Semidefinite optimization and LMIs

August 31, 2021

1 3-dimensional eliptope

[457]:
```
import traceback
load("cvxopt_lmi.sage")

PR=PolynomialRing(QQ,'x,y,z')
x,y,z=PR.gens()

obj=x+2*y+z
print("objective {}".format(obj))
A=matrix(PR,[(1,x,y),(x,1,z),(y,z,1)])
print(A)

lmi_min(obj,[A],verbose=True)
```

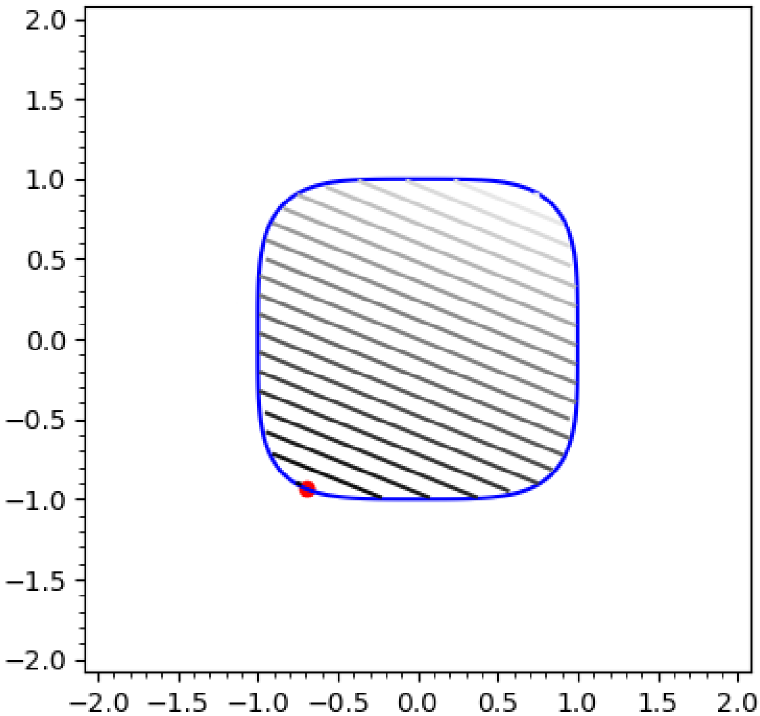
2 l4-disc

[517]:
```
load("cvxopt_lmi.sage")
PR=PolynomialRing(QQ,'x1,x2,y1,y2')
x1,x2,y1,y2=PR.gens()
A=matrix(PR,[(1-y1,y2),(y2,1+y1)])
B=matrix(PR,[(y1,x1),(x1,1)])
C=matrix(PR,[(y2,x2),(x2,1)])
obj=2*x1+5*x2
print(60*"=")
print("Objective {}".format(obj))
print("LMIs")
print()
print(A)
print()
print(B)
print()
print(C)
print(60*"=")
```

```
opt,sol=lmi_min(obj,[A,B,C],verbose=True)

PRx=PolynomialRing(QQ,2,[x1,x2])

l4curve=implicit_plot( PRx(1-x1^4 - x2^4), (x1,-2,2),(x2,-2,2))
objectivePlot=contour_plot(PRx(obj),␣
 ↪(x1,-2,2),(x2,-2,2),fill=False,region=PRx(1-x1^4 - x2^4),contours=20)
solPlot=point((sol[x1],sol[x2]),size=40,color='red')
show(solPlot+l4curve+objectivePlot)
```

```
============================================================
Objective 2*x1 + 5*x2
LMIs

[-y1 + 1      y2]
[     y2  y1 + 1]

[y1 x1]
[x1  1]

[y2 x2]
[x2  1]
============================================================
     pcost       dcost       gap    pres   dres   k/t
 0:  0.0000e+00 -1.4000e+01  4e+01  1e+00  9e-01  1e+00
 1: -5.4953e+00 -6.3765e+00  2e+00  1e-01  8e-02  4e-01
 2: -6.0602e+00 -6.1079e+00  1e-01  6e-03  4e-03  2e-02
 3: -6.0695e+00 -6.0721e+00  6e-03  3e-04  2e-04  1e-03
 4: -6.0688e+00 -6.0690e+00  4e-04  2e-05  1e-05  8e-05
 5: -6.0688e+00 -6.0688e+00  9e-06  4e-07  3e-07  2e-06
 6: -6.0688e+00 -6.0688e+00  5e-07  2e-08  2e-08  9e-08
Optimal solution found.
{'x': <4x1 matrix, tc='d'>, 'y': <0x1 matrix, tc='d'>, 'status': 'optimal',
'gap': 4.68265493999793e-07, 'relative gap': 7.715953916287094e-08, 'primal
objective': -6.068795888106103, 'dual objective': -6.068796075713321, 'primal
infeasibility': 2.3025290647126176e-08, 'dual infeasibility':
1.7279992122895726e-08, 'primal slack': 1.2008832905254807e-08, 'dual slack':
1.4449666921309765e-09, 'residual as primal infeasibility certificate': None,
'residual as dual infeasibility certificate': None, 'iterations': 6, 'sl': <0x1
matrix, tc='d'>, 'ss': [<2x2 matrix, tc='d'>, <2x2 matrix, tc='d'>, <2x2 matrix,
tc='d'>], 'zl': <0x1 matrix, tc='d'>, 'zs': [<2x2 matrix, tc='d'>, <2x2 matrix,
tc='d'>, <2x2 matrix, tc='d'>]}

Optimal value: -6.068795888106103

x1 = -0.6906462550367309
x2 = -0.9375006756065283
```

```
y1 = 0.47699225225625075
y2 = 0.8789075040025888
```

