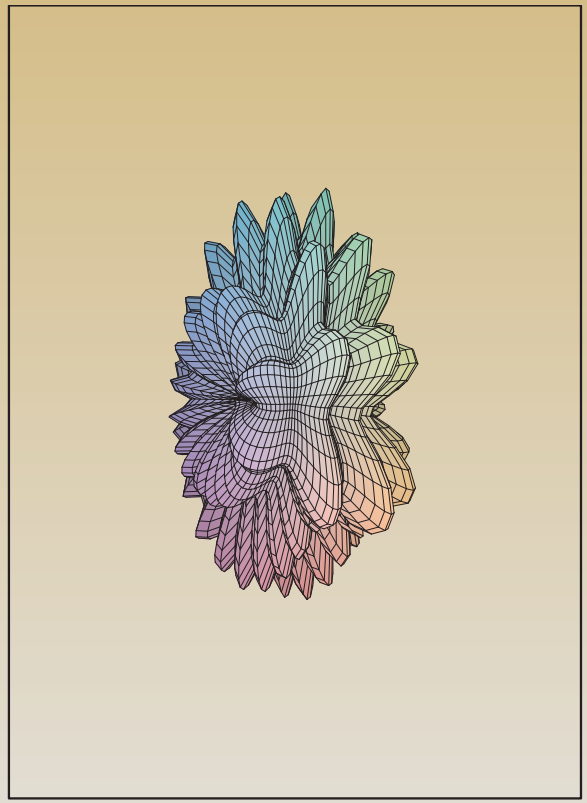
Wykład z Algebry Symbolicznej

dr Andrzej Odrzywołek

pokój 427, IV piętro, środa 11-13

odrzywolek@th.if.uj.edu.pl



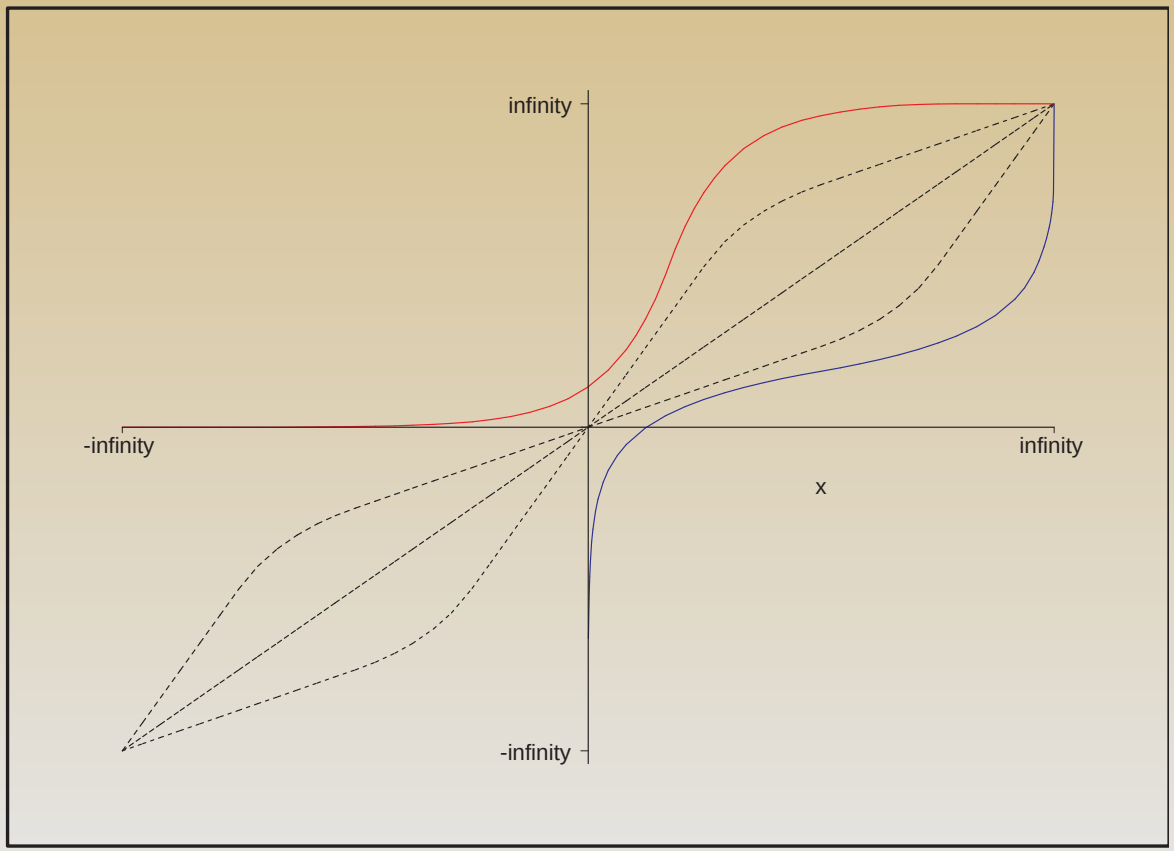
Wykresy 1D

Wykres funkcji $y = f(x)$ w przedziale (a, b): plot(f(x),x=a..b);
Kilka funkcji równocześnie: plot([f1(x),f2(x),f3(x)],x=a..b);
W postaci parametrycznej $x = x(t), y = y(t)$: plot([x(t),y(t),t=t1..t2]);
Podstawowe opcje:

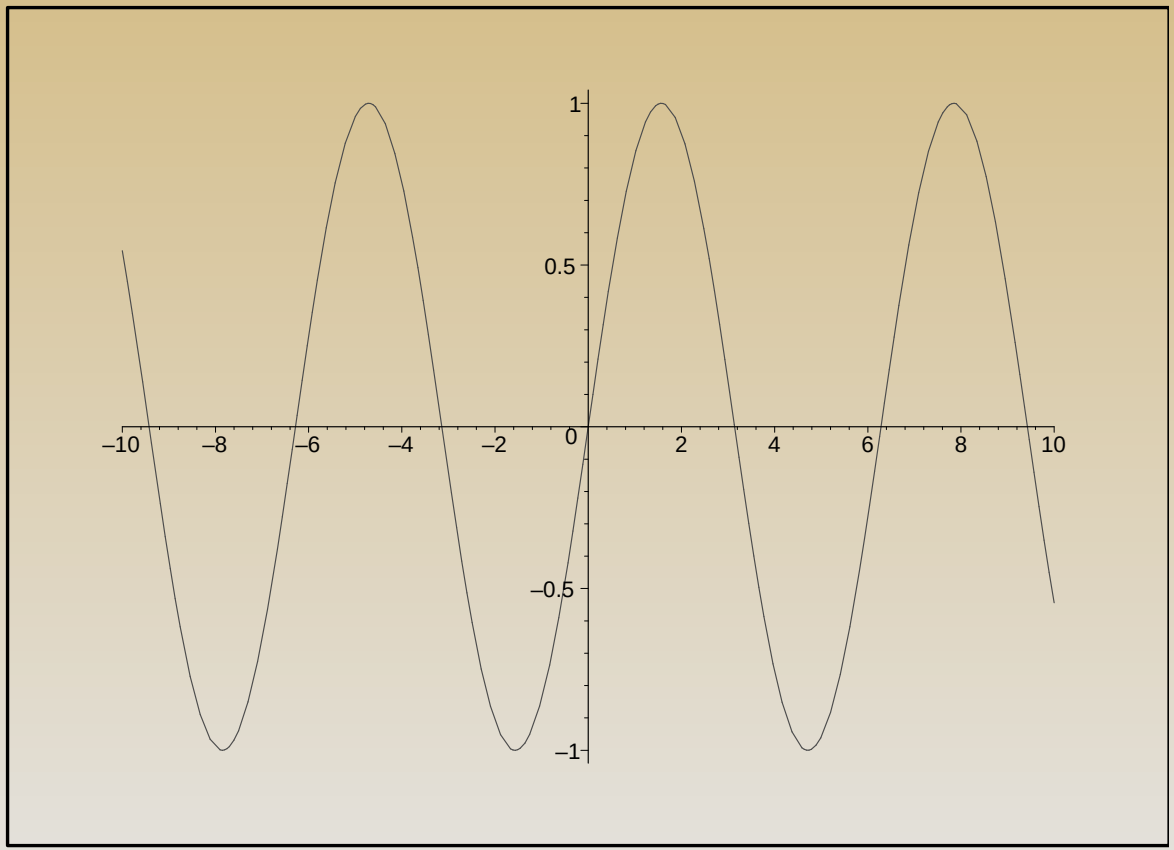
- Zakres na osi pionowej jako trzeci argument: y_min..y_max lub jako jeden z dalszych argumentow jako: view=y_min..y_max
- Kolor linii: color=nazwa_coloru lub color=COLOR(RGB,0.2,0.3,0.9)
- Więcej opcji: > ?plot,options

