Sisteme de ecuatii diferentiale

Definirea si rezolvarea sistemelor de ecuatii diferentiale

In [1]:

```
reset()
```

Se considera sistemul de ecuatii diferentiale:

$$\begin{cases} x'(t) = & x(t) + y(t) \\ y'(t) = & x(t) - y(t) \end{cases}$$

Definirea sistemului de ecuatii diferentiale se face in mod similar cu cel din cazul ecuatiilor diferentiale, definindu-se pe rand fiecare dintre ecuatii:

In [2]:

```
t = var('t')
x=function('x')(t)
y=function('y')(t)
deq1=diff(x,t)==x+y
deq2=diff(y,t)==x-y
syst=[deq1,deq2]
syst
```

Out[2]:

```
[diff(x(t), t) == x(t) + y(t), diff(y(t), t) == x(t) - y(t)]
```

```
Comanda de rezolvare a sistemelor de ecuatii diferentiale este:

desolve_system(des, vars, ics=None, ivar=None)

INPUT:
des : lista ecuatiilor diferentiale

vars : lista variabilelor dependente

ics : (optional) lista conditiilor initiale pentru ivar si vars. Aceasta furnizeaza
valorile functiilor necunoscute intr-un punct si trebuie data ca o list de forma [x0,
y1(x0), y2(x0), ...]

ivar : (optional) variabila independenta. Aceasta trebuie specificata atunci cand in
expresia sistemului apar si alti parametrii.
```

Solutia generala se obtine apeland comanda *desolve_system*

In [3]:

```
desolve_system(syst, [x,y])
```

Out[3]:

```
[x(t) == 1/2*sqrt(2)*(x(0) + y(0))*sinh(sqrt(2)*t) + cosh(sqrt(2)*t)*x(0),
 y(t) == 1/2*sqrt(2)*(x(0) - y(0))*sinh(sqrt(2)*t) + cosh(sqrt(2)*t)*y(0)]
```

In [4]:

```
sol_syst=desolve_system(syst, [x,y])
show(sol_syst)
```

$$\left[x\left(t\right) = \frac{1}{2}\,\sqrt{2}(x\left(0\right) + y\left(0\right))\,\sinh\left(\sqrt{2}t\right) + \cosh\left(\sqrt{2}t\right)x\left(0\right), y\left(t\right) = \frac{1}{2}\,\sqrt{2}(x\left(0\right) - y\left(0\right))\,\sin\right.$$

Se observa ca solutia generala a sistemului este returnata in termenii valorilor $x(0)$ si $y(0)$, daca dorim ca solutia sa apara in termenii a doua constante de integrare vom inlocui $x(0) = C1$, $y(0) = C2$ rezolvand problema Cauchy corespunzatoare:

In [5]:

```
C1,C2=var('C1,C2')
desolve_system(syst, [x,y],[0,C1,C2])
```

Out[5]:

```
[x(t) == 1/2*sqrt(2)*(C1 + C2)*sinh(sqrt(2)*t) + C1*cosh(sqrt(2)*t),
 y(t) == 1/2*sqrt(2)*(C1 - C2)*sinh(sqrt(2)*t) + C2*cosh(sqrt(2)*t)]
```

In [6]:

```
sol_syst=desolve_system(syst, [x,y],[0,C1,C2])
show(sol_syst)
```

$$\left[x\left(t\right) = \frac{1}{2}\,\sqrt{2}(C_1 + C_2)\,\sinh\left(\sqrt{2}t\right) + C_1\,\cosh\left(\sqrt{2}t\right), y\left(t\right) = \frac{1}{2}\,\sqrt{2}(C_1 - C_2)\,\sinh\left(\sqrt{2}t\right)\right.$$

Reprezentarea grafica a solutiilor

Daca se doreste vizualizarea graficului unor solutii ale sistemului atunci trebuie asignate valori numerice constantelor de integrare. Mai intai construim solutiile ca functii de variabile $t, C1, C2$.

In [7]:

```
sol=desolve_system(syst, [x,y],[0,C1,C2])
sol_x(t,C1,C2)=sol[0].rhs()
sol_x
```

Out[7]:

```
(t, C1, C2) |--> 1/2*sqrt(2)*(C1 + C2)*sinh(sqrt(2)*t) + C1*cosh(sqrt(2)*t)
```

In [8]:

```
show(sol_x)
```

$$(t, C_1, C_2) \mapsto \frac{1}{2}\sqrt{2}(C_1 + C_2)\sinh\left(\sqrt{2}t\right) + C_1\cosh\left(\sqrt{2}t\right)$$

In [9]:

```
sol_y(t,C1,C2)=sol[1].rhs()
sol_y
```

Out[9]:

```
(t, C1, C2) |--> 1/2*sqrt(2)*(C1 - C2)*sinh(sqrt(2)*t) + C2*cosh(sqrt(2)*t)
```

