```
import os, sys, os.path, shutil
import matplotlib as mpl

import xlrd, openpyxl
import hashlib
import pweave
from utils_p9 import *

mode_serv = False
for param in sys.argv[1:]:
    if param[:len("serv")]=="serv" and param[len("serv")] == "=":
        mode_serv = param[len("serv")+1:] in "1 y Y yes Yes YES"
#if not mode_serv:
#    mpl.use('TkAgg')
#else:
#    mpl.use('Agg')
if mode_serv:
    mpl.use('Agg')
try:
    import appnope
    appnope.nope()    # stop the apple power nap
except ImportError:
    pass

import pandas
from math import pi
import math
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import matplotlib.patches as mpatches
from matplotlib.collections import PatchCollection
import numpy as np
from numpy.random import rand
from pylab import pcolor, show, colorbar, xticks, yticks, pcolormesh, imshow
from random import gauss
from scipy.interpolate import interp2d
import threading, multiprocessing, time
from decimal import Decimal
import scipy

from scipy.optimize import curve_fit
mpl.rc('text', usetex=True)
#plt.rc('text', usetex=True)
mpl.rc('text.latex',preamble=r"\usepackage{amsmath} \usepackage{graphicx} \usepackage{ni

import scipy as sp
import subprocess
import datetime
from matplotlib.ticker import PercentFormatter
from mpl_toolkits.mplot3d import Axes3D

from mpl_toolkits.mplot3d.art3d import Poly3DCollection, Line3DCollection
import itertools
import sympy

def renorm(a):
 v=a.to_numpy()
 # print(v)
```

```
 s=[]
 min_v=np.amin(v)
 max_v=np.amax(v)
 dif=max_v-min_v
 for val in v:
     s+=[(val-min_v)/dif*2-1]
 return s

def move_sympyplot_to_axes(p, ax): # to do multiple plot with sympy (from stackoverflow,
    backend = p.backend(p)
    backend.ax = ax
    # backend.process_series()
    backend.ax.spines['right'].set_color('none')
    backend.ax.spines['bottom'].set_position('zero')
    backend.ax.spines['top'].set_color('none')
    plt.close(backend.fig)



# In : The experiment matrix, the number of rows to be reduced to, and the maximum numbe
# Out: "D-optimized information matrix" and information matrix determinant
# principle: a replacemnt for matlab candexch function, iterative procedure, do not sort
def candexch(C,nrows,n_try=800):
    rows=range(len(C))
    opti_mat=[]
    start_id=np.random.choice(len(C),size=nrows,replace=True)
    start_point=[C.tolist()[i] for i in start_id]
    # print("start_point",start_point)
    X=np.mat(start_point)
    ref_det=np.linalg.det(np.matmul(X.T,X))
    min_dif_det=1e100
    det=0
    max_det=0
    opti_mat1=np.zeros(X.shape)
    opti_mat_old=np.zeros(X.shape)
    opti_mat2=X
    itera=0
    while((not np.array_equal(opti_mat_old,opti_mat2)) and (itera<n_try)):
        opti_mat_old=opti_mat2
        min_dif_det=math.inf
        det=0
        max_det=0
        for row in C:
            X=np.vstack([opti_mat2,row])
            # print(X)
            det=np.linalg.det(np.matmul(X.T,X))
            if abs(det) > max_det:
                max_det=abs(det)
                opti_mat1=X
        # print(opti_mat1)
        for i in range(len(opti_mat1)):
            X=np.delete(opti_mat1,i,axis=0)
            # print(X)
            det=np.linalg.det(np.matmul(X.T,X))
            if abs(max_det-det) < min_dif_det:
                min_dif_det=abs(max_det-det)
                opti_mat2=X
        itera +=1
        # print(itera)
```

```
        # print(opti_mat2)
    print("number of iter",itera)
    return opti_mat2, max_det




c1=["H      ",    "t-C4H9    ",    "i-C3H7    ",    "C2H5      ",    "CH3       ",    "CH
c2=[0     ,    0     ,    0     ,    0     ,    0     ,    0.14 ,    0.1  ,    0.06 ,    0.14
c3=[0,-0.07,-0.07,-0.07,-0.08,-0.06,-0.22,-0.15,-0.52,-0.58,-0.64,-0.38,0     ,-0.42,-0.1
c4=[4390,4180,4060,3950,3850,3510,3350,3260,2600,2520,2250,2110,2020,1700,1290,1230,1060


df = pandas.DataFrame(data={'Amines': c1, 'SigmaF': c2, 'SigmaR': c3, 'K': c4},)

print(df)


fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter3D(df["SigmaF"],df["SigmaR"],df["K"])
ax.set_xlabel("$\sigma_F$")
ax.set_ylabel("$\sigma_R$")
ax.set_zlabel("K")
# print(df["SigmaF"].dtype)

X_lin=np.mat((np.ones(len(df["SigmaF"])),renorm(df["SigmaF"]),renorm(df["SigmaR"]))).T
print(X_lin)
X_int=np.mat((np.ones(len(df["SigmaF"])),renorm(df["SigmaF"]),renorm(df["SigmaR"]),np.mu
print(X_int)
X_quad=np.mat((np.ones(len(df["SigmaF"])),renorm(df["SigmaF"]),renorm(df["SigmaR"]),np.r
print(X_quad)