Wykres funkcji dwóch zmiennych $F(x, y)$:
plot3d(F(x,y),x=-x_min..x_max,y=y_,min..y_max);

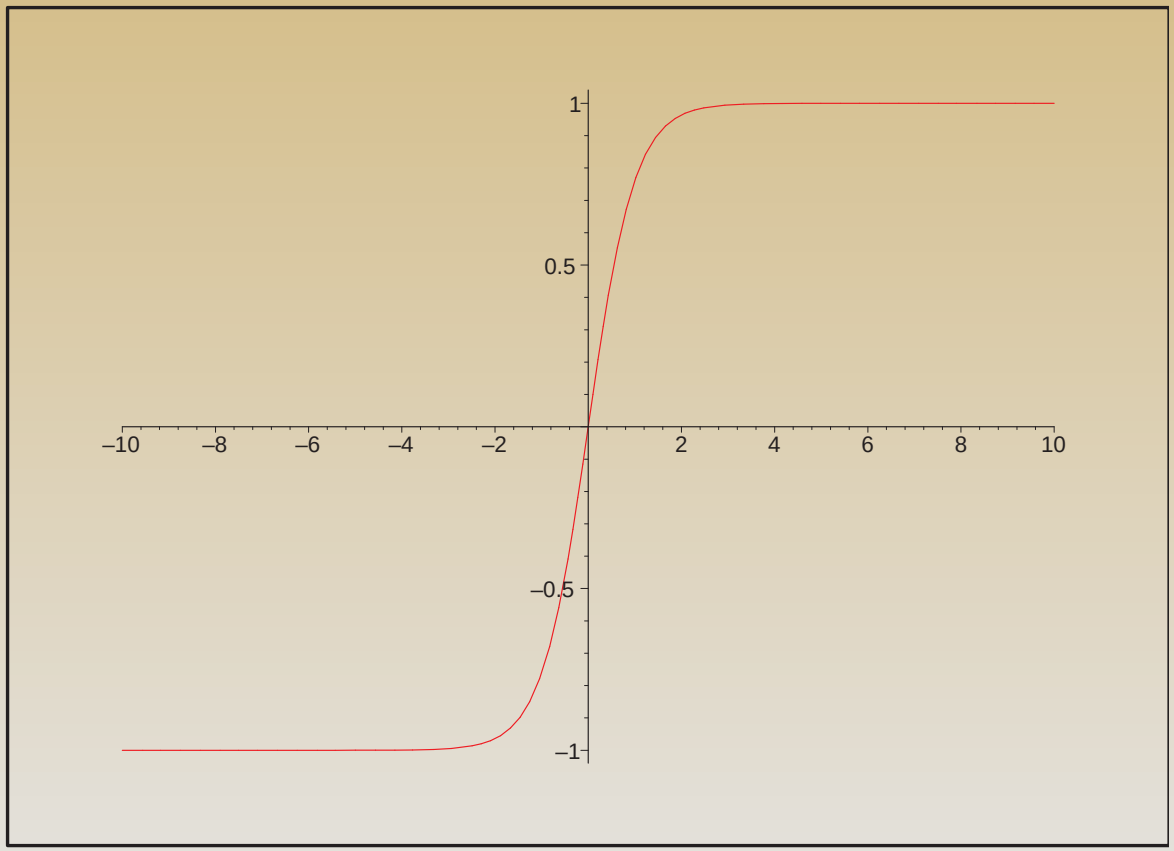
```
plot([exp(x),ln(x),x/2,x,2*x],x=-infinity..infinity,color=[red,blue,black,
black,black], linestyle=[SOLID,SOLID,DOT,DASH,DASHDOT]);
```