In [10]:

```
show(sol_y)
```

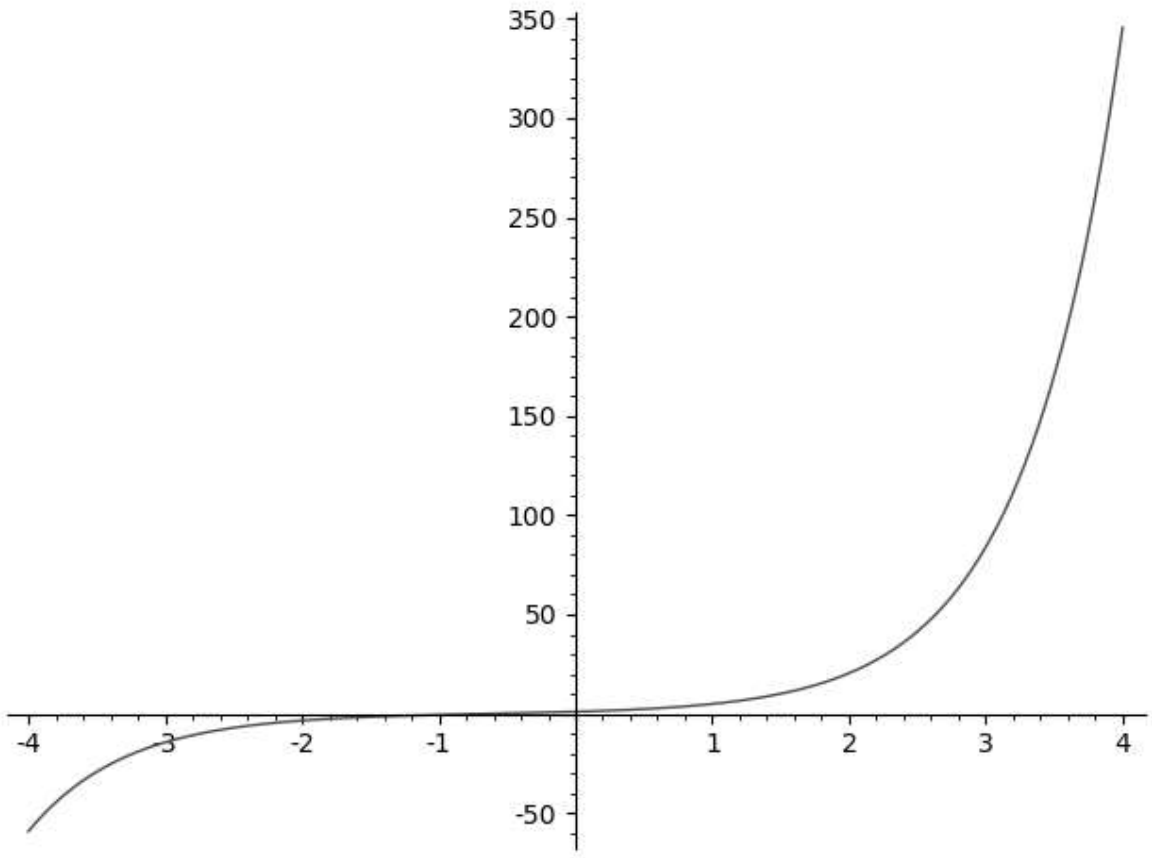
$$(t, C_1, C_2) \mapsto \frac{1}{2}\sqrt{2}(C_1 - C_2)\sinh\left(\sqrt{2}t\right) + C_2\cosh\left(\sqrt{2}t\right)$$

Graficul solutiilor in cazul $C1 = 1$ si $C2 = 1$ se obtine astfel:

In [11]:

```
plot(sol_x(t,1,1),t,-4,4,color='red')
```

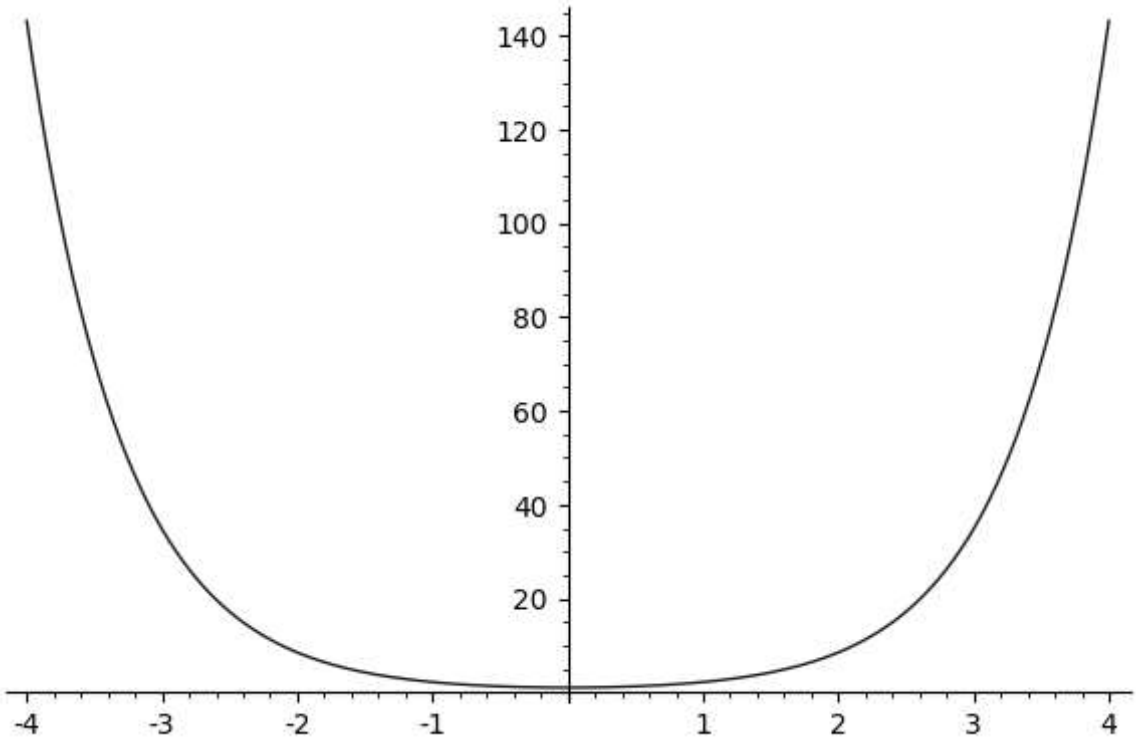
Out[11]:



In [12]:

```
plot(sol_y(t,1,1),t,-4,4,color='blue')
```

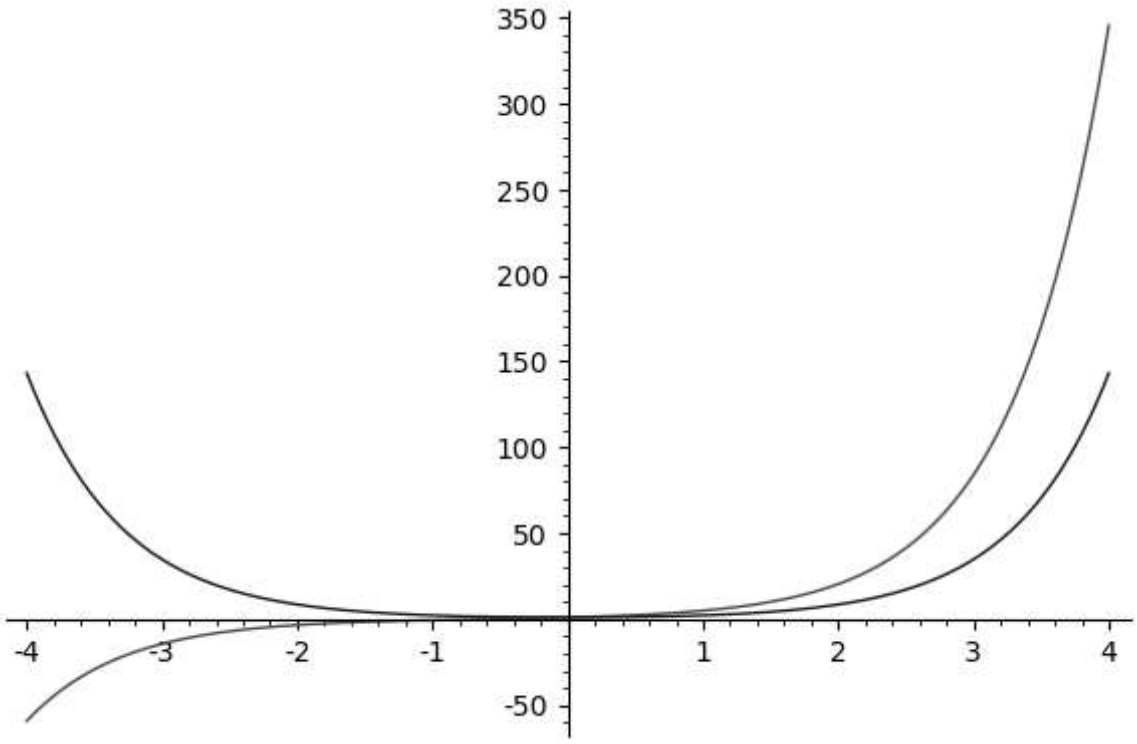
Out[12]:



sau daca dorim reprezentarea ambelor solutii in aceeasi fereastra folosim:

In [13]:

```
g1=plot(sol_x(t,1,1),t,-4,4,color='red')
g2=plot(sol_y(t,1,1),t,-4,4,color='blue')
show(g1+g2)
```



Problema Cauchy pentru sisteme de ecuatii diferentiale

Pentru obtinerea solutiei unei probleme Cauchy (problema ca valori initiale) atasate unui sistem vom proceda in felul urmator. Sa consideram problema Cauchy:

$$\begin{cases} x'(t) = & x(t) + y(t) \\ y'(t) = & x(t) - y(t) \\ x(0) = & 1 \\ y(0) = & 0 \end{cases}$$

In [14]:

```
t = var('t')
x=function('x')(t)
y=function('y')(t)
deq1=diff(x,t)==x+y
deq2=diff(y,t)==x-y
syst=[deq1,deq2]
vars=[x,y]
in_cond=[0,1,0]
```

In [15]:

```
sol=desolve_system(syst, vars,in_cond)
sol
```

Out[15]:

```
[x(t) == 1/2*sqrt(2)*sinh(sqrt(2)*t) + cosh(sqrt(2)*t),
 y(t) == 1/2*sqrt(2)*sinh(sqrt(2)*t)]
```