# X_quad2=np.mat((np.ones(len(df["SigmaF"])),renorm(np.power(df["SigmaF"],2)),renorm(np.
# print(X_quad2)


d_lin= np.linalg.inv( np.matmul(X_lin.T,X_lin))

d_int= np.linalg.inv( np.matmul(X_int.T,X_int))

d_quad= np.linalg.inv( np.matmul(X_quad.T,X_quad))

print(d_lin)
print(d_int)
print(d_quad)


x1,x2,k=sympy.symbols('x1 x2 k') #not for symbolic calculation
f_lin=np.array([1,x1,x2])
f_int=np.array([1,x1,x2,x1*x2])
f_quad=np.array([1,x1,x2,x1*x2,x1*x1,x2*x2])


std_lin, corr_lin=cov_to_sig_cor(d_lin)
std_int, corr_int=cov_to_sig_cor(d_int)
```

```
std_quad, corr_quad=cov_to_sig_cor(d_quad)


VIF_lin=np.diag(np.linalg.inv(corr_lin))
VIF_int=np.diag(np.linalg.inv(corr_int))
VIF_quad=np.diag(np.linalg.inv(corr_quad))

Var_lin =f_lin.T.dot(d_lin.dot(f_lin).T)
Var_int =f_int.T.dot(d_int.dot(f_int).T)
Var_quad=f_quad.T.dot(d_quad.dot(f_quad).T)


print(d_lin)
print(d_int)
print(d_quad)

print(VIF_lin)
print(VIF_int)
print(VIF_quad)

Var_lin =Var_lin[0,0]
Var_int =Var_int[0,0]
Var_quad=Var_quad[0,0]

# fig1=plt.figure(2)
# ax1=plt.subplot(131,projection="3d")
# ax2=plt.subplot(132,projection="3d")
# ax3=plt.subplot(133,projection="3d")



# fig, (ax1,ax2,ax3) = plt.subplots(ncols=3)




p1=sympy.plotting.plot3d(Var_lin, (x1, -1, 1), (x2, -1, 1))
# ax2=plt.subplot(312)
p2=sympy.plotting.plot3d(Var_int, (x1, -1, 1), (x2, -1, 1))
# ax3=plt.subplot(313)
p3=sympy.plotting.plot3d(Var_quad, (x1, -1, 1), (x2, -1, 1))


# move_sympyplot_to_axes(p1, ax)
# move_sympyplot_to_axes(p2, ax2)
# move_sympyplot_to_axes(p3, ax3)

X_int_5,det_5=candexch(  X_int,5)
# exit()
X_int_10,det_10=candexch(X_int,10)
X_int_15,det_15=candexch(X_int,15)
print(X_int_5 )
print(X_int_10)
print(X_int_15)
print( np.sort( X_int_5.view("float,float,float,float") ,order=['f3','f2','f1'], axis=0)
print( np.sort(X_int_10.view("float,float,float,float"),order=['f3','f2','f1'], axis=0))
print( np.sort(X_int_15.view("float,float,float,float"),order=['f3','f2','f1'], axis=0))
print( np.sort((np.multiply( X_int_5+1,[1,0.65/2,0.64/2,1])[:,1:3]-[0,0.64]).view("float
```

```
print( np.sort((np.multiply(X_int_10+1,[1,0.65/2,0.64/2,1])[:,1:3]-[0,0.64]).view("float
print( np.sort((np.multiply(X_int_15+1,[1,0.65/2,0.64/2,1])[:,1:3]-[0,0.64]).view("float
print(1/det_5)
print(1/det_10)
print(1/det_15)
d_int_5 = np.linalg.inv( np.matmul(X_int_5.T,X_int_5))
d_int_10= np.linalg.inv( np.matmul(X_int_10.T,X_int_10))
d_int_15= np.linalg.inv( np.matmul(X_int_15.T,X_int_15))
print(d_int_5 )
print(d_int_10)
print(d_int_15)
print(d_int_5.trace() )
print(d_int_10.trace())
print(d_int_15.trace())
print(1/det_5)
print(1/det_10)
print(1/det_15)



std_int, corr_int_5 =cov_to_sig_cor(d_int_5)
std_int, corr_int_10=cov_to_sig_cor(d_int_10)
std_int, corr_int_15=cov_to_sig_cor(d_int_15)


VIF_int_5=np.diag(np.linalg.inv( corr_int_5 ))
VIF_int_10=np.diag(np.linalg.inv(corr_int_10))
VIF_int_15=np.diag(np.linalg.inv(corr_int_15))

Var_int_5 =f_int.T.dot(d_int_5.dot(f_int).T)
Var_int_10=f_int.T.dot(d_int_10.dot(f_int).T)
Var_int_15=f_int.T.dot(d_int_15.dot(f_int).T)

print(VIF_int_5 )
print(VIF_int_10)
print(VIF_int_15)

Var_int_5  =Var_int_5[0,0]
Var_int_10 =Var_int_10[0,0]
Var_int_15 =Var_int_15[0,0]

p4=sympy.plotting.plot3d(Var_int_5 , (x1, -1, 1), (x2, -1, 1))
p5=sympy.plotting.plot3d(Var_int_10, (x1, -1, 1), (x2, -1, 1))
p6=sympy.plotting.plot3d(Var_int_15, (x1, -1, 1), (x2, -1, 1))

'''
EXP_l=np.mat([[ 1,-1,-1,-1],
              [ 1,-1,-1,0],
              [ 1,-1,-1,1],
              [ 1,-1,1,0],
              [ 1,0,0,0],
              [ 1,1,-1,0],
              [ 1,1,1,-1],
              [ 1,1,1,0],
              [ 1,1,1,1]])

EXP_int=np.mat([[1,-1,-1,-1, 1],
                [1,-1,-1,0 , 1],
                [1,-1,-1,1 , 1],
```

```
                [1,-1,1,0  , -1],
                [1,0,0,0   , 0],