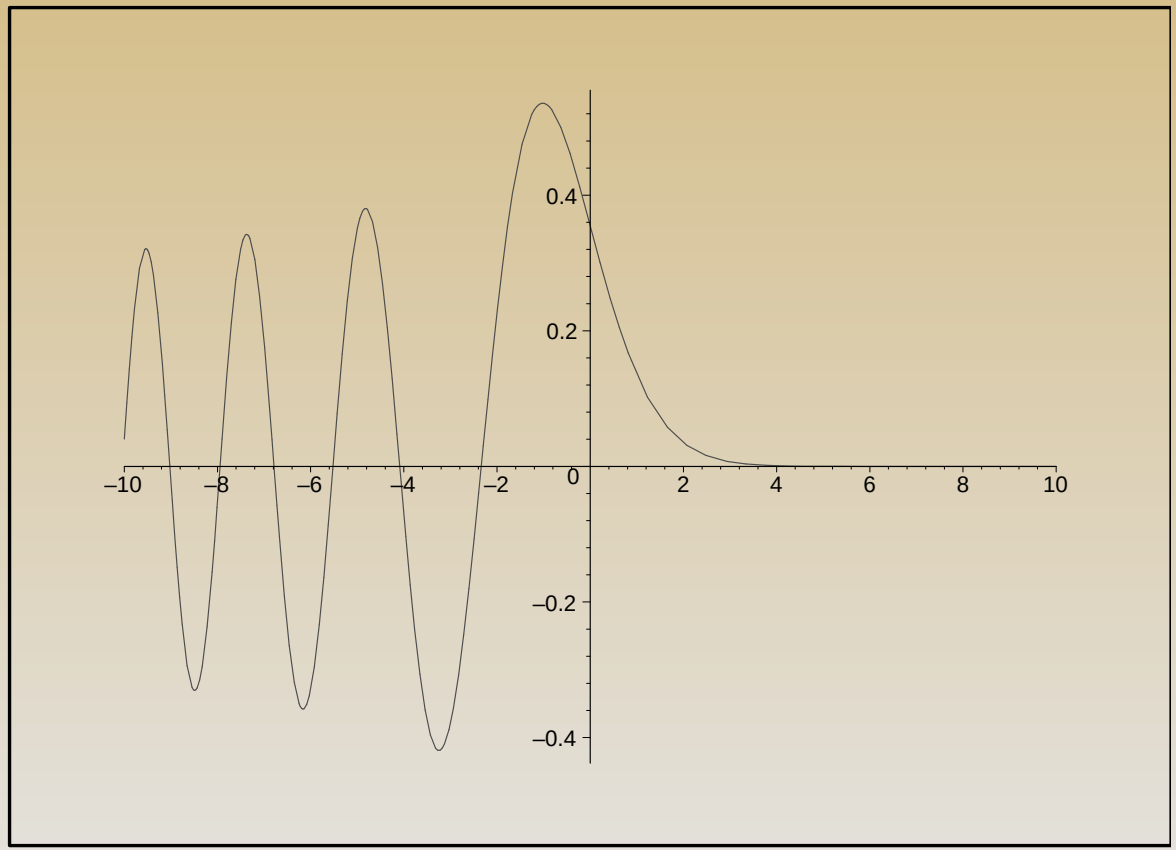
```
plot(sin);
```



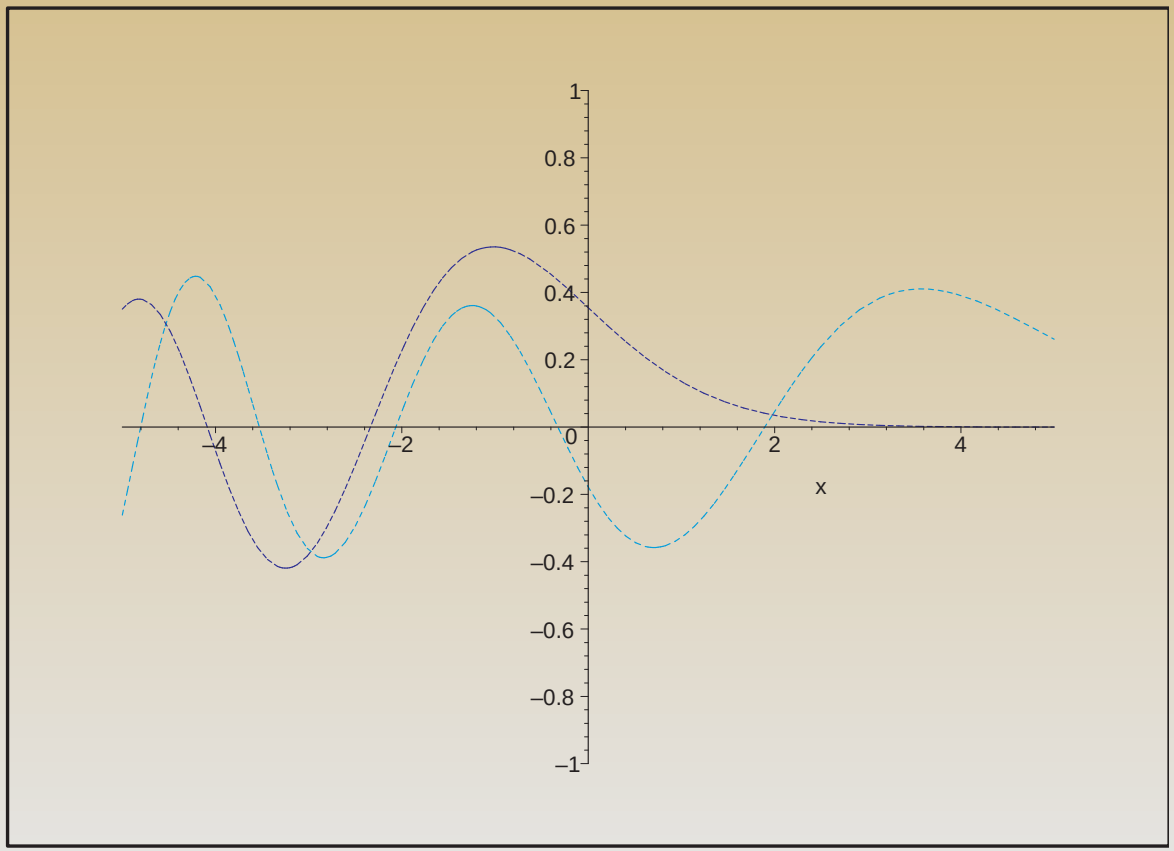
```
plot(tanh);
```



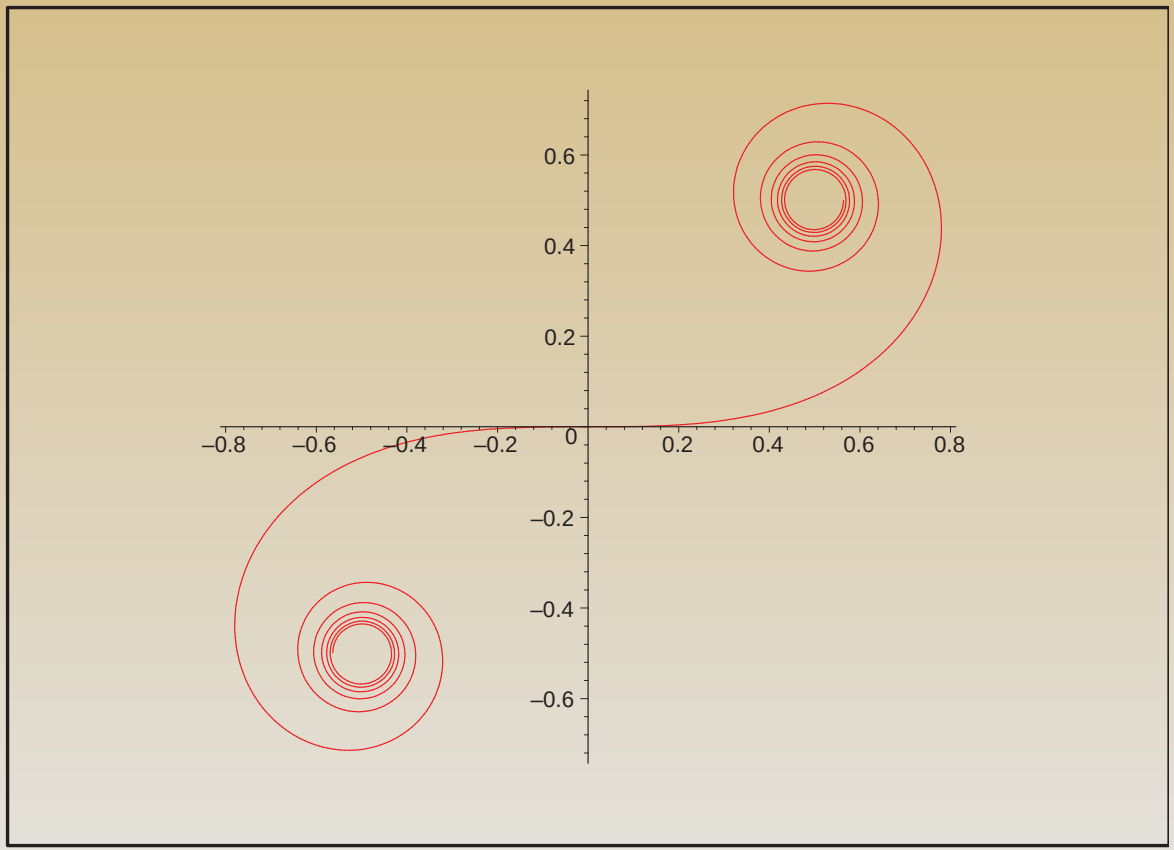
```
plot(AiryAi);
```



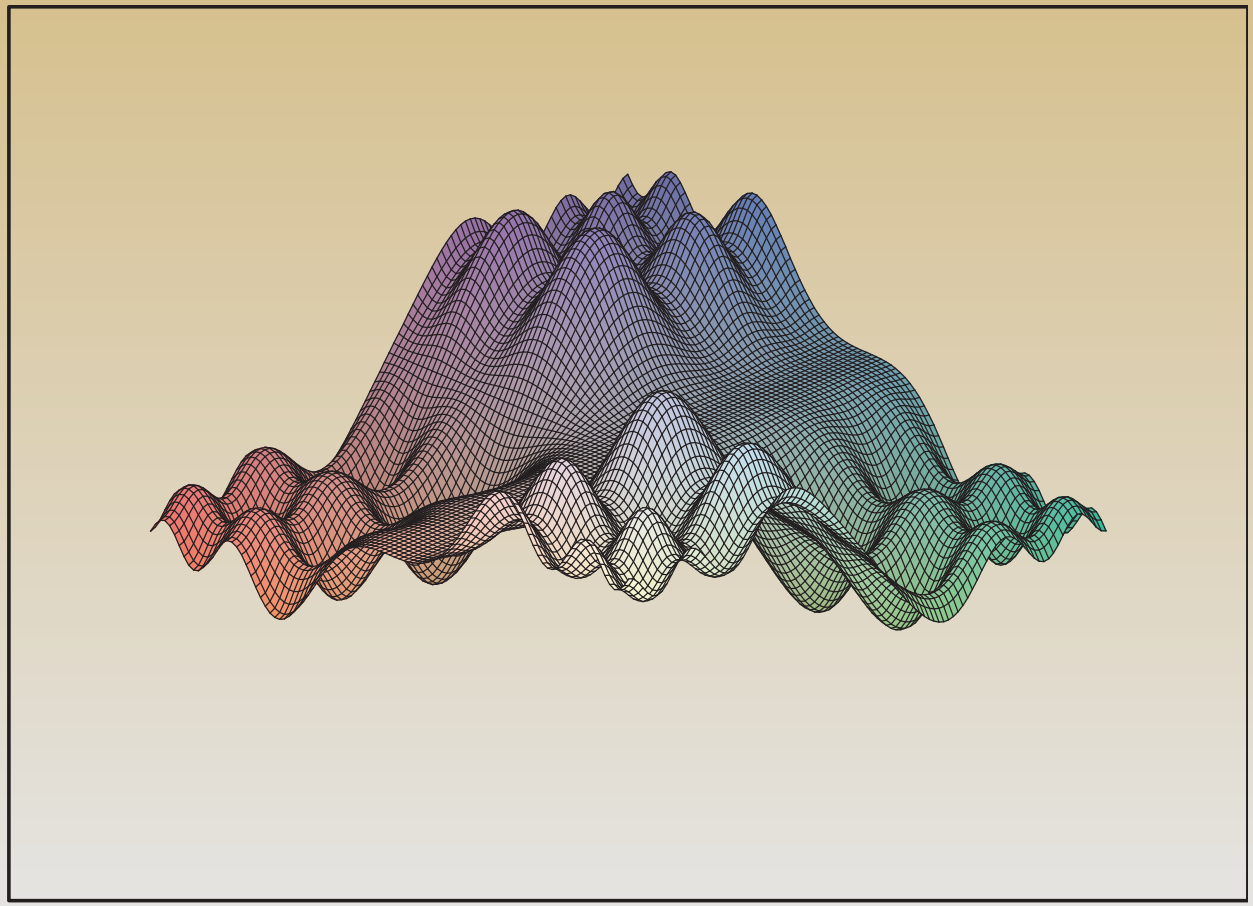
```
plot([AiryAi(x), BesselJ(x,5)], x=-5..5, -1..1,
color=[blue, COLOR(RGB,0.2,0.8,1)], linestyle=[DASH,DOT]);
```