In [16]:

```
show(sol)
```

$$\left[x\left(t\right)=\frac{1}{2}\,\sqrt{2}\sinh\left(\sqrt{2}t\right)+\cosh\left(\sqrt{2}t\right),y\left(t\right)=\frac{1}{2}\,\sqrt{2}\sinh\left(\sqrt{2}t\right)\right]$$

Pentru reprezentarea grafica a solutiei problemei Cauchy, mai intai construim cele componente ale solutiei ca functii de variabila t si apoi vom folosi comanda *plot*:

In [17]:

```
sol_x(t)=sol[0].rhs()
sol_x
```

Out[17]:

```
t |--> 1/2*sqrt(2)*sinh(sqrt(2)*t) + cosh(sqrt(2)*t)
```

In [18]:

```
show(sol_x)
```

$$t\mapsto\frac{1}{2}\,\sqrt{2}\sinh\left(\sqrt{2}t\right)+\cosh\left(\sqrt{2}t\right)$$

In [19]:

```
sol_y(t)=sol[1].rhs()
sol_y
```

Out[19]:

```
t |--> 1/2*sqrt(2)*sinh(sqrt(2)*t)
```

In [20]:

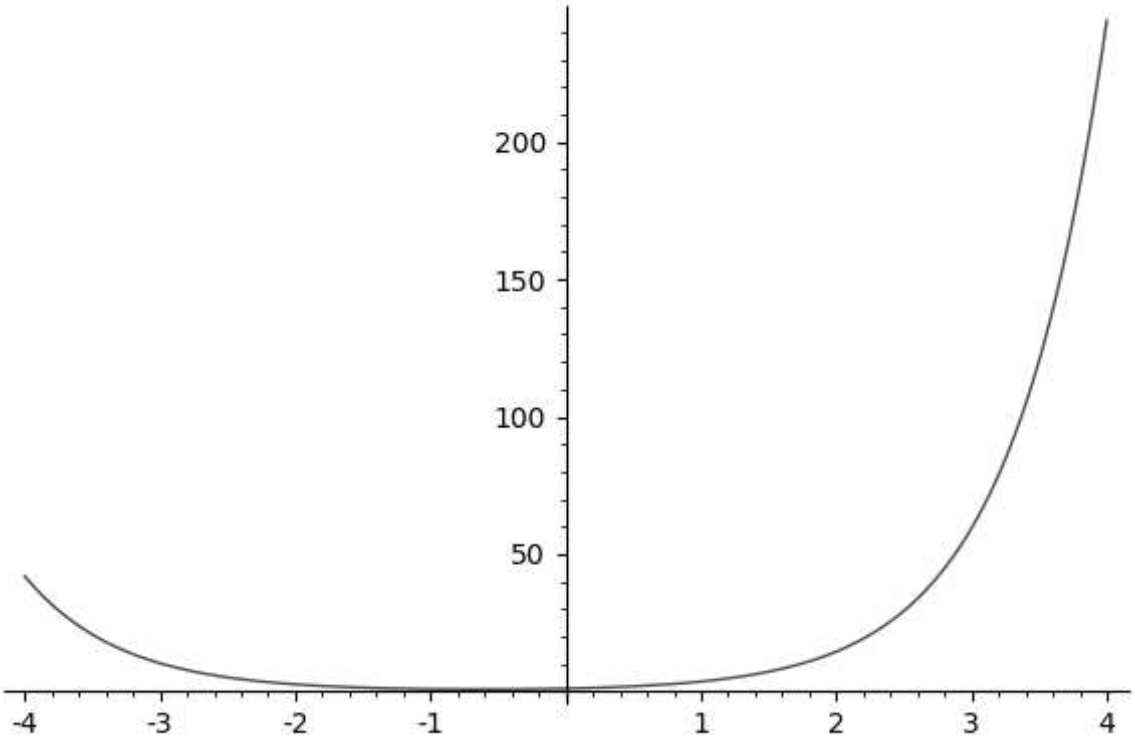
```
show(sol_y)
```

$$t\mapsto\frac{1}{2}\,\sqrt{2}\sinh\left(\sqrt{2}t\right)$$

In [21]:

```
plot(sol_x(t),t,-4,4,color='red')
