# 【多変量正規分布 Multivariate Normal Distribution】

$n$ 次元ベクトル　$\boldsymbol{u} = (u_1, u_2, \cdots u_n)$

$$P(\boldsymbol{u}) = \prod_{i=1}^{n} \frac{1}{(\sqrt{2\pi})^n} \exp(-\frac{u_i^2}{2}) = \frac{1}{(\sqrt{2\pi})^n} \exp(-\sum_{i=1}^{n} \frac{u_i^2}{2}) = \frac{1}{(\sqrt{2\pi})^n} \exp(-\frac{1}{2}\boldsymbol{u}^T\boldsymbol{u})$$

正則一次変換 $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{u} + \boldsymbol{\mu}$ より,$\boldsymbol{u} = \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})$.

これを代入して,

$$\begin{aligned} P(\boldsymbol{x}) &= \frac{1}{(\sqrt{2\pi})^n} \exp[(-\frac{1}{2}(\boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}))^T \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})] = \frac{1}{(\sqrt{2\pi})^n} \exp[(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T (\boldsymbol{A}^T)^{-1} \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})] \\ &= \frac{1}{(\sqrt{2\pi})^n} \exp[(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T (\boldsymbol{A}\boldsymbol{A}^T)^{-1}(\boldsymbol{x} - \boldsymbol{\mu})] \qquad \because (\boldsymbol{A}^T)^{-1}(\boldsymbol{A})^{-1} = (\boldsymbol{A}\boldsymbol{A}^T)^{-1} \end{aligned}$$

ここで,$(\boldsymbol{A}\boldsymbol{A}^T)^{-1} = \sum^{-1}$ とおくと,

$$\|\boldsymbol{A}\| = \sqrt{\sum} = det\boldsymbol{A} \qquad \because \|\sum\| = \boldsymbol{A}\boldsymbol{A}^T = \|\boldsymbol{A}\|^2$$

面要素は,

$$du_1 du_2 du_3 \cdots du_n = \frac{1}{det\boldsymbol{A}} dx_1 dx_2 dx_3 \cdots dx_n$$

以上より,

$$\text{多変量正規分布} \quad \underline{P(\boldsymbol{x}) = \frac{1}{(\sqrt{2\pi})^n \sqrt{\sum}} \exp[(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x} - \boldsymbol{\mu})]}$$

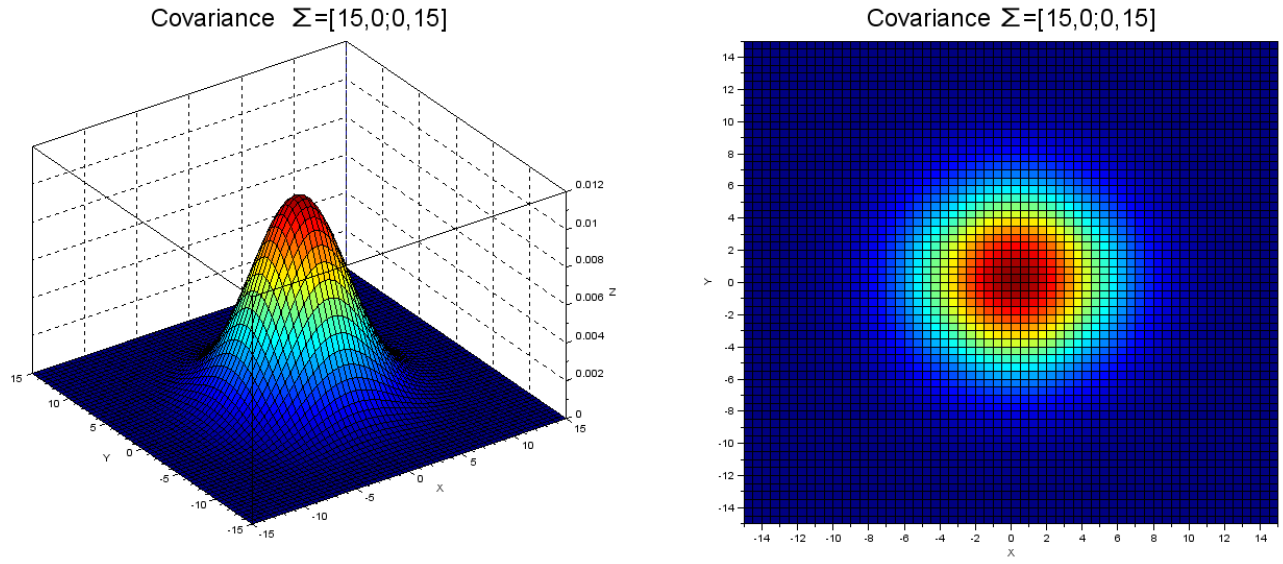


Figure 1: Scilab 実行結果

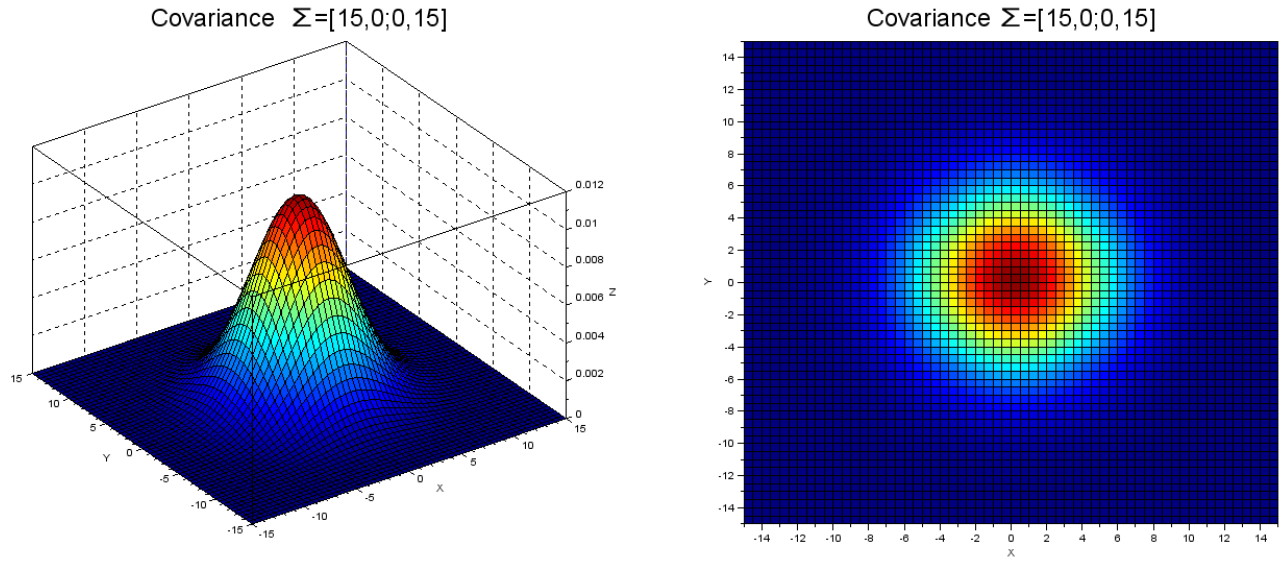


Figure 2: Scilab 実行結果

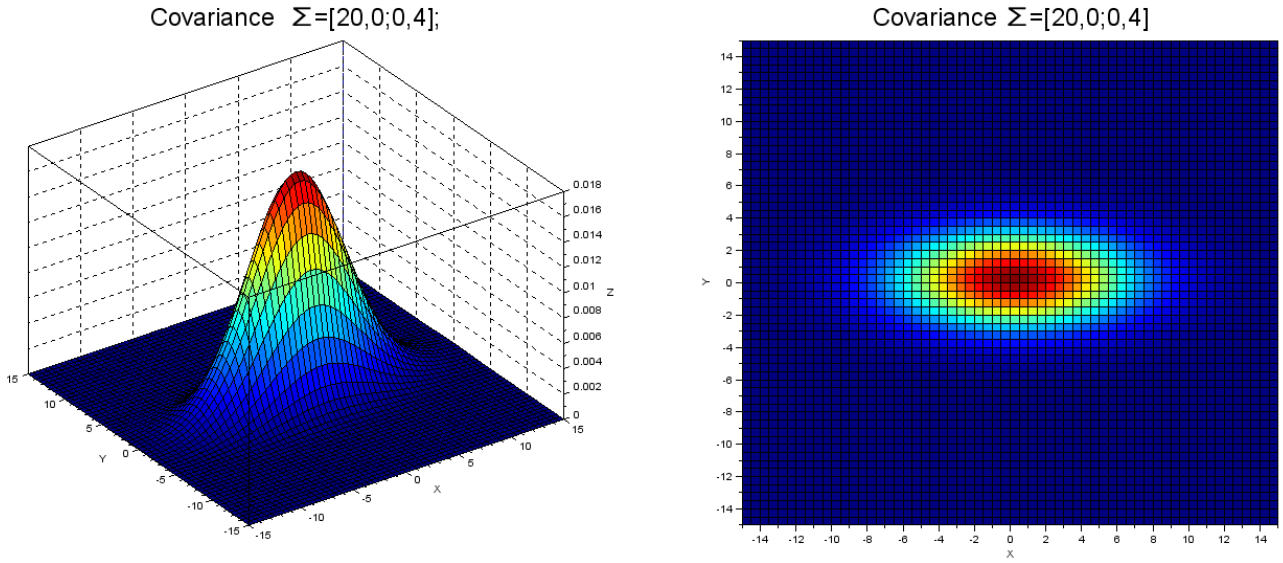


Figure 3: Scilab 実行結果

Source Code 1: Scilab

```
////////////////////////////////////////
//            多変量正規分布
//   Multivariate Normal Distribution
//
//                          M.Tsutsui
////////////////////////////////////////

clear;

funcprot(0);//2次元正規分布_関数
function[y]=fx(i,j)
        x=[i;j];
        y=1/((2*%pi)^(dim/2)*sqrt(det(siguma)))*%e^(-1/2*(x-myu)'*inv(siguma)*(x-myu));
```

```
endfunction

dim=2;//次元

myu=[0;0];//平均

siguma=[15,0;0,15];//共分散

loop_n=−15:0.5:15;//範囲指定

[d_size1,d_size2]=size(loop_n);//範囲_サイズチェック

fx_buf=[]; //関数ループ用バッファ

for(v=loop_n);
        for(u=loop_n);
                fx_buf=[fx_buf,fx(u,v)];
        end
end

fx_res=[];

for K=1:1:d_size2 //バッファ解体(正方行列へ)
        fx_res=[fx_res;fx_buf(d_size2∗K−(d_size2−1):d_size2∗K)];
end

[x1_m,x2_m]=meshgrid(loop_n,loop_n);//メッシュ用

subplot(1,2,1);
surf(x1_m,x2_m,fx_res);
set(gcf(),'color_map',jetcolormap(256));
xgrid();
title('3D View','fontsize',5);

subplot(1,2,2);
surf(x1_m,x2_m,fx_res);
set(gcf(),'color_map',jetcolormap(256));
fig_c=gca();
fig_c.view="2d";
xgrid();
title('2D View','fontsize',5);
```

Source Code 2: Python

```
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
# Module Name: Multivariate Normal Distribution
# Author: m_tsutsui
#−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
#Library_Import##############################
from numpy import∗
from mpl_toolkits.mplot3d import Axes3D #3Dim_Plot
import math, numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
#Library_Import_end#########################


def prob_d(i,j):
    x=matrix([[i],[j]])
    return float (1/((2∗pi)∗∗(dim/2)∗sqrt(np.linalg.det(matrix(siguma))))∗exp(−1/2∗(matrix(x)−matrix(myu)).T∗(matrix(
        siguma)).I∗(matrix(x)−matrix(myu))))

if __name__ == '__main__':
```

```
dim=2 #dimension

myu=matrix([[0],[0]]) #Expectation

siguma=matrix([[10,0],[0,10]]) #Covariance


loop_s=−15
loop_e=15
loop_step=0.5

loop_n=arange(loop_s,loop_e,loop_step) #scope

d_size2=np.size(loop_n,0) #size check

fx_buf=[] #buffer

for v in np.arange(loop_s,loop_e,loop_step):
    for u in np.arange(loop_s,loop_e,loop_step):
        fx_buf.append(prob_d(u,v))

fx_res=[]

for K in range(1,d_size2+1,1):
            fx_res.append(fx_buf[d_size2∗K−d_size2:d_size2∗K])

x1_m,x2_m=meshgrid(loop_n,loop_n)

fig = plt.figure(facecolor='w')
ax = Axes3D(fig)
ax.plot_surface(x1_m, x2_m, fx_res, rstride=1, cstride=1, cmap=cm.jet)

fig = plt.figure(facecolor='w')
plt.contourf(x1_m, x2_m, fx_res,100)
plt.colorbar()

plt.show()
```