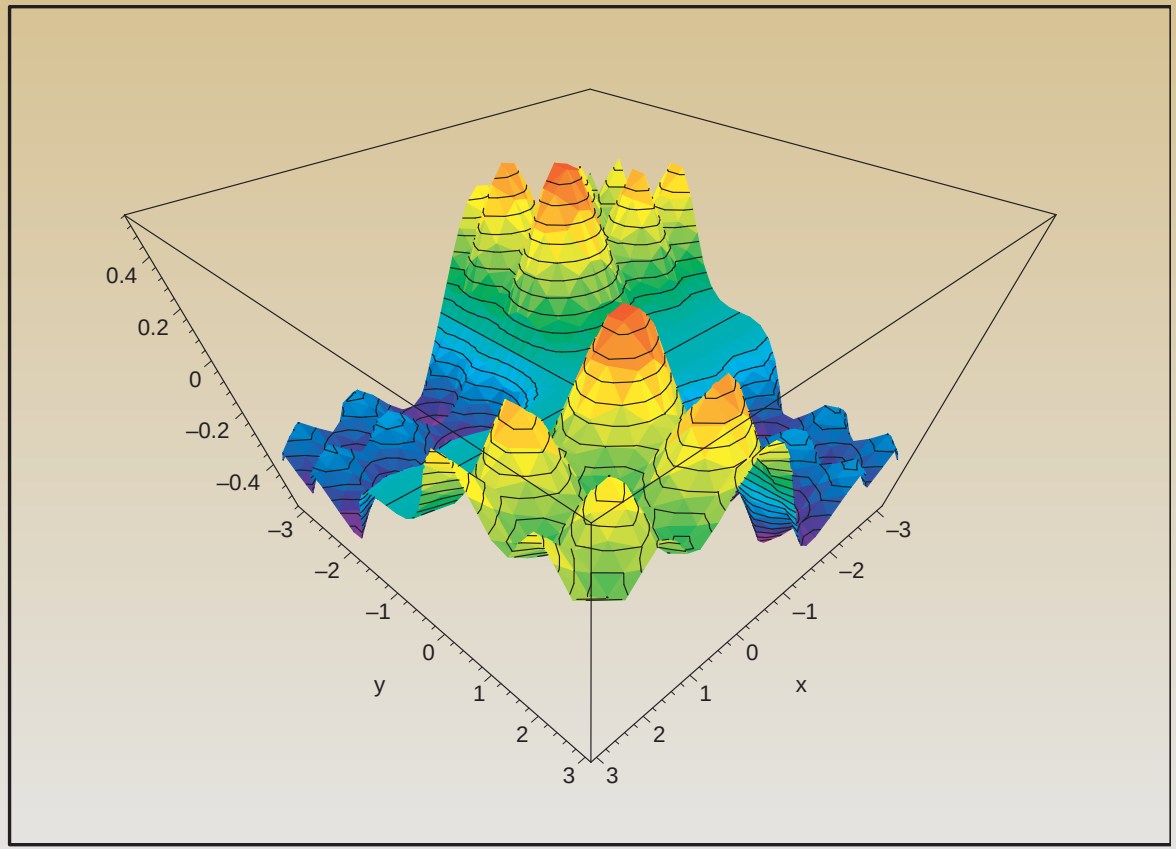
```
plot([FresnelC(t),FresnelS(t),t=-5..5],numpoints=1000);
```



```
plot3d(FresnelC(x)*FresnelS(y),x=-3..3,y=-3..3,numpoints=10000);
```



```
plot3d(FresnelC(x)*FresnelS(y),x=-3..3,y=-3..3,gridstyle=triangular,numpoints=
10000, style=patchcontour,shading=zhue,contours=15,projection=0.1,axes=boxed);
```