```

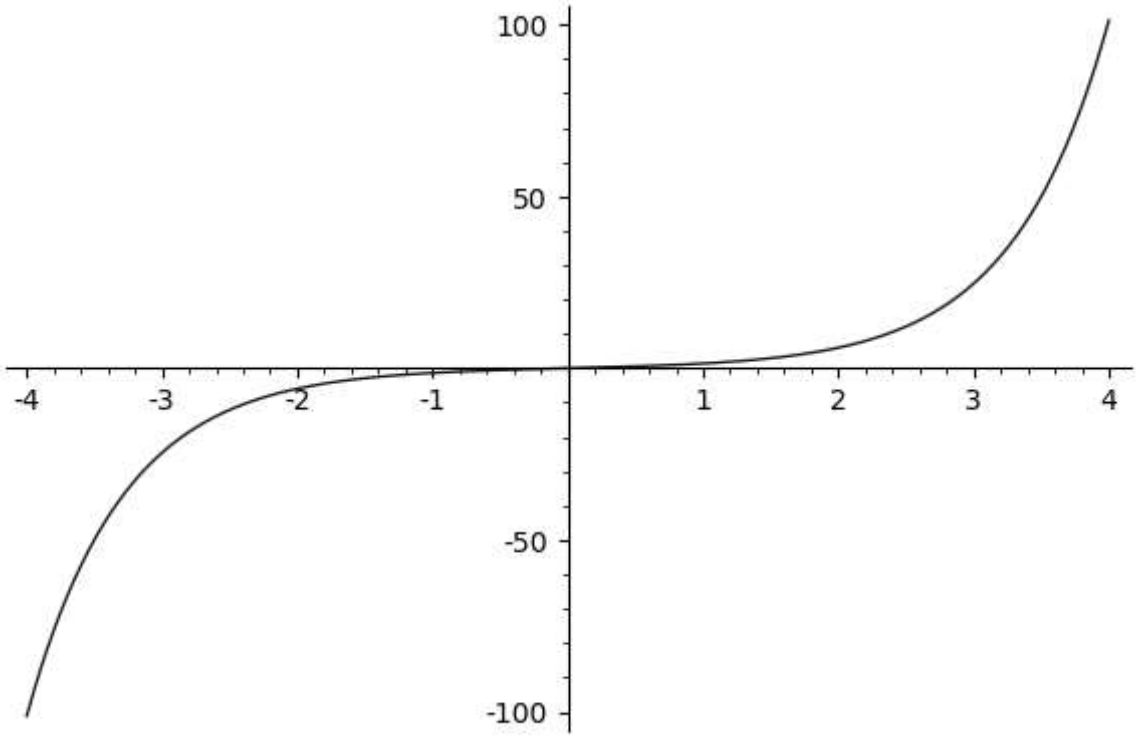
Out[21]:



In [22]:

```
plot(sol_y(t),t,-4,4,color='blue')
```

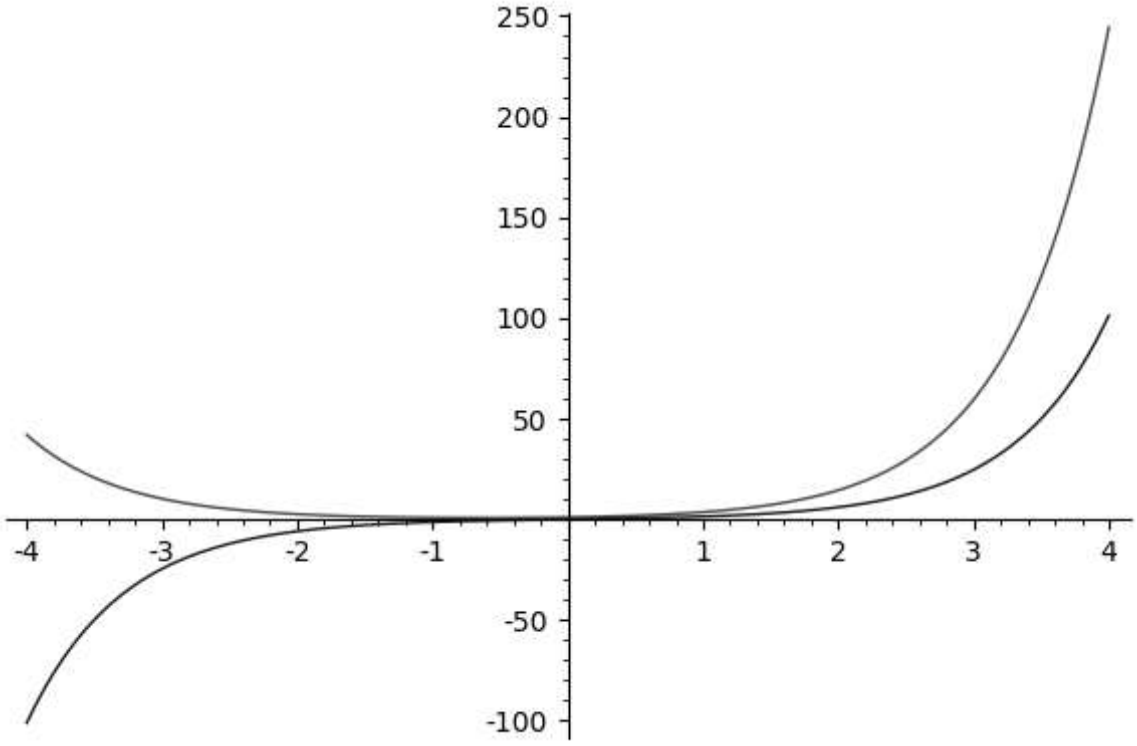
Out[22]:



sau daca dorim reprezentarea ambelor componente ale solutiei in aceeasi fereastra procedam astfel:

In [23]:

```
g1=plot(sol_x(t),t,-4,4,color='red')
g2=plot(sol_y(t),t,-4,4,color='blue')
show(g1+g2)
```



Camp de directii. Portret fazic

Pentru un sistem planar (sistem cu doua ecuatii), *portretul de fazic* este dat de familia de curbe determinata de ecuatiile parametrice (x, y) = (x (t), y (t)), $t \in \mathbb{R}$, corespunzatoare solutiei sistemului si reprezinta proiectia tripletului $(t, x(t), y(t))$ in planul xOy. In acest context, planul cartezian xOy în care se află portretul fazic se numește plan de fază. Curbele parametrice determinate de solutii se mai numesc si traiectorii sau orbite ale acestora.

Portretul fazic este reuniunea tuturor orbitelor impreuna cu sensul de parcurgere al acestora si este un instrument grafic pentru a vizualiza modul in care soluțiile unui sistem de ecuatii diferentiale dat se comporta pe termen lung.

Campul de directii este setul de tangente la traiectoriile sistemului, acestea ne arata sensul de parcurgere al orbitelor.

In general, pentru un sistem de forma

$$\begin{cases} x' = & f_1(x,y) \\ y' = & f_2(x,y) \end{cases}$$

se foloseste comanda *plot_vector_field([f1(x,y),f2(x,y)], [x,a,b], [y,c,d])* pentru a reprezenta campul de directii.

De exemplu, pentru sistemul de ecuatii diferentiale:

$$\begin{cases} x'(t) = & x(t) + y(t) \\ y'(t) = & x(t) - y(t) \end{cases}$$

portretul fazic se obtine in felul urmator:

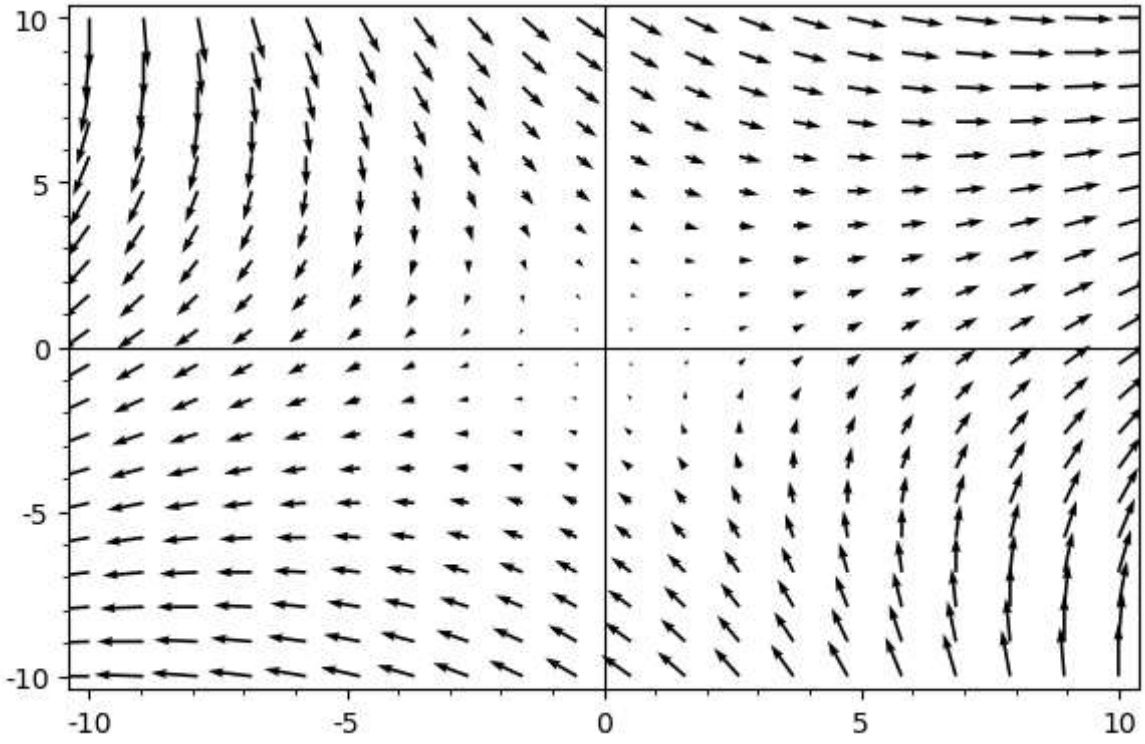
In [24]:

```
f1(u,v)=u+v
f2(u,v)=u-v
```

In [25]:

```
plot_vector_field( [f1(u,v),f2(u,v)], [u,-10,10], [v,-10,10] )
```

Out[25]:

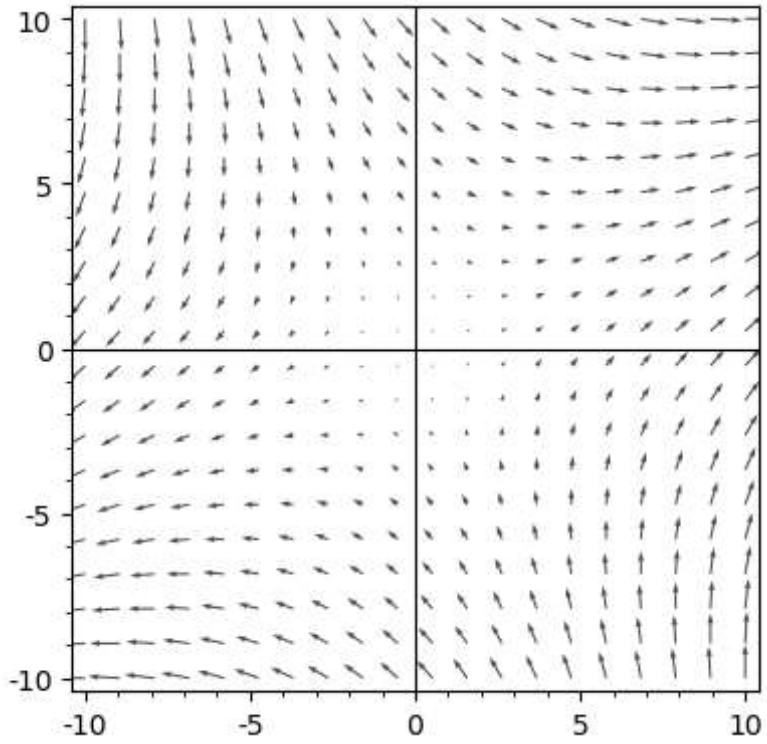


Pentru a folosi aceeasi scala pe ambele axe vom folosi optiunea *aspect_ratio* = *1*

In [26]:

```
plot_vector_field( [f1(u,v),f2(u,v)], [u,-10,10], [v,-10,10], color='red' ,aspect_ratio = 1
```

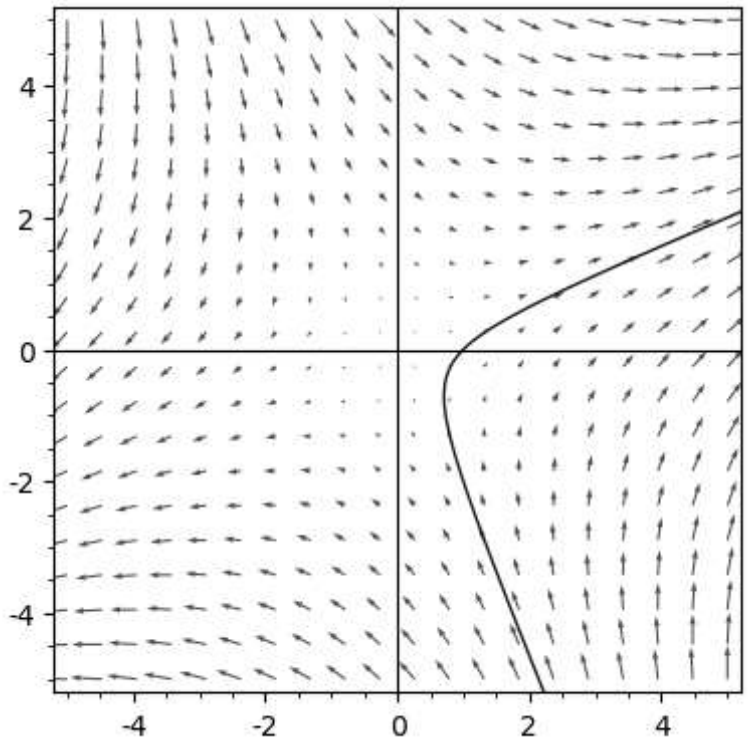
Out[26]:



Pentru reprezentarea orbitei corespunzatoare solutiei ce satisface setul de conditii initiale $x(0) = 1, y(0) = 0$, adica sa reprezentam orbita ce trece prin puctul de coordonate $(1, 0)$, va trebui sa reprezentam curba data de ecuatiile parametrice $(x(t), y(t))$ corespunzatoare acestei solutii. Fiind o curba data prin ecuatii parametrice va trebui sa folosim comanda *parametric_plot*

In [27]:

```
g1=plot_vector_field( [f1(u,v),f2(u,v)], [u,-5,5], [v,-5,5],color='red' ,aspect_ratio = 1)
g2=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
g=g1+g2
g.show(xmin=-5,xmax=5,ymin=-5,ymax=5)
```

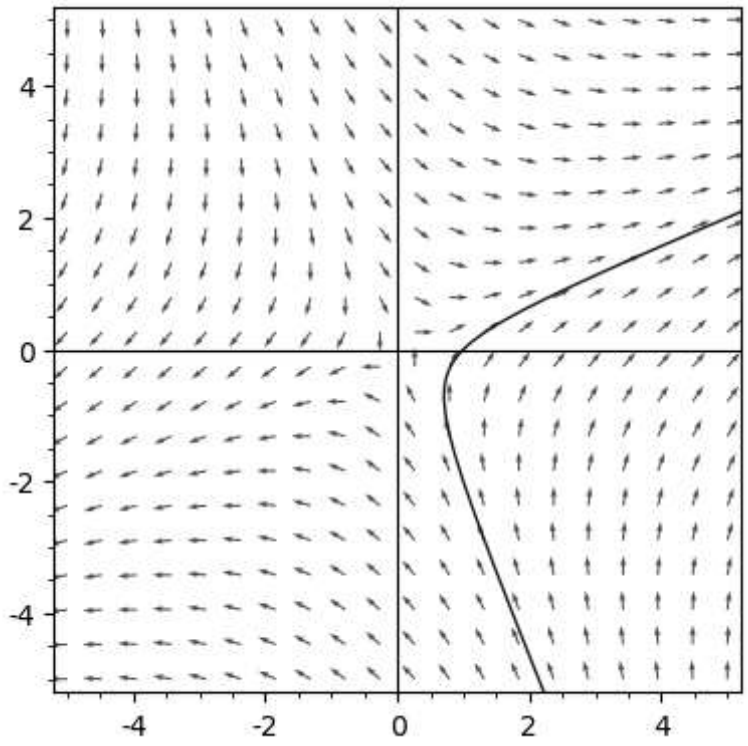


In graficul de mai sus nu vedem ce se intampla in apropierea punctului (0,0) deoarece sagetile sunt prea mici.

Deoarece SageMath nu scalează automat vectorii din campul de directii, pentru a vizualiza mai bine este mai util sa reprezentam fiecare vector în campul de directii cu aceeași lungime. Acest lucru se realizeaza prin scalarea vectorului la norma sa, facandu-l un vector unitate.

In [28]:

```
n=sqrt(f1(u,v)^2+f2(u,v)^2)
g1=plot_vector_field( [f1(u,v)/n,f2(u,v)/n], [u,-5,5], [v,-5,5],color='red' ,aspect_ratio =
g2=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
g=g1+g2
g.show(xmin=-5,xmax=5,ymin=-5,ymax=5)
```



Daca dorim reprezentarea a mai multor orbite va trebui sa generam curba parametrica corespunzatoare fiecarei perechi de conditii initiale $x(0) = x0, y(0) = y0$.

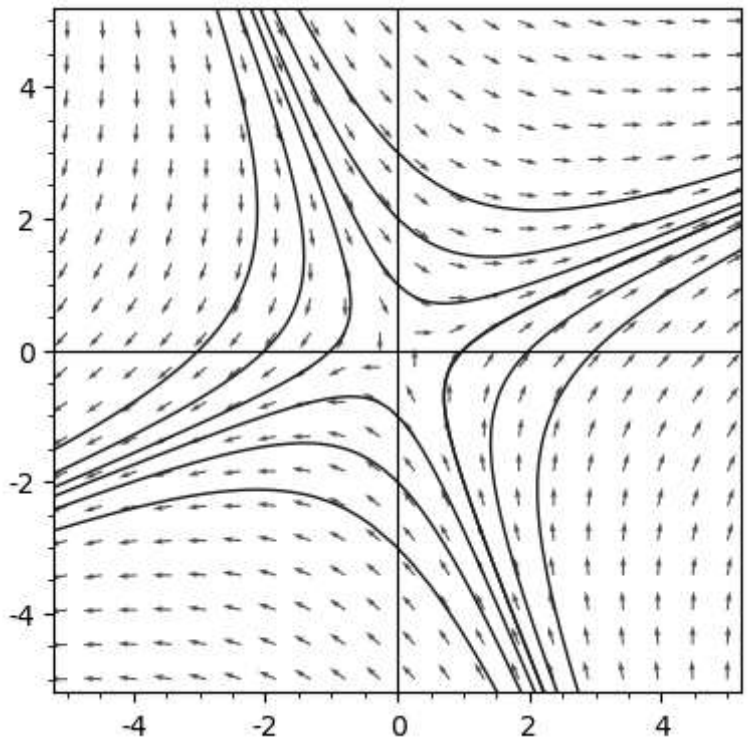
De exemplu, sa reprezentam orbitele corespunzatoare urmatoarelor puncte (corespunzatoare setului de conditii initiale x(0) = x0, y(0) = y0)

(1,0), (2,0), (3,0), (-1,0), (-2,0), (-3,0), (0,1), (0,2), (0,3), (0,-1), (0,-2), (0,-3):

In [29]:

```
g1=plot_vector_field( [f1(u,v),f2(u,v)], [u,-5,5], [v,-5,5],color='red' ,aspect_ratio = 1)
for k in [1..3]:
    in_cond=[0,k,0]
    sol=desolve_system(syst, vars,in_cond)
    sol_x(t)=sol[0].rhs()
    sol_y(t)=sol[1].rhs()
    g1=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
    in_cond=[0,-k,0]
    sol=desolve_system(syst, vars,in_cond)
    sol_x(t)=sol[0].rhs()
    sol_y(t)=sol[1].rhs()
    g2=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
    in_cond=[0,0,k]
    sol=desolve_system(syst, vars,in_cond)
    sol_x(t)=sol[0].rhs()
    sol_y(t)=sol[1].rhs()
    g3=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
    in_cond=[0,0,-k]
    sol=desolve_system(syst, vars,in_cond)
    sol_x(t)=sol[0].rhs()
    sol_y(t)=sol[1].rhs()
    g4=parametric_plot((sol_x(t), sol_y(t)), (t, -10, 10),color='blue')
    g=g+g1+g2+g3+g4

g.show(xmin=-5,xmax=5,ymin=-5,ymax=5)
```



In [ ]:

```
```