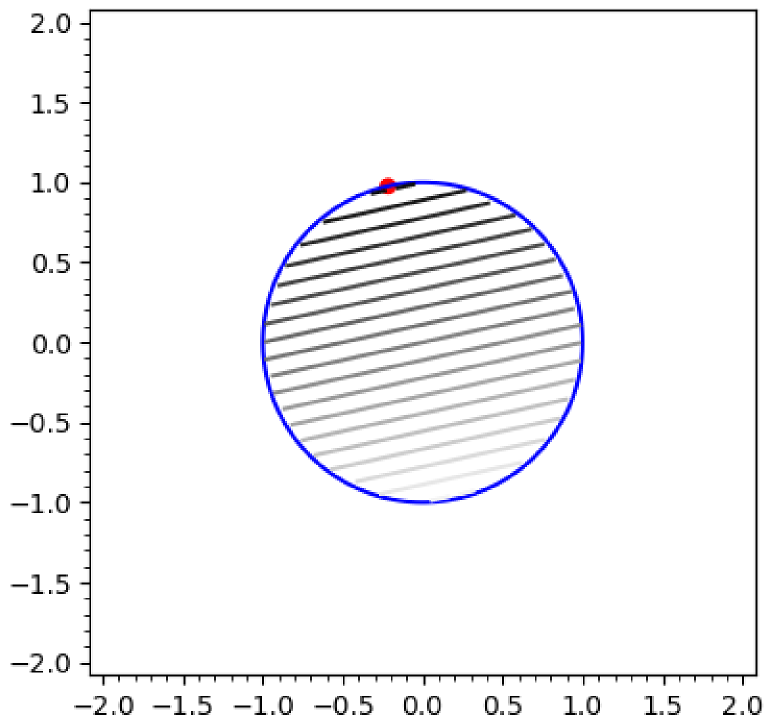


3 Euclidean disc

[519]:
```
load("cvxopt_lmi.sage")
PR=PolynomialRing(QQ,'x,y')
x,y=PR.gens()
A=matrix(PR,[(1-x,y),(y,1+x)])
obj=2*x-9*y
print(60*"=")
print(obj)
print()
print(A)
print(60*"=")

opt,sol=lmi_min(obj,LMIs=[A],verbose=True)


curve=implicit_plot( A.det(), (x,-2,2),(y,-2,2))
objectivePlot=contour_plot(obj, (x,-2,2),(y,-2,2),fill=False,region=A.
 ↪det(),contours=20)
```

```
solPlot=point((sol[x],sol[y]),size=40,color='red')
show(solPlot+curve+objectivePlot)
```

```
============================================================
2*x - 9*y

[-x + 1      y]
[     y  x + 1]
============================================================
     pcost       dcost       gap    pres   dres   k/t
 0: -4.7103e-16 -1.1220e+01  1e+01  0e+00  2e-16  1e+00
 1: -9.1273e+00 -1.0823e+01  2e+00  1e-15  2e-16  4e-01
 2: -9.2180e+00 -9.2356e+00  2e-02  5e-16  4e-16  4e-03
 3: -9.2195e+00 -9.2197e+00  2e-04  3e-16  3e-16  4e-05
 4: -9.2195e+00 -9.2195e+00  2e-06  6e-16  4e-16  4e-07
Optimal solution found.
{'x': <2x1 matrix, tc='d'>, 'y': <0x1 matrix, tc='d'>, 'status': 'optimal',
'gap': 1.755190446233421e-06, 'relative gap': 1.9037713665391923e-07, 'primal
objective': -9.219544306016788, 'dual objective': -9.219546061207232, 'primal
infeasibility': 5.978733960232999e-16, 'dual infeasibility':
3.85345902417251e-16, 'primal slack': 1.640819662979043e-08, 'dual slack':
8.019571744183832e-07, 'residual as primal infeasibility certificate': None,
'residual as dual infeasibility certificate': None, 'iterations': 4, 'sl': <0x1
matrix, tc='d'>, 'ss': [<2x2 matrix, tc='d'>], 'zl': <0x1 matrix, tc='d'>, 'zs':
[<2x2 matrix, tc='d'>]}

Optimal value: -9.219544306016788

x = -0.2169304542592185
y = 0.9761870441664834
```