Niektóre z mozliwości pakietu plots

Wykresy we współrzędnych biegunowych $r = f(\phi)$:

```
polarplot(f(phi), phi=0..2*Pi, scaling=constrained);
```

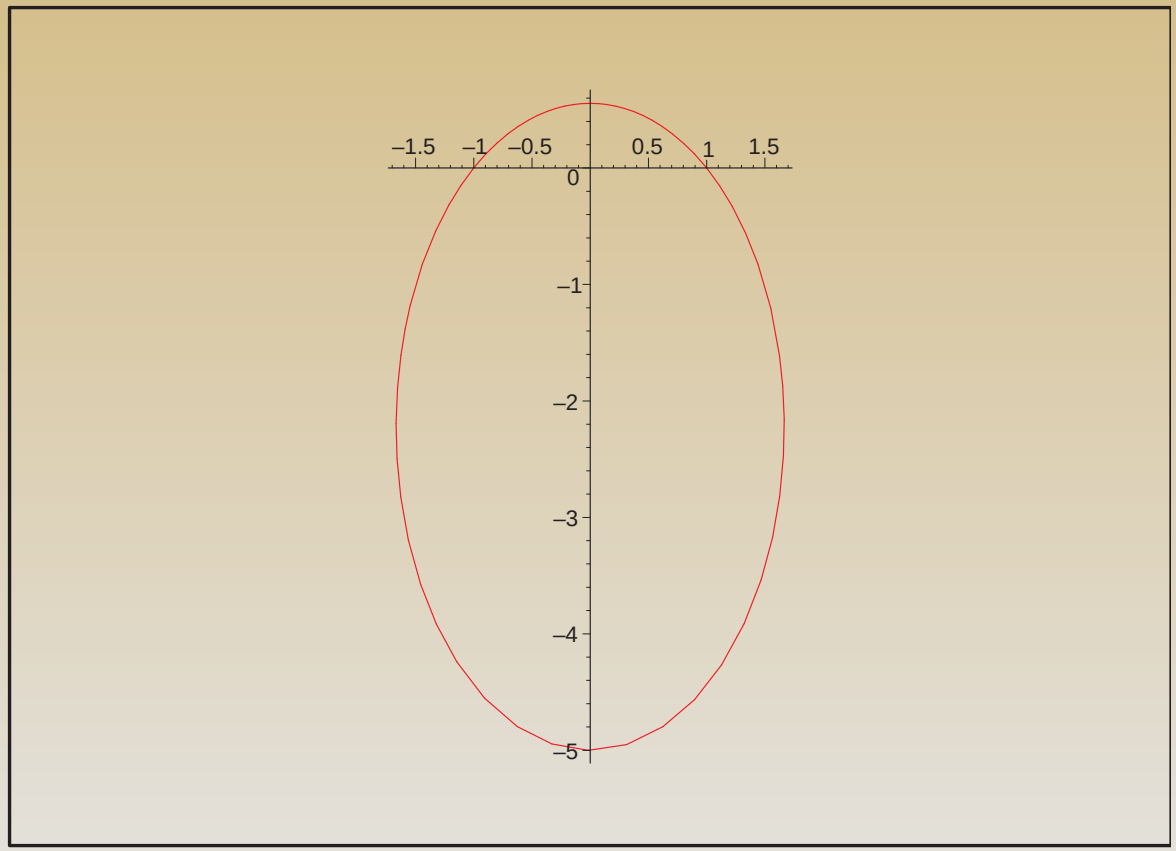
Tzw. wykres w skali logarytmicznej:

```
logplot(f(x), x=x_min..x_max);
```

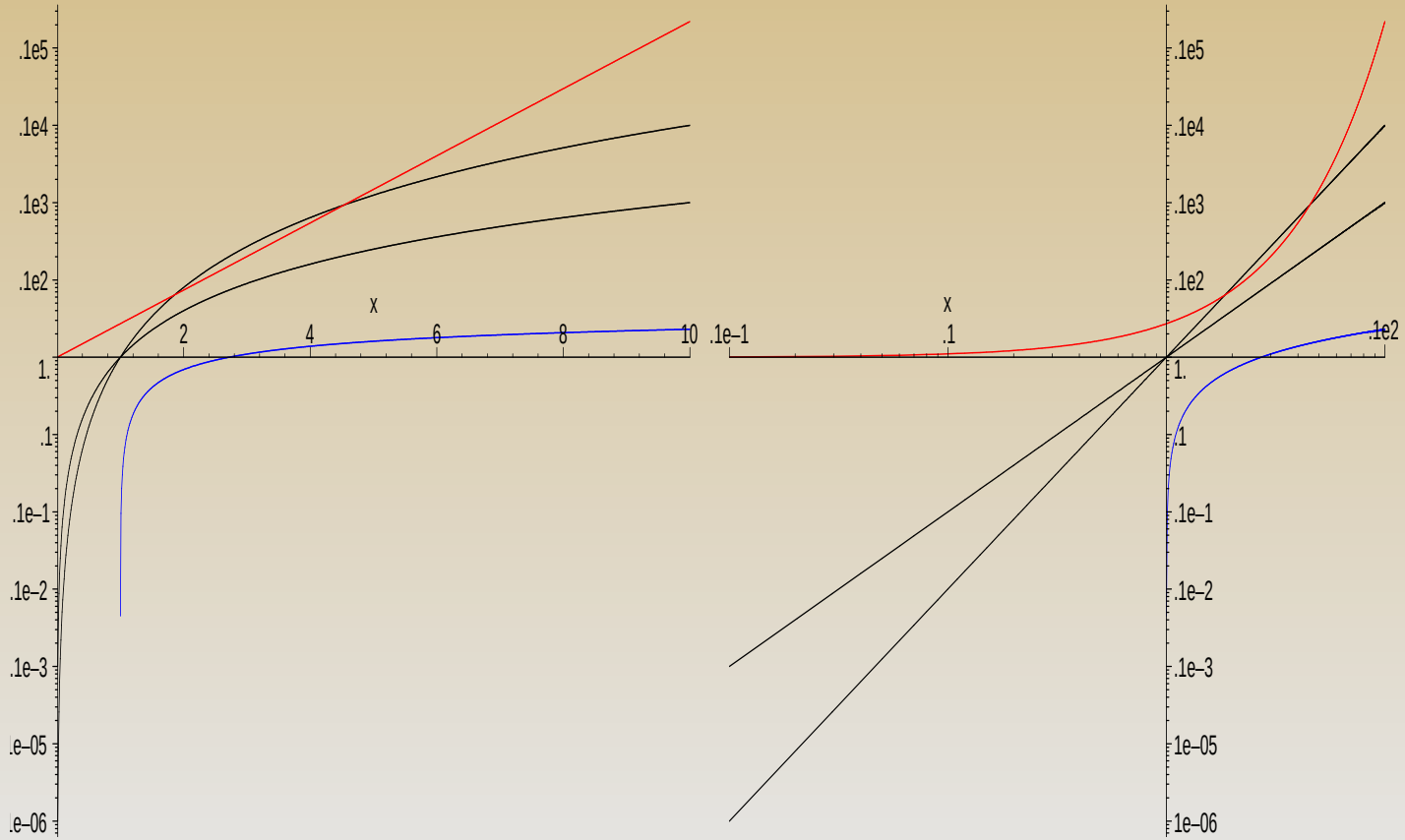
i podwójnie logarytmicznej:

```
loglogplot(f(x), x=x_min..x_max);
```

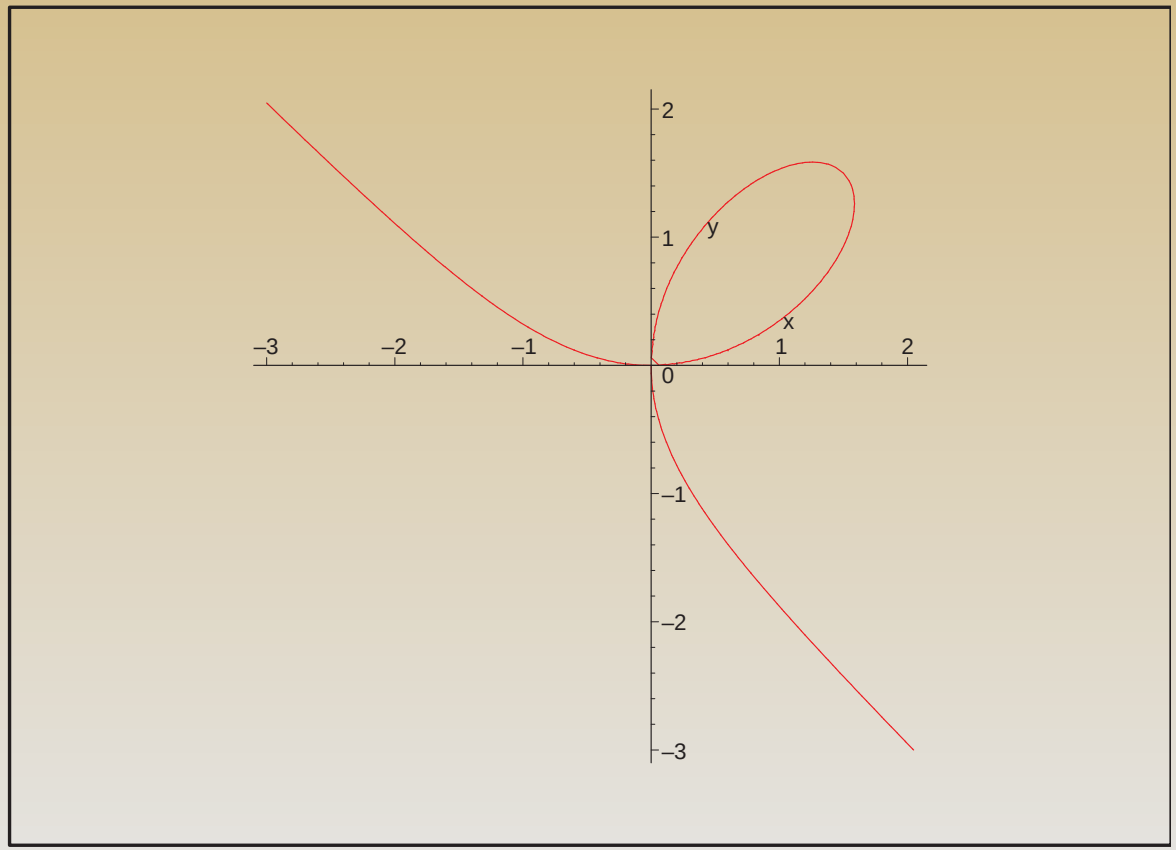
```
polarplot(1/(1+0.8*sin(phi)),phi=0..2*Pi,scaling=constrained);
```



```
(log)logplot([exp(x),log(x),x^2,x^3],x=0.01..10,
color=[red,blue,black,black],numpoints=1000);
```



```
implicitplot(x^3+y^3=3*x*y,x=-3..3,y=-3..3,numpoints=10000,scaling=constrained);
```



Przekształcenia konforemne

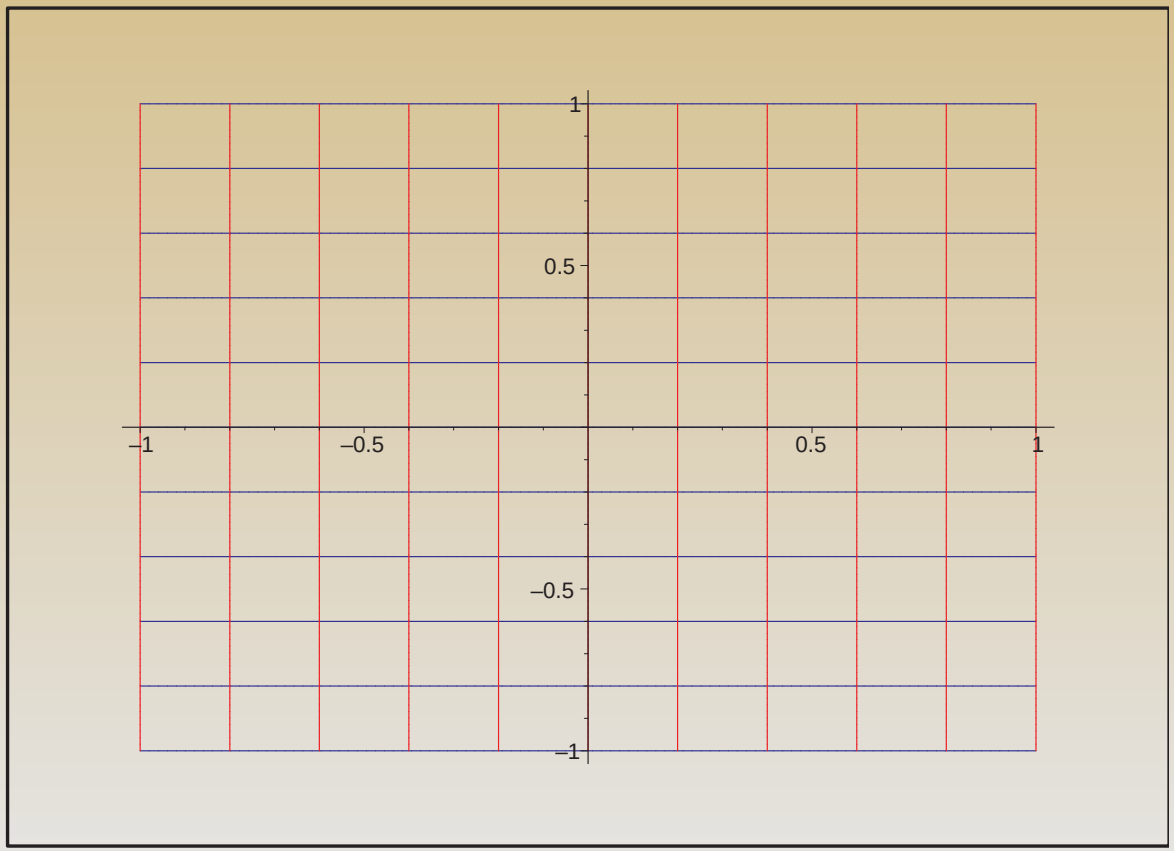
Analityczna funkcja zespolona $f(z)$ definiuje przekształcenie konforemne płaszczyzny zespolonej.

Następująca funkcja wykreśla przekształcenie siatki prostokątnej (kwadratu o rogach z_min, z_max):

```
conformal(f(z),z=z_min..z_max);)
```

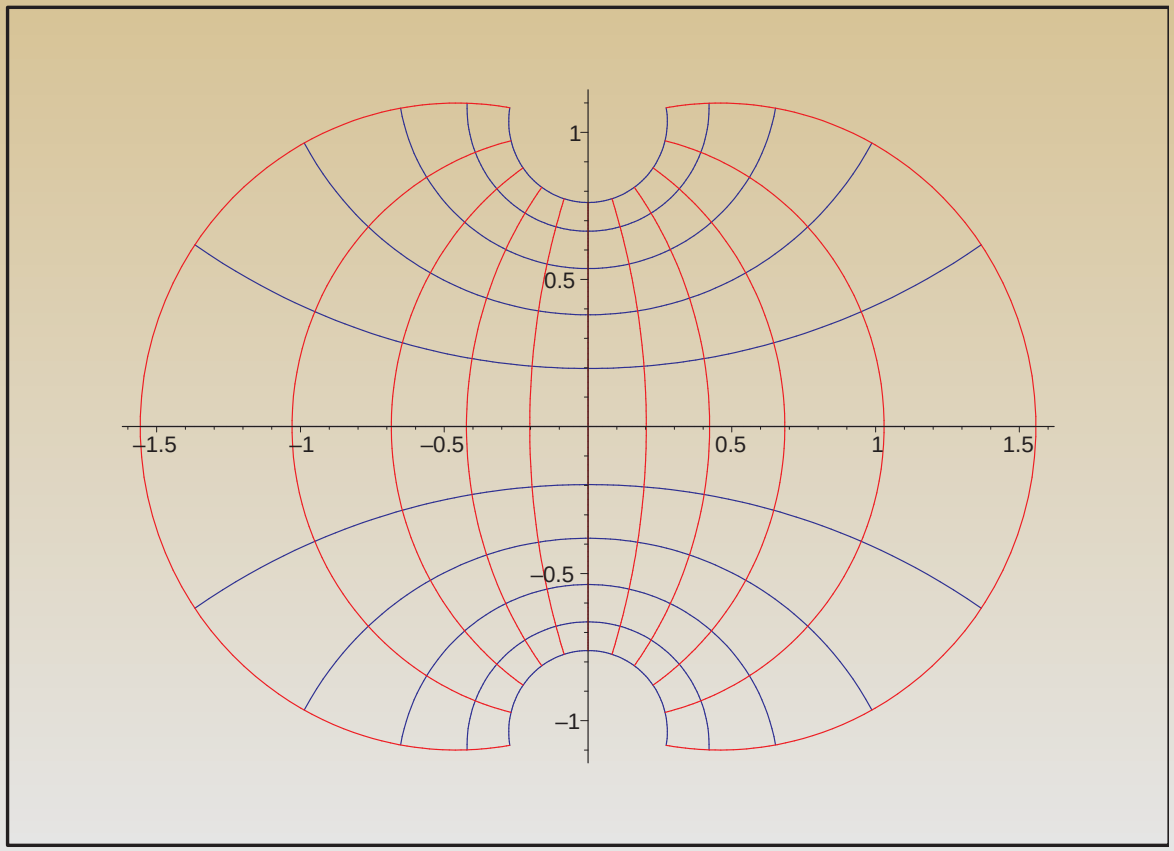
$f(z) = z$

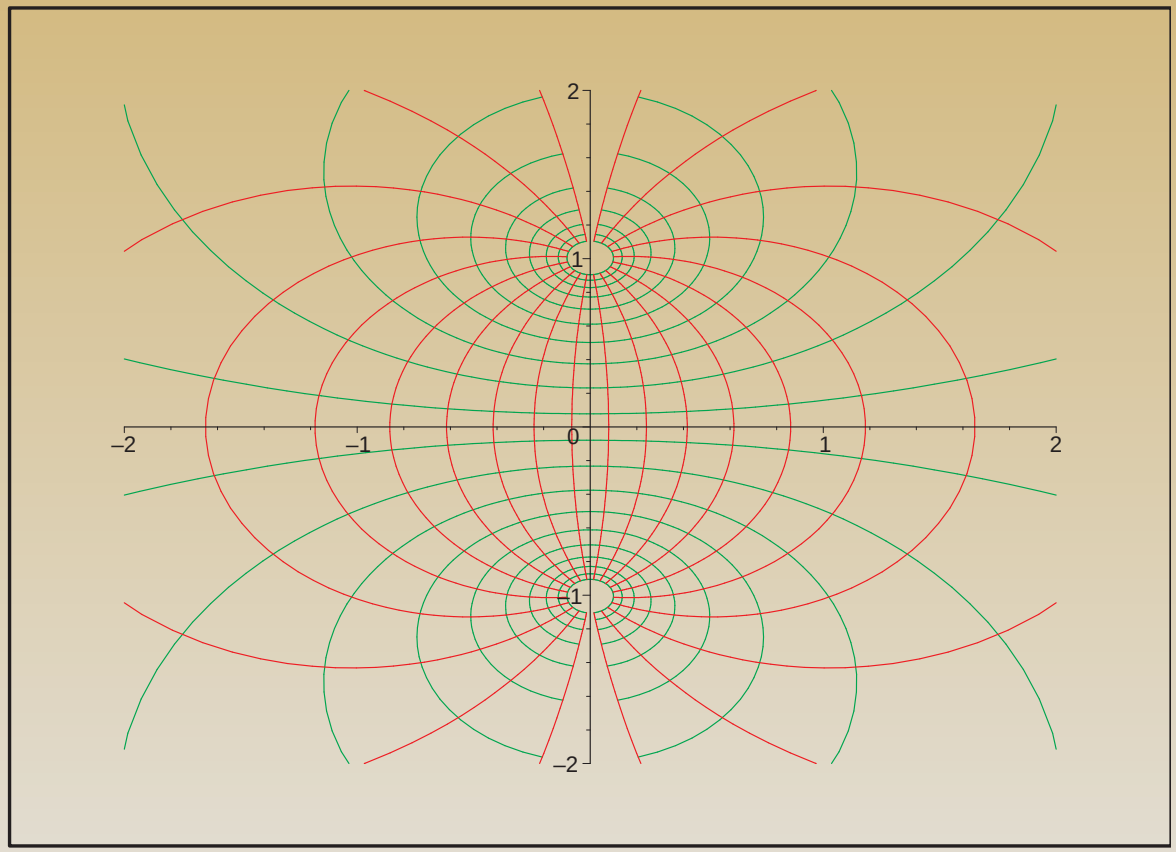
```
conformal(tan(z),z=-1-I..1+I,numxy=[100,100],color=[red,blue]);
```



$f(z) = \tan(z)$

```
conformal(tan(z),z=-1.5*(1+I)..1.5*(1+I),-2-2*I..2+2*I,
grid=[20,20],numxy=[100,100]);
```




2
1
0
−1
−2
−2
−1
1
2

Czy funkcja $\tan(z)$ przyjmuje wartość $\pm i$?

```
solve(tan(z)=I,z);
```

Brak wyniku.

```
> fsolve(tan(z)=I,z);
```

$$\text{-7.207846668+186.3505898*I}$$

```
> evalf(subs(z=%,tan(z)));
```

$$\text{-.2641777612}\cdot 10^{-161}\text{+1.*I}$$

Uwaga: Równanie $\tan z = \pm i$ nie posiada *żadnego* rozwiązania. (*Jak to pokazać?*)

Zastosowanie animacji

Najprostsze przekształcenia konforemne:

1. $f(z) = z \longrightarrow$ przekształcenie tożsamościowe
2. $f(z) = z + c \longrightarrow$ przesunięcie o wektor $c = a + bi$
3. $f(z) = e^{i\phi} z \longrightarrow$ obrót o kąt ϕ
4. $f(z) = kz \longrightarrow$ podobieństwo w skali k

Idealnie to zobaczenia tego nadaje się animacja.

Schemat jest następujący:

Jeżeli pojedynczy wykres rysujemy jako np:

```
> plot_instr(f(x), zakres, parametry);
```

to animacja w ktorej zmienia się parametr t jest generowana przez:

```
> animate(plot_instr, [f(x),zakres,parametry] , t=t_min..t_max,opcje_animacji);
```

Np: kolejne sekwencje przekształceń konforemnych z zostały wygenerowane jako:

```
animate(conformal,
[ t*z, z=0..1+I,-2-2*I..2+2*I,
grid=[10,10],numxy=[20,20] ],
t=1..2,scaling=constrained);


animate(conformal,
[2*z-(1+I)*t,z=0..1+I,-2-2*I..2+2*I,
grid=[10,10],numxy=[20,20]],
t=0..1,scaling=constrained);


animate(conformal,
[(2*z-(1+I)*1)*exp(I*t),z=0..1+I,-2-2*I..2+2*I,
grid=[10,10],numxy=[20,20]],
t=0..Pi/2,scaling=constrained);
```

Obrót krzywej płaskiej

Jak widzimy mnożenie przez liczbę $e^{i\phi}$m gdzie ϕ jest rzeczywiste, odpowiada obrotowi płaskiemu o kąt ϕ.

PROBLEM: narysować wykres funkcji $y = f(x)$ obróconej o kąt alpha.

1. Funkcję $y = f(x)$ przedstawiamy w postaci zespolonej:

$$z = x + i\ f(x)$$

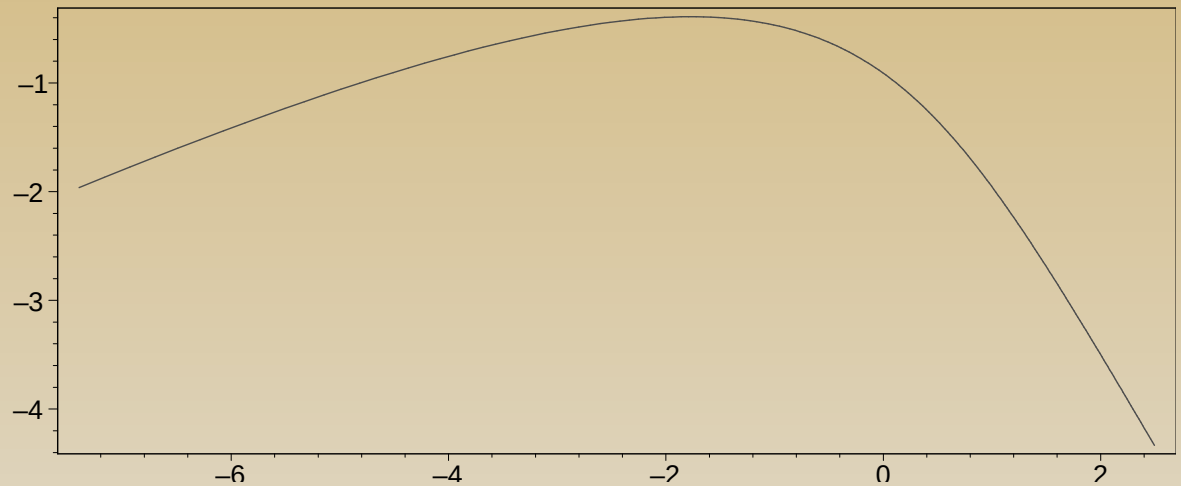
2. Mnożymy z przez $e^{i\phi}$

3. Obliczamy $X(t) = Re(ze^{i\phi}), Y(t) = Im(ze^{i\phi})$

4. Rysujemy wykres funkcji danej w postaci parametrycznej:

$$x = X(t), \quad y = Y(t)$$

Oto przykładowy wynik obrócenia funkcji $y = e^x$ o kąt $\phi = \pi/3$:



`Czy zostaliśmy przekonani, że wyżej opisana procedura rzeczywiście daje obrót??`

Tylko animacja rozwiewa wątpliwości ostatecznie...

Formaty graficzne

Maple pozwala na użycie kilku sposobów wyświetlania wykresów oraz zapisu plików graficznych w różnych formatach.

Do zmiany standardowych ustawień, czyli wykresów wyświelanych *inline* służy funkcja plotsetup.

- plotsetup(inline);
  .– ustawienie domyślne
- plotsetup(window);
  — wykres w osobnym oknie.
- plotsetup(x11);
  — tylko Linux (UNIX): osobny proces zarządzający danym wykresem.
- plotsetup(ps, plotoptions=‘colour=rgb,width=4in,height=3in, noborder‘, plotoutput=‘nazwa_pliku.ps‘)); – postscript.
- plotsetup(jpeg, plotoptions=”height=600,width=800”, plotoutput=‘nazwa_pliku.jpg‘); — zapisanie wykresu do pliku jpg.

- plotsetup(gif, plotoptions="height=600,width=800", plotoutput=‘nazwa_pliku.gif‘);.— GIF w tym animowane
- plotsetup(bmp, plotoptions="height=600,width=800", plotoutput=‘nazwa_pliku.bmp‘);. — bitmapa
- plotsetup(pov, plotoutput=‘nazwa_pliku.pov‘); — eksport do formatu POVRay-a, tylko wykresy 3D.

Generowanie sekwencji plików graficznych

Ze względu na skromne możliwości eksportu animacji (tylko GIF), przydatna jest umiejętność zapisywania sekwencji pojedynczych „klatek”:

```
for i from 1 to 100 do
plotsetup(jpeg,
plotoptions="height=600,width=800",
plotoutput=cat('movie/frame',1000+i,'.jpg'));
conformal((i/50.0+0.5)*z*exp(I*2*Pi*i/10)+i/100.0*exp(I*2*Pi*i/100), z=-1-I..1+I,-
3-3*I..3+3*I,grid=[10,10],numxy=[100,100]);
#print(i);
od;
```

Klatki te można następnie przetworzyć do dowolnego formatu np: AVI skompresowane przez MPEG-4 przez linuksowy program `MPlayer` (www.mplayerhq.hu):

```
$ mencoder "mf://*.jpg" -mf fps=15 -o movie.avi -ovc lavc -lavcopts vcodec=mpeg4
```

Analogiczne narzędzia są dostępne dla Windows np: RAD Video Tools (www.radgametools.com)