                [1,1,-1,0  , -1],
                [1,1,1,-1  , 1],
                [1,1,1,0   , 1],
                [1,1,1,1   , 1]])


d_lin= np.linalg.inv( np.matmul(EXP_l.T,EXP_l))
d_int= np.linalg.inv( np.matmul(EXP_int.T,EXP_int))

std_lin, corr_lin=cov_to_sig_cor(d_lin)
std_int, corr_int=cov_to_sig_cor(d_int)

VIF_lin=np.diag(np.linalg.inv(corr_lin))
VIF_int=np.diag(np.linalg.inv(corr_int))

C,S,T=sympy.symbols('C S T')
f_lin=np.array([1,C,S,T])
f_int=np.array([1,C,S,T,S*C])


Var_lin=f_lin.T.dot(d_lin.dot(f_lin).T)
Var_int=f_int.T.dot(d_int.dot(f_int).T)

Var_lin=Var_lin[0,0]
print(Var_lin)
print(Var_lin.subs(T,0))

print(EXP_l)
print(d_lin)
print(VIF_lin)
print(corr_lin)

# fig1=plt.figure(1)
# ax1=plt.subplot(111)
# ax1=sympy.plotting.plot3d(Var_lin.subs(T,0), (C, -1, 1), (S, -1, 1))



Var_int=Var_int[0,0]
print(Var_int)
print(Var_int.subs(T,0))

print(EXP_int)
print(d_int)
print(VIF_int)
print(corr_int)


c_lin=scipy.stats.t(df=5).ppf((0.025, 0.975))[1]

sympy.plotting.plot3d(Var_lin.subs(T,0),Var_int.subs(T,0), (C, -1, 1), (S, -1, 1))
'''
plt.show(block=True)
```

```
        Amines  SigmaF  SigmaR     K
0       H         0.00    0.00  4390
1   t-C4H9        0.00   -0.07  4180
```

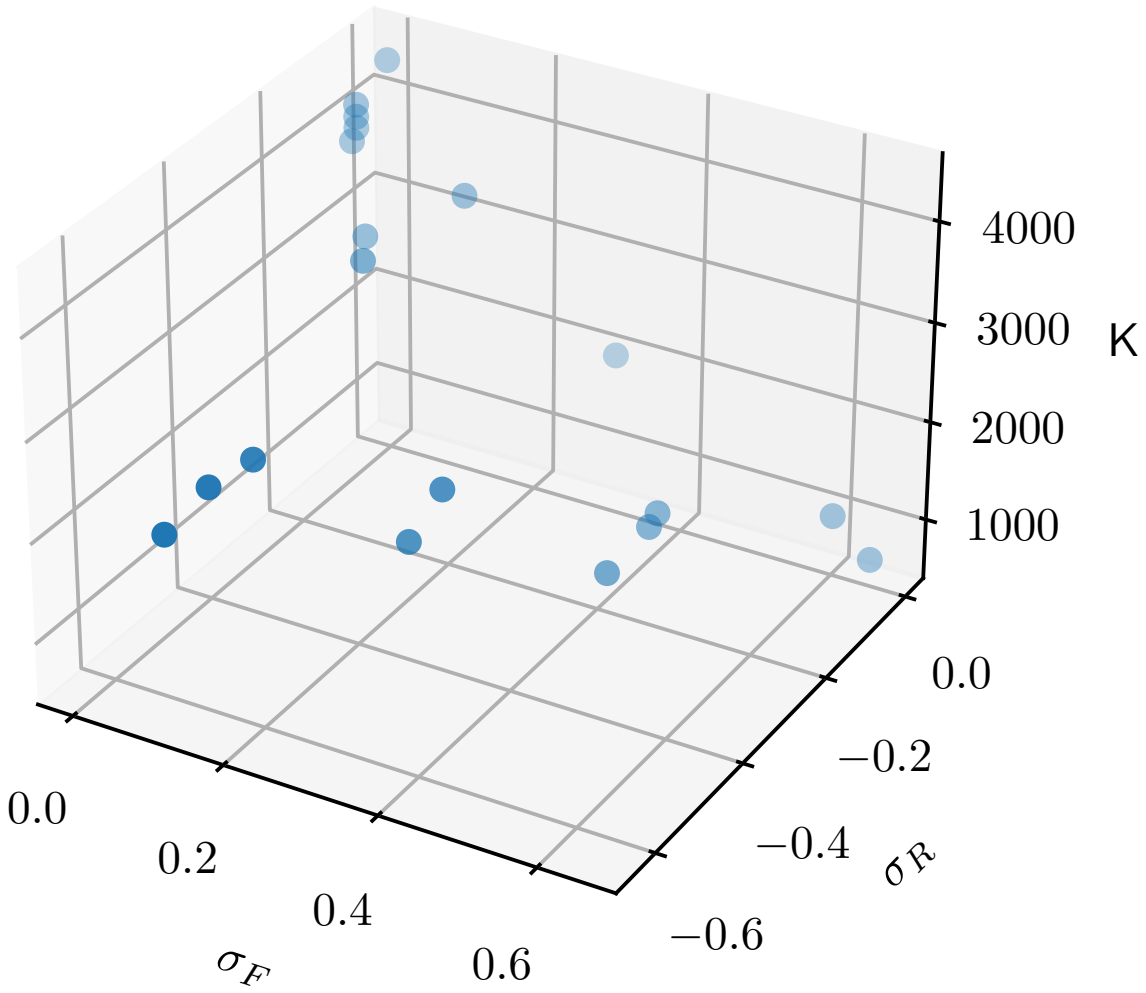
```
2   i-C3H7        0.00   -0.07  4060
3   C2H5          0.00   -0.07  3950
4   CH3           0.00   -0.08  3850
5   CH2OH         0.14   -0.06  3510
6   C6H5          0.10   -0.22  3350
7   CH=CH2        0.06   -0.15  3260
8   NH2           0.14   -0.52  2600
9   NHCH3         0.12   -0.58  2520
10  N(CH3)2       0.10   -0.64  2250
11  OH            0.30   -0.38  2110
12  CO2C2H5       0.31    0.00  2020
13  OCH3          0.28   -0.42  1700
14  Br            0.45   -0.15  1290
15  Cl            0.45   -0.17  1230
16  F             0.44   -0.25  1060
17  CN            0.60    0.00  1000
18         NO2    0.65    0.00   660
[[ 1.          -1.           1.         ]
 [ 1.          -1.           0.78125    ]
 [ 1.          -1.           0.78125    ]
 [ 1.          -1.           0.78125    ]
 [ 1.          -1.           0.75       ]
 [ 1.          -0.56923077   0.8125     ]
 [ 1.          -0.69230769   0.3125     ]
 [ 1.          -0.81538462   0.53125    ]
 [ 1.          -0.56923077  -0.625      ]
 [ 1.          -0.63076923  -0.8125     ]
 [ 1.          -0.69230769  -1.         ]
 [ 1.          -0.07692308  -0.1875     ]
 [ 1.          -0.04615385   1.         ]
 [ 1.          -0.13846154  -0.3125     ]
 [ 1.           0.38461538   0.53125    ]
 [ 1.           0.38461538   0.46875    ]
 [ 1.           0.35384615   0.21875    ]
 [ 1.           0.84615385   1.         ]
 [ 1.           1.           1.         ]]
[[ 1.          -1.           1.          -1.         ]
 [ 1.          -1.           0.78125    -0.78125    ]
 [ 1.          -1.           0.78125    -0.78125    ]
 [ 1.          -1.           0.78125    -0.78125    ]
 [ 1.          -1.           0.75       -0.75       ]
 [ 1.          -0.56923077   0.8125     -0.4625     ]
 [ 1.          -0.69230769   0.3125     -0.21634615]
 [ 1.          -0.81538462   0.53125    -0.43317308]
 [ 1.          -0.56923077  -0.625       0.35576923]
 [ 1.          -0.63076923  -0.8125      0.5125     ]
 [ 1.          -0.69230769  -1.          0.69230769]
 [ 1.          -0.07692308  -0.1875      0.01442308]
 [ 1.          -0.04615385   1.         -0.04615385]
 [ 1.          -0.13846154  -0.3125      0.04326923]
 [ 1.           0.38461538   0.53125     0.20432692]
 [ 1.           0.38461538   0.46875     0.18028846]
 [ 1.           0.35384615   0.21875     0.07740385]
 [ 1.           0.84615385   1.          0.84615385]
 [ 1.           1.           1.          1.        ]]
[[ 1.          -1.           1.          -1.          1.          1.
]
 [ 1.          -1.           0.78125    -0.78125     1.
0.61035156]
```

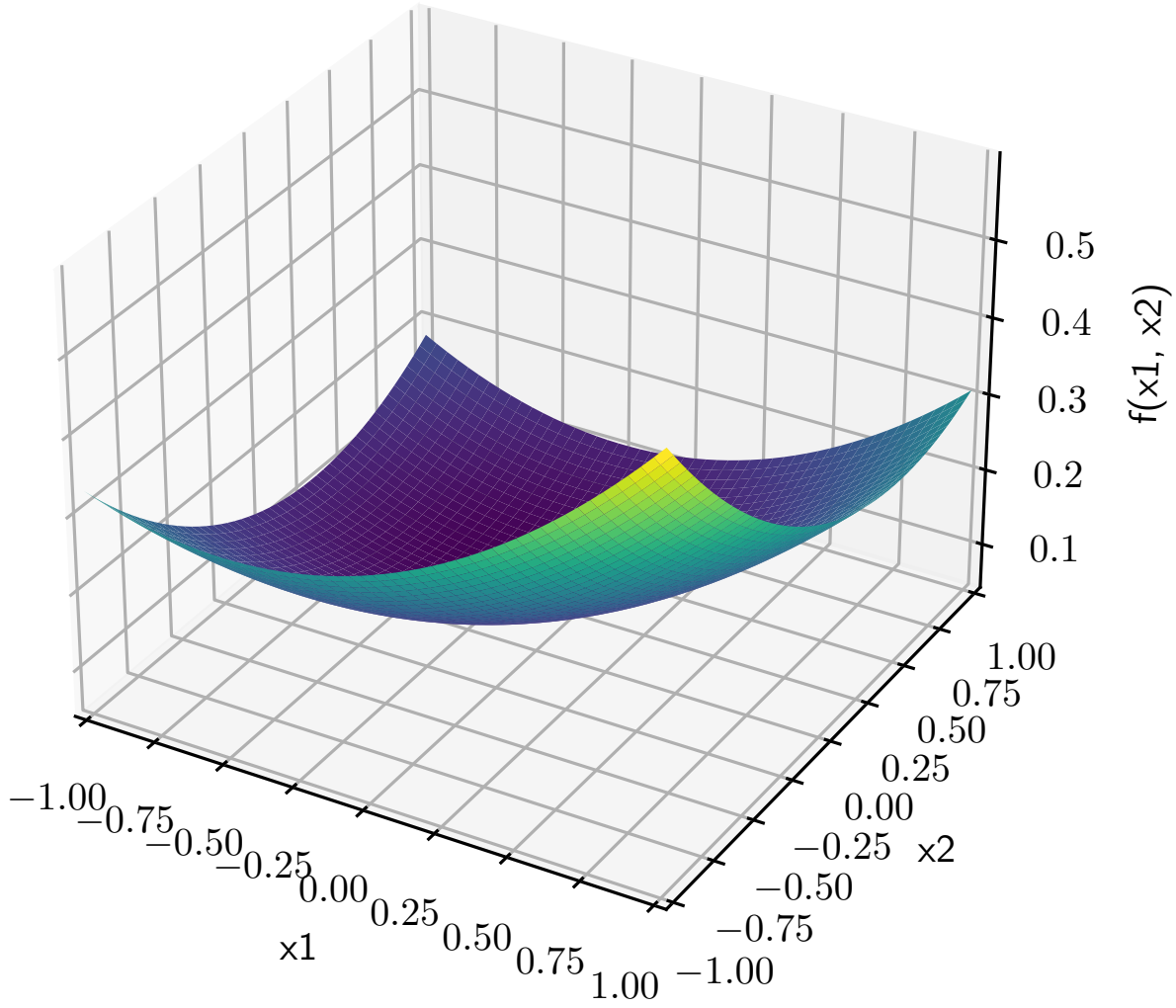
```
 [ 1.         -1.          0.78125    -0.78125     1.
0.61035156]
 [ 1.         -1.          0.78125    -0.78125     1.
0.61035156]
 [ 1.         -1.          0.75       -0.75        1.          0.5625
]
 [ 1.         -0.56923077  0.8125     -0.4625      0.32402367
0.66015625]
 [ 1.         -0.69230769  0.3125     -0.21634615  0.47928994
0.09765625]
 [ 1.         -0.81538462  0.53125    -0.43317308  0.66485207
0.28222656]
 [ 1.         -0.56923077 -0.625       0.35576923  0.32402367
0.390625  ]
 [ 1.         -0.63076923 -0.8125      0.5125      0.39786982
0.66015625]
 [ 1.         -0.69230769 -1.          0.69230769  0.47928994  1.
]
 [ 1.         -0.07692308 -0.1875      0.01442308  0.00591716
0.03515625]
 [ 1.         -0.04615385  1.         -0.04615385  0.00213018  1.
]
 [ 1.         -0.13846154 -0.3125      0.04326923  0.0191716
0.09765625]
 [ 1.          0.38461538  0.53125     0.20432692  0.14792899
0.28222656]
 [ 1.          0.38461538  0.46875     0.18028846  0.14792899
0.21972656]
 [ 1.          0.35384615  0.21875     0.07740385  0.1252071
0.04785156]
 [ 1.          0.84615385  1.          0.84615385  0.71597633  1.
]
 [ 1.          1.          1.          1.          1.          1.
]]
[[ 0.0894431   0.04948465 -0.05540536]
 [ 0.04948465  0.13088856 -0.01715833]
 [-0.05540536 -0.01715833  0.13443759]]
[[ 0.20736547  0.2972985  -0.21613842 -0.33295499]
 [ 0.2972985   0.65166941 -0.35493888 -0.69970496]
 [-0.21613842 -0.35493888  0.3535234   0.45383144]
 [-0.33295499 -0.69970496  0.45383144  0.94010183]]
[[ 0.21953193  0.17176864 -0.1121879  -0.18794423 -0.05352792
-0.10125839]
 [ 0.17176864  1.9803888  -1.4536474  -2.18835077  0.70603181
0.9357907 ]
 [-0.1121879  -1.4536474   1.26211161  1.68692841 -0.57734478
-0.78010751]
 [-0.18794423 -2.18835077  1.68692841  2.67016115 -0.60346954
-1.23135596]
 [-0.05352792  0.70603181 -0.57734478 -0.60346954  0.94027386
-0.05398271]
 [-0.10125839  0.9357907  -0.78010751 -1.23135596 -0.05398271
1.196739  ]]
[[ 0.0894431   0.04948465 -0.05540536]
 [ 0.04948465  0.13088856 -0.01715833]
 [-0.05540536 -0.01715833  0.13443759]]
[[ 0.20736547  0.2972985  -0.21613842 -0.33295499]
 [ 0.2972985   0.65166941 -0.35493888 -0.69970496]
 [-0.21613842 -0.35493888  0.3535234   0.45383144]
```

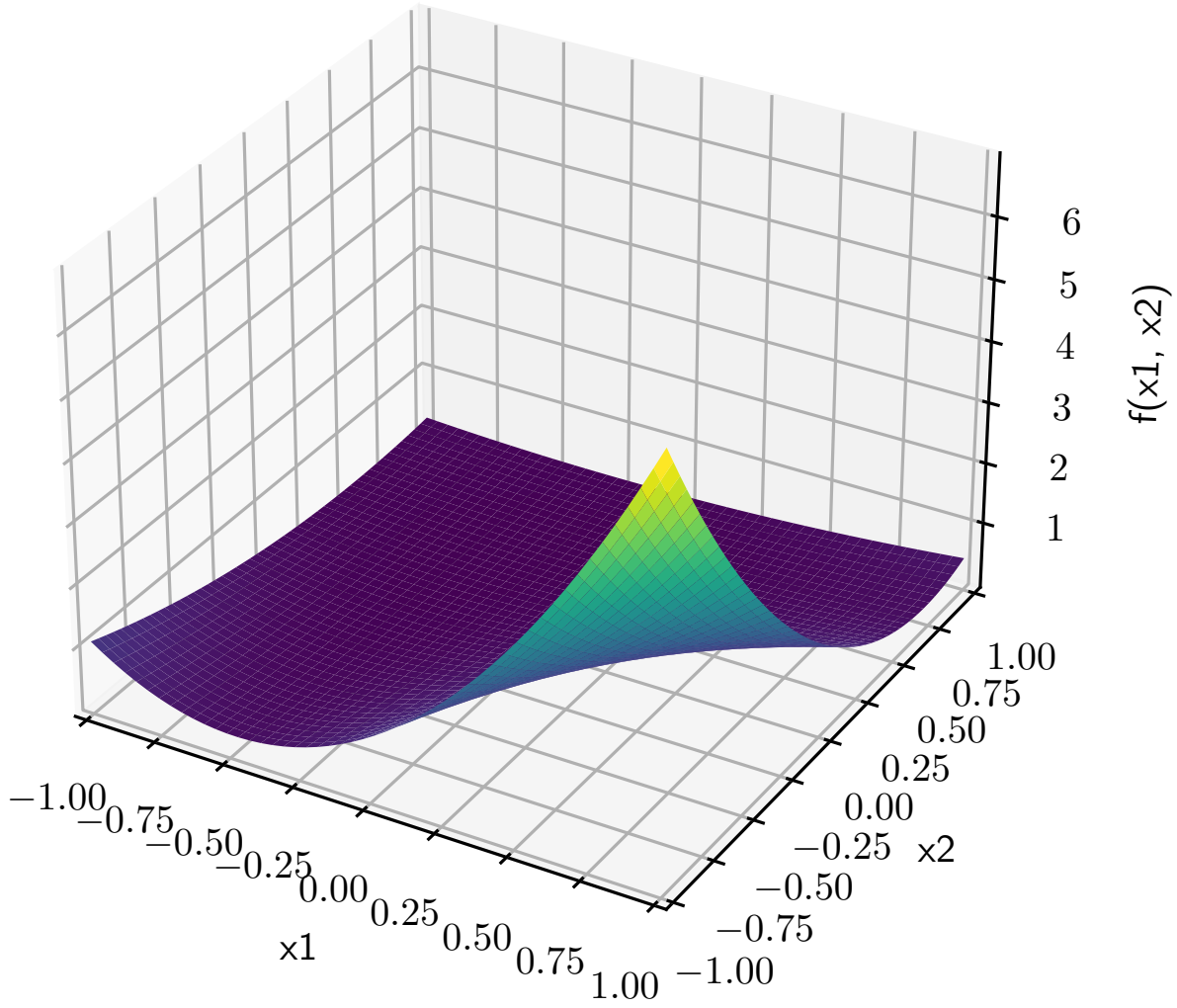
```
 [-0.33295499 -0.69970496  0.45383144  0.94010183]]
[[ 0.21953193  0.17176864 -0.1121879  -0.18794423 -0.05352792
-0.10125839]
 [ 0.17176864  1.9803888  -1.4536474  -2.18835077  0.70603181
0.9357907 ]
 [-0.1121879  -1.4536474   1.26211161  1.68692841 -0.57734478
-0.78010751]
 [-0.18794423 -2.18835077  1.68692841  2.67016115 -0.60346954
-1.23135596]
 [-0.05352792  0.70603181 -0.57734478 -0.60346954  0.94027386
-0.05398271]
 [-0.10125839  0.9357907  -0.78010751 -1.23135596 -0.05398271
1.196739  ]]
[1.69941896 1.28710698 1.3668259 ]
[3.93994385 6.40826244 3.59426965 6.12086668]
[ 4.17110659 19.47437005 12.83187884 17.385032    7.37383544
9.20097994]
number of iter 9
number of iter 12
number of iter 19
[[ 1.          1.          1.          1.        ]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.         -1.          1.         -1.        ]
 [ 1.          0.35384615  0.21875     0.07740385]
 [ 1.         -0.69230769 -1.          0.69230769]]
[[ 1.         -0.69230769 -1.          0.69230769]
 [ 1.          1.          1.          1.        ]
 [ 1.          0.35384615  0.21875     0.07740385]
 [ 1.          1.          1.          1.        ]
 [ 1.          1.          1.          1.        ]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.         -1.          1.         -1.        ]
 [ 1.         -1.          1.         -1.        ]
 [ 1.         -1.          1.         -1.        ]
 [ 1.          0.35384615  0.21875     0.07740385]]
[[ 1.          0.35384615  0.21875     0.07740385]
 [ 1.          0.35384615  0.21875     0.07740385]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.         -1.          1.         -1.        ]
 [ 1.          1.          1.          1.        ]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.         -0.69230769 -1.          0.69230769]
 [ 1.          1.          1.          1.        ]
 [ 1.          0.35384615  0.21875     0.07740385]
 [ 1.          0.35384615  0.21875     0.07740385]
 [ 1.          1.          1.          1.        ]
 [ 1.          1.          1.          1.        ]
 [ 1.         -1.          1.         -1.        ]
 [ 1.         -1.          1.         -1.        ]]
[[(1., -1.        ,  1.     , -1.        )]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1.,  1.        ,  1.     ,  1.        )]]
[[(1., -1.        ,  1.     , -1.        )]
 [(1., -1.        ,  1.     , -1.        )]
 [(1., -1.        ,  1.     , -1.        )]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
```

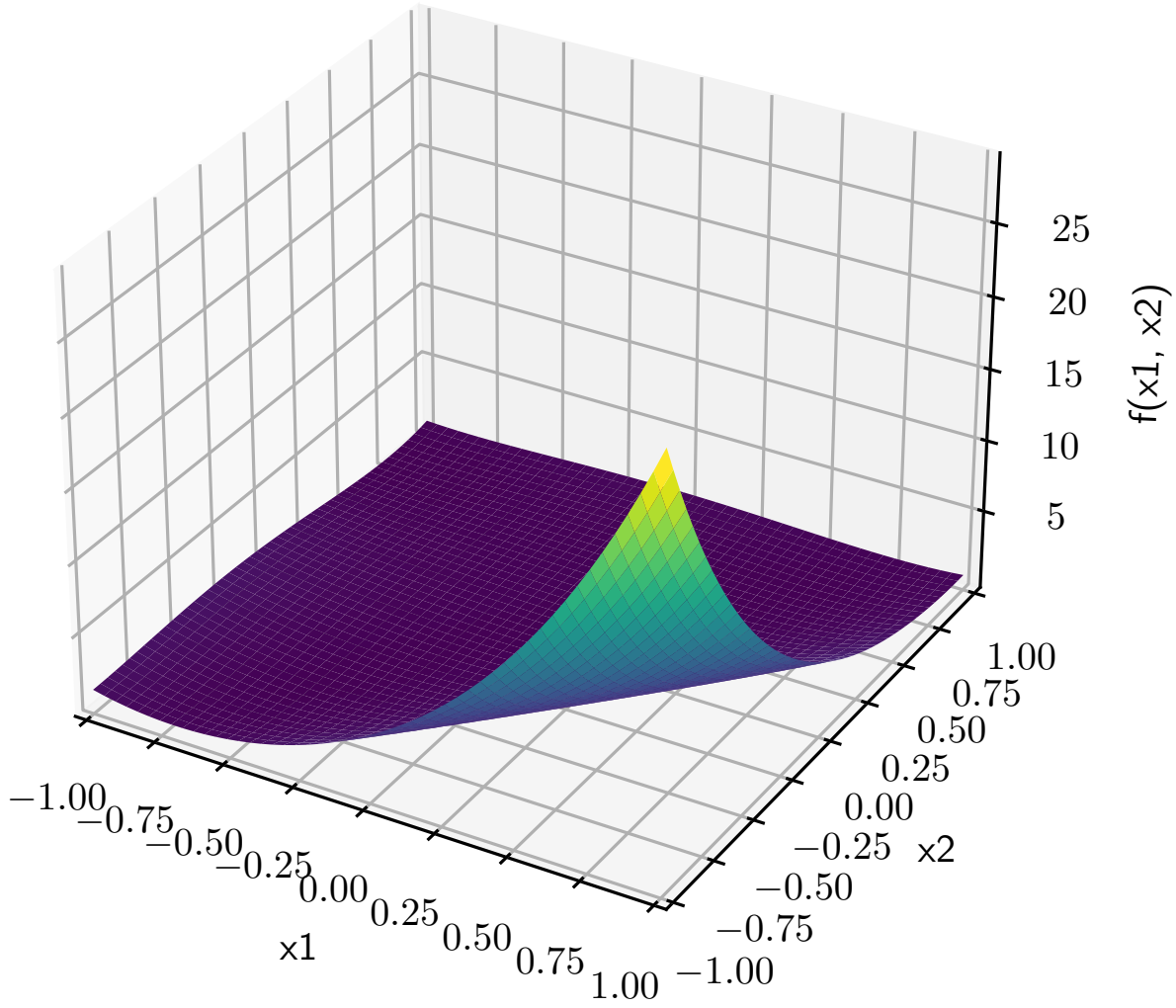
```
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1.,  1.        ,  1.     ,  1.        )]
 [(1.,  1.        ,  1.     ,  1.        )]
 [(1.,  1.        ,  1.     ,  1.        )]]
[[(1., -1.        ,  1.     , -1.        )]
 [(1., -1.        ,  1.     , -1.        )]
 [(1., -1.        ,  1.     , -1.        )]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1.,  0.35384615,  0.21875,  0.07740385)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1., -0.69230769, -1.     ,  0.69230769)]
 [(1.,  1.        ,  1.     ,  1.        )]
 [(1.,  1.        ,  1.     ,  1.        )]
 [(1.,  1.        ,  1.     ,  1.        )]
 [(1.,  1.        ,  1.     ,  1.        )]]
[[(0.  ,  0.  )]
 [(0.1 , -0.64)]
 [(0.1 , -0.64)]
 [(0.44, -0.25)]
 [(0.65,  0.  )]]
[[(0.  ,  0.  )]
 [(0.  ,  0.  )]
 [(0.  ,  0.  )]
 [(0.1 , -0.64)]
 [(0.1 , -0.64)]
 [(0.44, -0.25)]
 [(0.44, -0.25)]
 [(0.65,  0.  )]
 [(0.65,  0.  )]
 [(0.65,  0.  )]]
[[(0.  ,  0.  )]
 [(0.  ,  0.  )]
 [(0.  ,  0.  )]
 [(0.1 , -0.64)]
 [(0.1 , -0.64)]
 [(0.1 , -0.64)]
 [(0.1 , -0.64)]
 [(0.44, -0.25)]
 [(0.44, -0.25)]
 [(0.44, -0.25)]
 [(0.44, -0.25)]
 [(0.65,  0.  )]
 [(0.65,  0.  )]
 [(0.65,  0.  )]
 [(0.65,  0.  )]]
0.02339100346020754
0.0017326669229783377
0.0003654844290657436
[[ 0.74861985  0.98044926 -0.75670809 -1.07177279]
 [ 0.98044926  1.91707868 -1.35324338 -1.79899044]
 [-0.75670809 -1.35324338  1.26479632  1.44456691]
 [-1.07177279 -1.79899044  1.44456691  2.18090221]]
[[ 0.37865832  0.47260722 -0.3813544  -0.5030484 ]
```

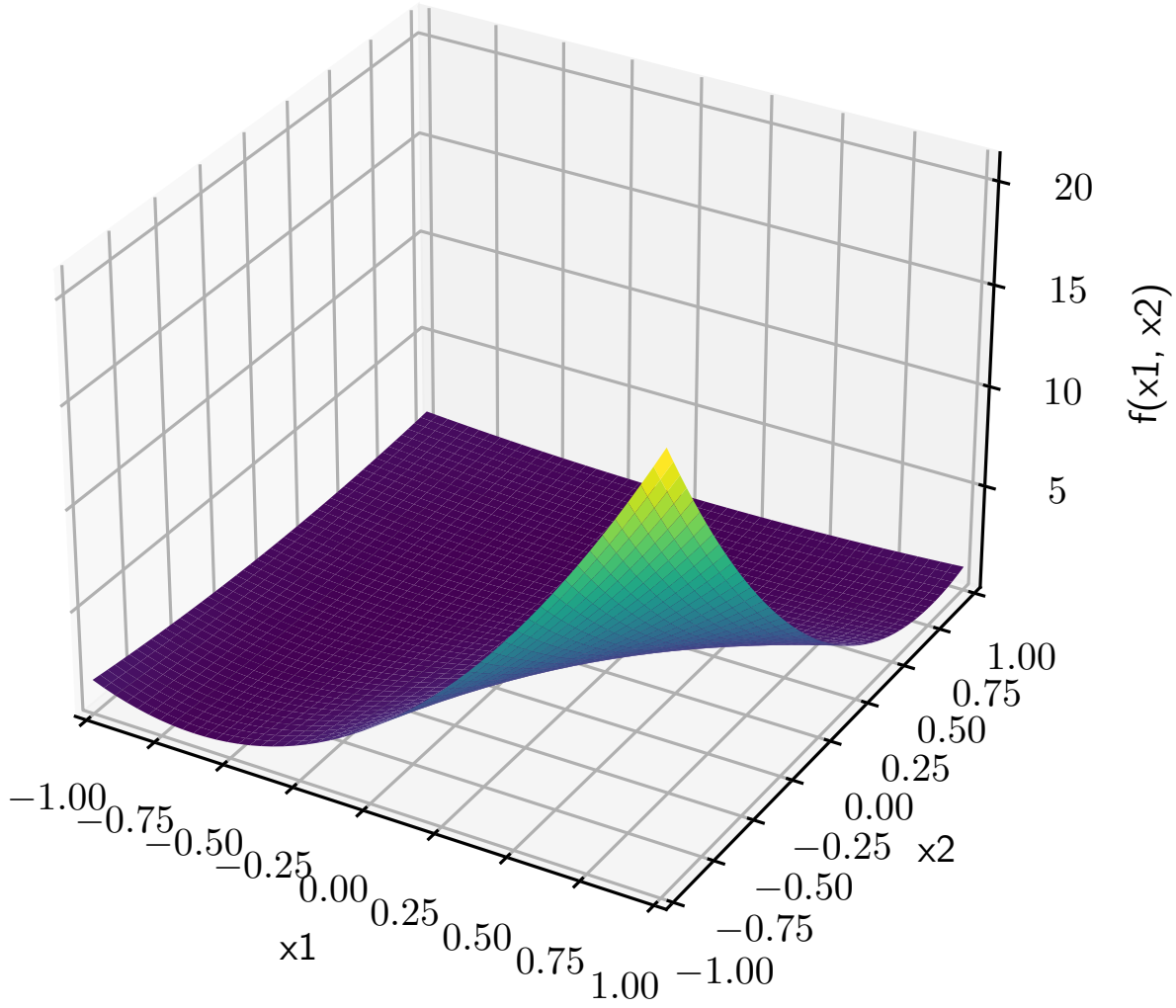
```
 [ 0.47260722  0.96467285 -0.59687193 -0.92531011]
 [-0.3813544  -0.59687193  0.55071714  0.62731311]
 [-0.5030484  -0.92531011  0.62731311  1.05261403]]
[[ 0.1913074   0.23160389 -0.18986132 -0.25790291]
 [ 0.23160389  0.5279036  -0.34525585 -0.47792811]
 [-0.18986132 -0.34525585  0.33424858  0.35072153]
 [-0.25790291 -0.47792811  0.35072153  0.57378595]]
[[6.11139706]]
[[2.94666234]]
[[1.62724553]]
0.02339100346020754
0.0017326669229783377
0.0003654844290657436
[3.74309926 5.91186227 5.11970777 6.46543995]
[3.78658319 6.95432088 4.45844249 7.33731197]
[2.86961105 4.97178983 3.74071165 5.13029201]
```

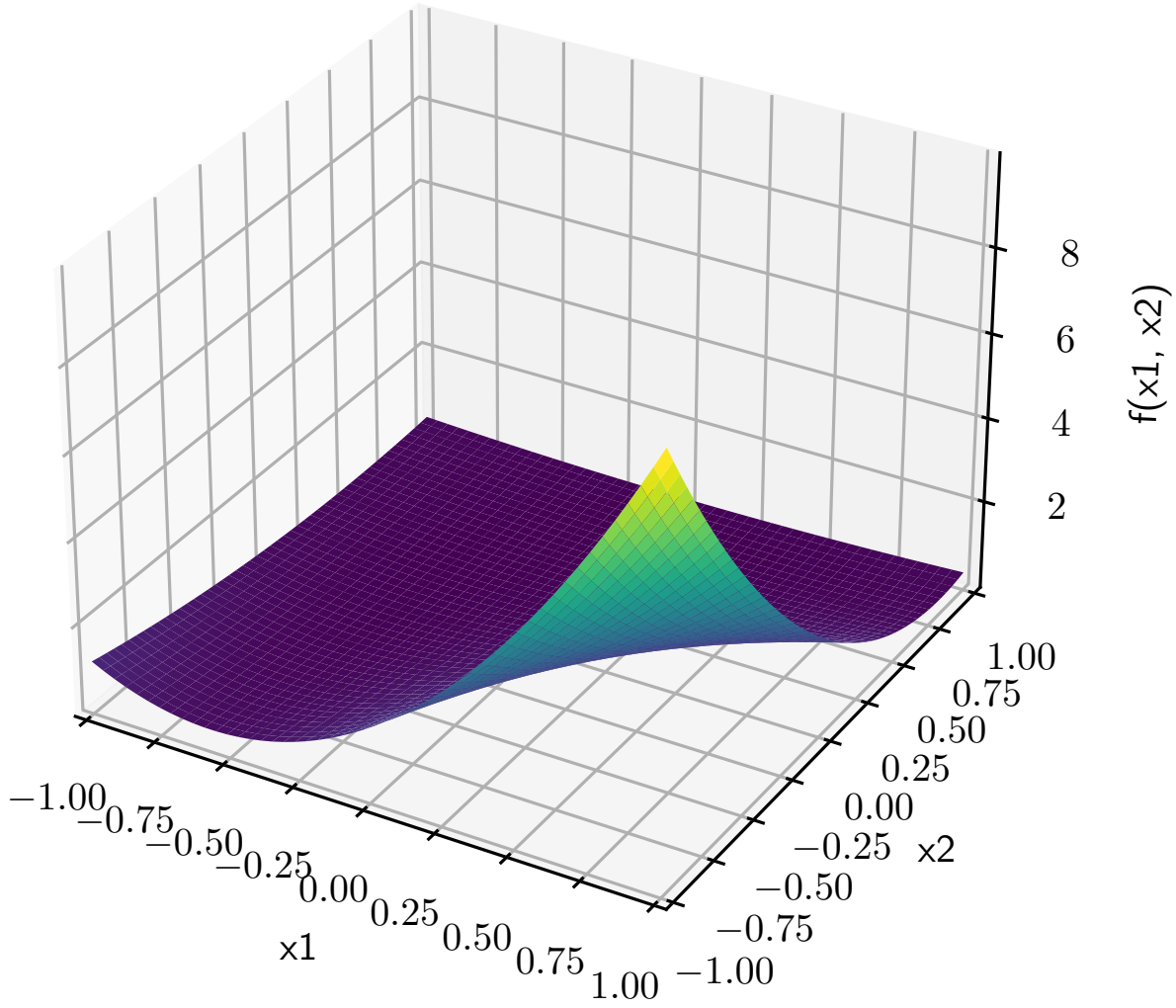


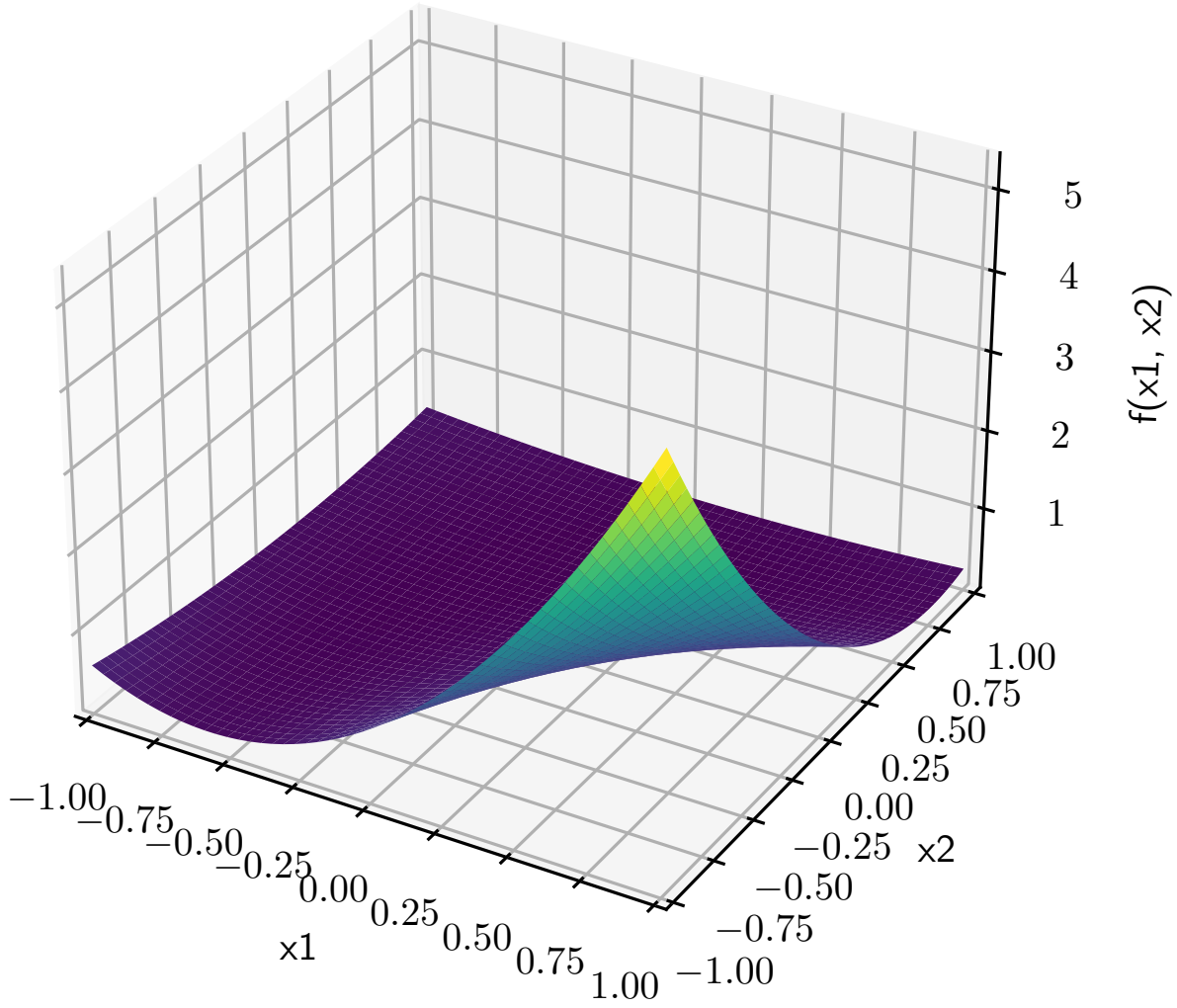

0.5
0.4
0.3
0.2
0.1
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1


6
5
4
3
2
1
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
x2
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1


25
20
15
10
5
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1


20
15
10
5
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1


8
6
4
2
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1


5
4
3
2
1
f(x1, x2)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
x2
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x1