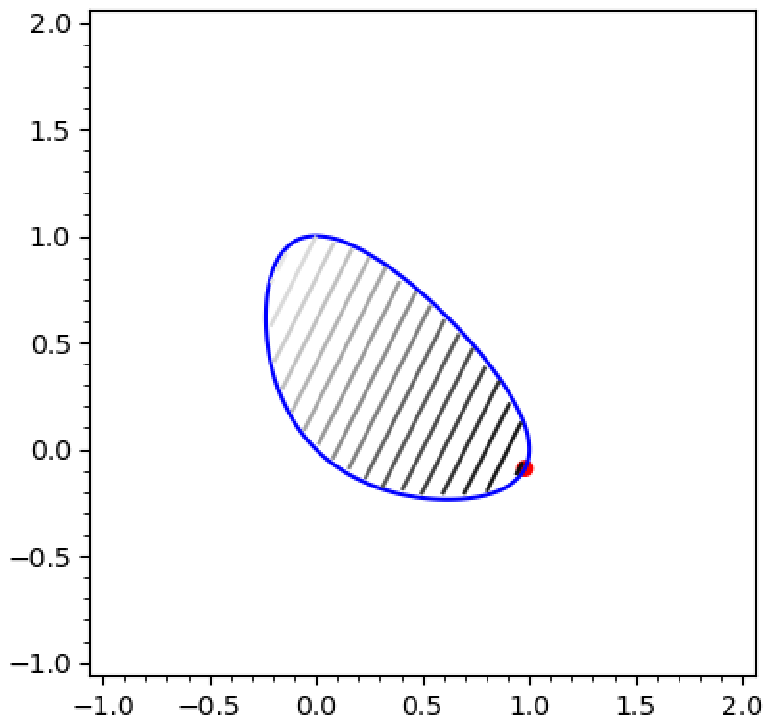


4 Slice of the 3-dimensional eliptope

[520]:
```
load("cvxopt_lmi.sage")
PR.<x,y>=QQ['x,y']
obj=-2*x+y

A=matrix(PR, [(1,x,y), (x,1,1-x-y), (y,1-x-y,1) ] )

print(60*"=")
print(obj)
print()
print(A)
print(60*"=")

opt,sol=lmi_min(obj,LMIs=[A],verbose=True)


curve=implicit_plot( A.det(), (x,-1,2),(y,-1,2))
objectivePlot=contour_plot(obj, (x,-1,2),(y,-1,2),fill=False,region=A.
 ↪det(),contours=20)
solPlot=point((sol[x],sol[y]),size=40,color='red')
```

```
show(solPlot+curve+objectivePlot)
```

```
============================================================
-2*x + y

[         1          x          y]
[         x          1 -x - y + 1]
[         y -x - y + 1          1]
============================================================
     pcost       dcost       gap    pres   dres   k/t
 0: -3.3333e-01 -6.7897e+00  6e+00  5e-17  3e-16  1e+00
 1: -1.9856e+00 -2.4691e+00  5e-01  1e-15  5e-16  1e-01
 2: -2.0492e+00 -2.0552e+00  6e-03  8e-16  2e-15  2e-03
 3: -2.0500e+00 -2.0503e+00  4e-04  1e-15  1e-15  1e-04
 4: -2.0500e+00 -2.0501e+00  9e-05  1e-15  3e-15  2e-05
 5: -2.0500e+00 -2.0500e+00  6e-06  8e-16  6e-16  2e-06
 6: -2.0500e+00 -2.0500e+00  1e-06  3e-16  4e-15  3e-07
Optimal solution found.
{'x': <2x1 matrix, tc='d'>, 'y': <0x1 matrix, tc='d'>, 'status': 'optimal',
'gap': 1.015702854839262e-06, 'relative gap': 4.954661320491577e-07, 'primal
objective': -2.0499945185728436, 'dual objective': -2.049995534275706, 'primal
infeasibility': 3.4443763517810415e-16, 'dual infeasibility':
3.55271367809424e-15, 'primal slack': 1.4324571492061142e-07, 'dual slack':
4.44301737936245e-08, 'residual as primal infeasibility certificate': None,
'residual as dual infeasibility certificate': None, 'iterations': 6, 'sl': <0x1
matrix, tc='d'>, 'ss': [<3x3 matrix, tc='d'>], 'zl': <0x1 matrix, tc='d'>, 'zs':
[<3x3 matrix, tc='d'>]}

Optimal value: -2.0499945185728436

x = 0.979963126065743
y = -0.09006826644135767
```


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

[ ]: