::
* Expressions::
* Evaluation::
<snip>
```

Pierwsze kroki w GNU Octave

```
octave:3> doc numeric
File: octave.info, Node: Numeric Objects, Next: Missing Data, Up: Built-in D\
ata Types

3.1.1 Numeric Objects
---------------------

Octave's built-in numeric objects include real, complex, and integer
scalars and matrices. All built-in floating point numeric data is
currently stored as double precision numbers. On systems that use the
IEEE floating point format, values in the range of approximately
2.2251e-308 to 1.7977e+308 can be stored, and the relative precision is
approximately 2.2204e-16. The exact values are given by the variables
'realmin', 'realmax', and 'eps', respectively.

   Matrix objects can be of any size, and can be dynamically reshaped
and resized. It is easy to extract individual rows, columns, or
submatrices using a variety of powerful indexing features. *Note Index
Expressions::.

   *Note Numeric Data Types::, for more information.
```

Reprezentacja liczb

- w GNU Octave liczby są przechowywane w postaci binarnej:

$$12,25_{10} = 1\times2^3+1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2} = 1101,01_2$$

- każdej liczbie przypisane są 64 bity
- więcej na ten temat pod adresem `https://www.gnu.org/software/gsl/manual/html_node/Representation-of-floating-point-numbers.html`

Działania na liczbach

```
octave:3> 1/7
ans = 0.14286
octave:4> 1/ans
ans = 7
octave:5> 1/0.14286
ans = 6.9999
octave:6> ans == 7 % sprawdź, czy równe?
ans = 0            % nie!
octave:7> 7 == 7
ans = 1
```

Dokładność obliczeń

```
octave:8> 1 - 0.2 - 0.2 - 0.2 - 0.2 - 0.2
ans =   5.5511e-17
```

- skończona liczba bitów . . .
- $\Rightarrow$ nie każdą liczbę można przedstawić dokładnie, np.

```
octave:9> 0.2
ans = 0.20000
octave:10> format bit % reprezentacja bitowa na wyjściu
octave:11> 0.2 % zwróć uwagę na pojawiający się cykl!!!
ans = 0011111111001001100110011001100110011001100110011001100110011010
octave:12> format
```

- w obliczeniach numerycznych trzeba z tym żyć
- należy być ostrożnym np. przy porównaniach, czy liczby są równe

Operatory arytmetyczne

```
octave:21> 2 + 2 % dodawanie
ans = 4
octave:22> 2 - 3 % odejmowanie
ans = -1
octave:23> 1/7 % dzielenie
ans = 0.14286
octave:24> 7\1 % dzielenie raz jeszcze
ans = 0.14286
octave:25> 4*5 % mnożenie
ans = 20
octave:26> 2**3 % potęgowanie
ans = 8
octave:27> 2^3 % potęgowanie raz jeszcze
ans = 8
```

Stałe matematyczne

```
octave:31> pi
ans = 3.1416
octave:32> format long
octave:33> ans
ans = 3.14159265358979
octave:34> format
octave:37> pi
ans = 3.1416
octave:38> sin(pi)
ans =   1.2246e-16
octave:39> cos(pi)
ans = -1
octave:40> sin(pi/2)
ans = 1
octave:41> e % liczba Eulera
ans = 2.7183
octave:42> log(e)
ans = 1
octave:43> i % jednostka urojona
ans = 0 + 1i
octave:44> i**2
ans = -1
octave:45> log(-1)
ans = 0.00000 + 3.14159i
octave:46> sqrt(-1)
ans = 0 + 1i
```

Wbudowane funkcje matematyczne

```
octave:47> log10(100) % logarytm dziesiętny
ans = 2
octave:48> sin(90) % argument w radianach!!!
ans = 0.89400
octave:49> sin(90*pi/180)
ans = 1
octave:50> factorial(5) % silnia
ans = 120
octave:51> factorial(50) % wynik w notacji naukowej:
ans =   3.0414e+64 % czyli 3.0414*10^{64}
octave:52> exp(1)
ans = 2.7183
octave:53> 2*sqrt(9) % w wyrażeniach znak '*' zawsze potrzebny
ans = 6
```

Wbudowane funkcje matematyczne

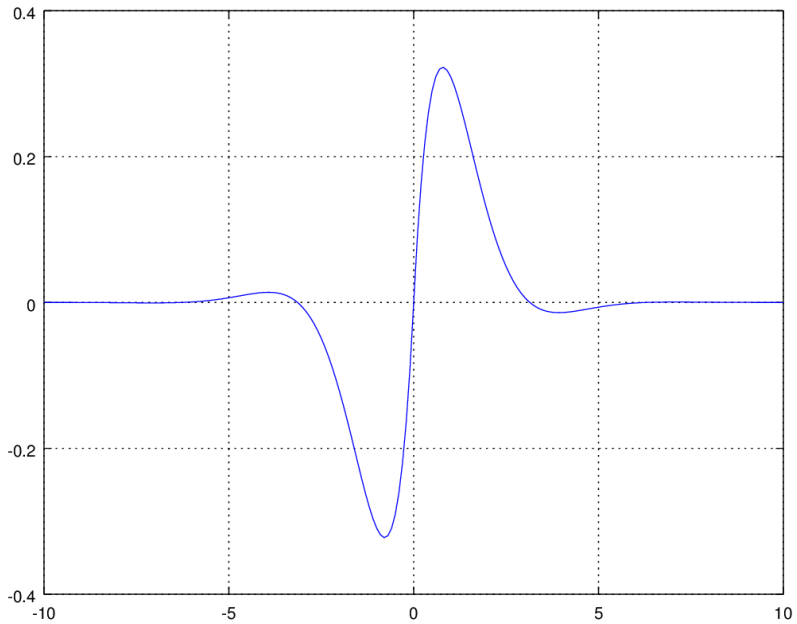
`cos`	kosinus kąta (w radianach)	`asinh`	arcus sinus hiperboliczny
`sin`	sinus kąta	`atan`	arcus tangens
`tan`	tangens	`atan2`	arcus tangens (2 argumenty)
`exp`	funkcja wykładnicza	`atanh`	arcus tangens hiperboliczny
`log`	logarytm naturalny	`abs`	wartość bezwzględna
`log10`	logarytm dziesiętny	`sign`	znak liczby
`sinh`	sinus hiperboliczny	`round`	zaokrąglanie do najbliższej liczby całkowitej
`cosh`	kosinus hiperboliczny	`floor`	zaokrąglanie w dół
`tanh`	tangens hiperboliczny	`ceil`	zaokrąglanie w górę
`acos`	arcus kosinus	`fix`	zaokrąglanie w kierunku zera
`acosh`	arcus kosinus hiperboliczny	`rem`	reszta z dzielenia całkowitego
`asin`	arcus sinus hiperboliczny	`sqrt`	pierwiastek kwadratowy

Zmienne

```
octave:54> deg = pi/180
deg =  0.017453
octave:55> sin(90*deg)
ans =  1
octave:56> new = 3*ans % zmienną ans już znamy
new =  3
octave:57> who
Variables in the current scope:

ans  deg  new
octave:58> x2 = sqrt(3)
x2 =  1.7321
octave:59> x2**2
ans =  3.0000
```

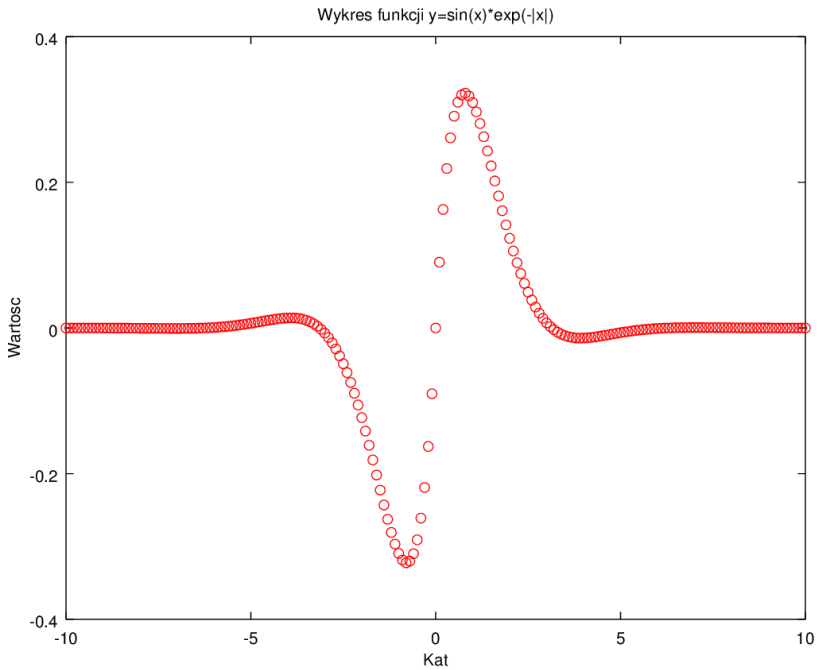
Wykresy funkcji

```
octave:1> x = -10 : 0.1 : 10; % sekwencja argumentów funkcji (wektor)
octave:2> y = sin(x).*exp(-abs(x)); % wektor wartości funkcji
octave:3> plot(x,y)
octave:4> grid
```



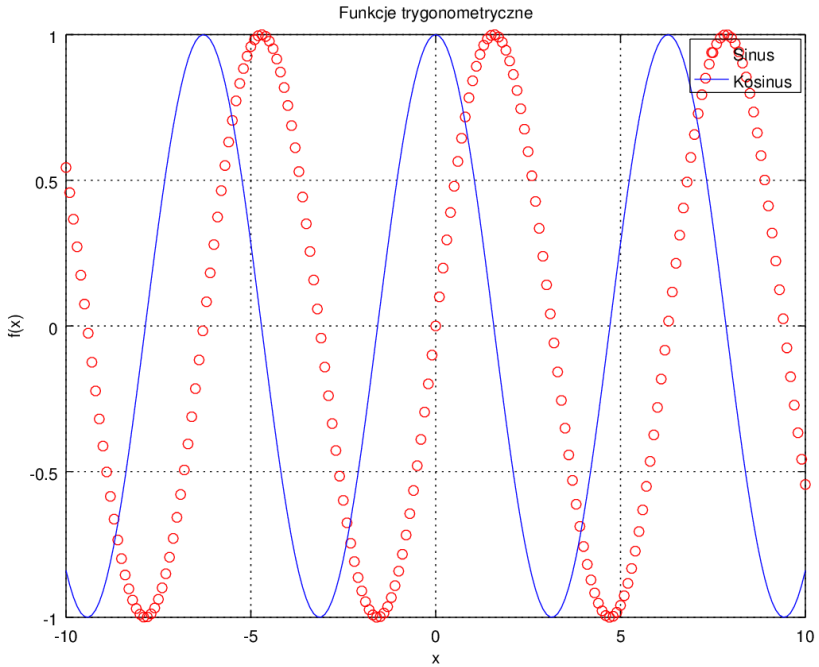
Wykresy funkcji

```
octave:5> plot(x,y,'ro')
octave:6> title('Wykres funkcji y=sin(x)*exp(-|x|)' )
octave:7> xlable('Kąt')
error: 'xlable' undefined near line 1 column 1
octave:7> xlabel('Kąt')
octave:8> ylabel('Wartość')
```



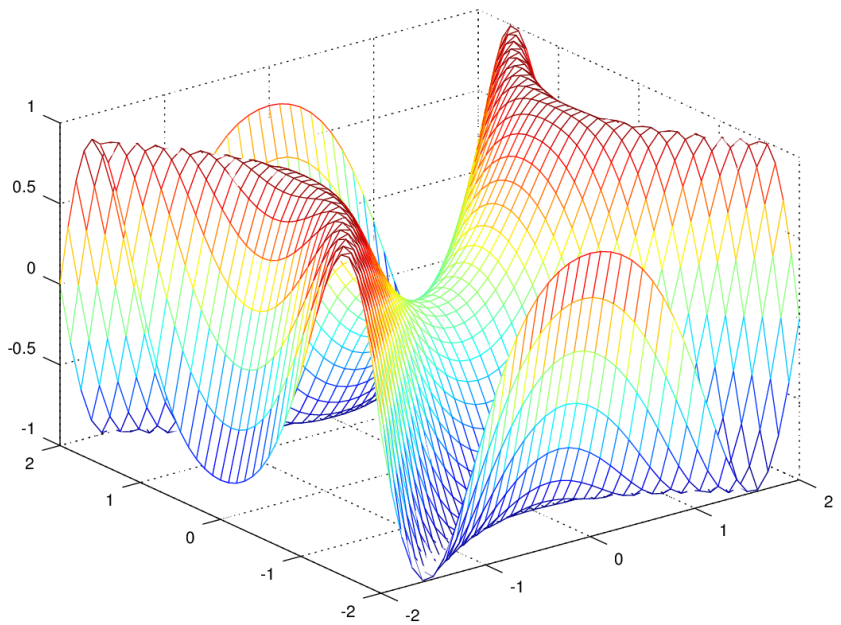
Wykresy funkcji

```
octave:11> plot(x,sin(x),'ro',x,cos(x),'b-')
octave:12> xlabel('x')
octave:13> ylabel('f(x)')
octave:14> title('Funkcje trygonometryczne')
octave:15> grid on
octave:16> legend('Sinus','Kosinus')
```



Wykresy funkcji dwóch zmiennych

```
octave:17> x = -2:0.1:2;
octave:18> [xx,yy] = meshgrid(x,x);
octave:19> z = sin(xx.^2 - yy.^2);
octave:20> mesh(x,y,z)
error: surface: rows (Z) must be the same as length (Y) and columns (Z) must be the same as
      length (X)
error: called from
    surface>__surface__ at line 128 column 11
    surface at line 60 column 19
    mesh at line 76 column 12
octave:20> mesh(x,x,z)
```



Równania nieliniowe

```
octave:21> function y = f(x) % definiujemy funkcję, której pierwiastka szukamy
> y = x.**2 -9;
> endfunction
octave:22> [x, info] = fsolve("f",2.0) % właściwe rozwiązanie
x =  3.0000
info =   8.2960e-08
octave:23> f(x) % sprawdzenie wyniku
ans =   8.2960e-08
octave:25> [x, info] = fsolve("f",0.0) % punkt startowy (czasami) ma znaczenie
warning: division by zero
warning: called from
    fsolve>__dogleg__ at line 587 column 5
    fsolve at line 303 column 11
warning: division by zero
<snip>
warning: division by zero
x = 0
info = -9
octave:28> [x, info] = fsolve("f",-1.0)
x = -3.0000
info =   2.3915e-08
octave:32> x = -4:0.1:4;
octave:33> plot(x,f(x))
```