Лабораторная работа №1 задание 3

Доскоч Роман вариант 9

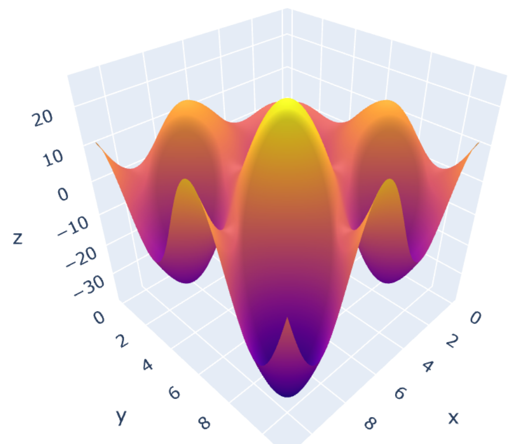
Приближение поверхностей

$$g(x, y) = (x + y)(\cos x + \cos y), x \in [0, 11], y \in [0, 11]$$

In [1]:
```
import plotly as pl
import numpy as np
import plotly.graph_objs as go
import plotly.express as px
from plotly.subplots import make_subplots
import matplotlib.pyplot as plt
import time
import pandas as pd
BicubickSpline, InterPolinomTime=[], []
```

In [2]:
```
f=lambda x,y: (x+y)*(np.cos(x)+np.cos(y))
```

In [3]:
```
x,y=np.meshgrid(np.linspace(0,11,100),np.linspace(0,11,100))
fig=go.Figure(go.Surface(x=x,y=y,z=f(x,y)))
fig.show()
```



Интерполяционные многочлены двух переменных функции $g(x, y)$

на прямоугольнике по сеткам 6×6, 12×12, 18×18 равноотстоящих узлов

$$P_n(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{n} z_{ij} \prod_{p \neq i} \frac{x - x_p}{x_i - x_p} \prod_{q \neq j} \frac{y - y_q}{y_j - y_q}$$

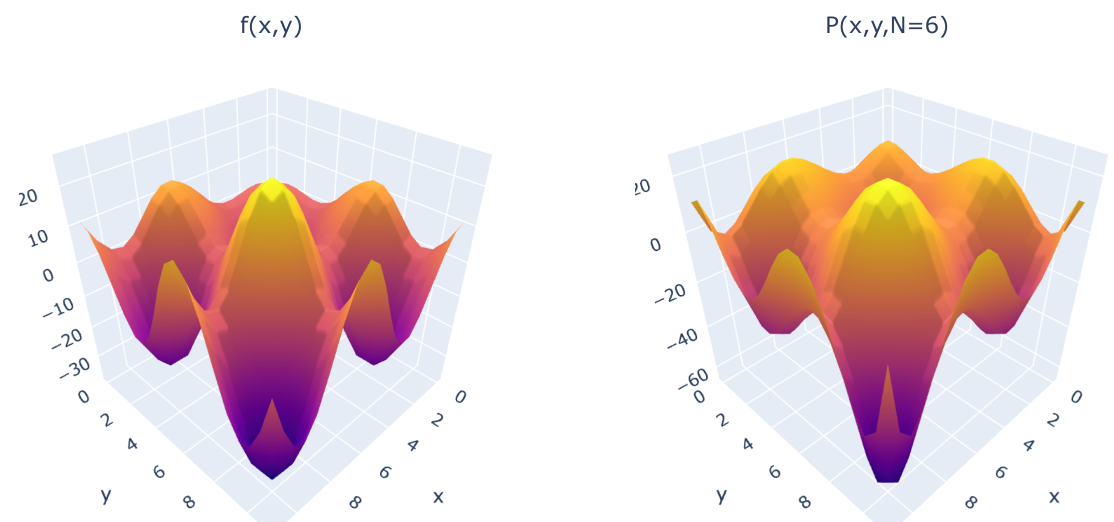
In [4]:
```
def P(x, y, N):
    start=time.time()
    X=np.linspace(0,11, N)
    Y=np.linspace(0,11, N)
    res=[]
    for i in range(len(x)):
        tmp=[]
        for j in range(len(x)):
            tmp.append(
                sum([sum([L(X,x[i][j],i_)*L(Y,y[i][j],j_)*f(X[i_],Y[j_]) for j_ in range(N)]) for i_ in range(N)]))
        res.append(np.array(tmp))
    InterPolinomTime.append(time.time()-start)
    return res

def L(X, x_, i):
    return np.prod([(x_-xj)/(X[i] - xj) for j, xj in enumerate(X) if i != j])
```

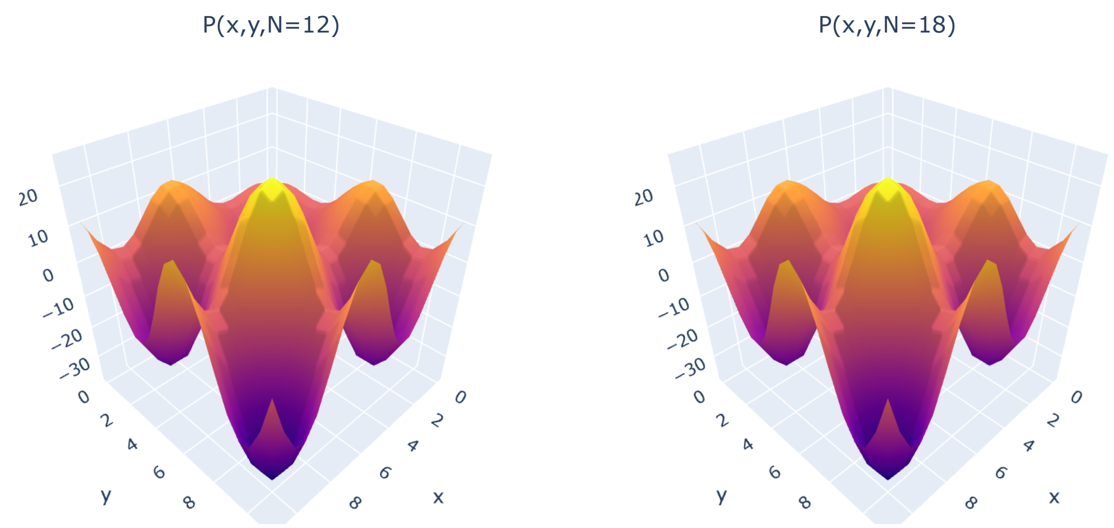
In [5]:
```
x,y=np.meshgrid(np.linspace(0,11,20),np.linspace(0,11,20))
P6,P12,P18=P(x,y,6),P(x,y,12),P(x,y,18)
fig = make_subplots(rows=1, cols=2,
                    subplot_titles=("f(x,y)", "P(x,y,N=6)"),
                    specs=[[{'type': 'surface'}, {'type': 'surface'}]])

fig.add_trace(go.Surface(x=x,y=y,z=f(x,y)),row=1, col=1)
fig.add_trace(go.Surface(x=x,y=y,z=P6),row=1, col=2)

fig.show()
```

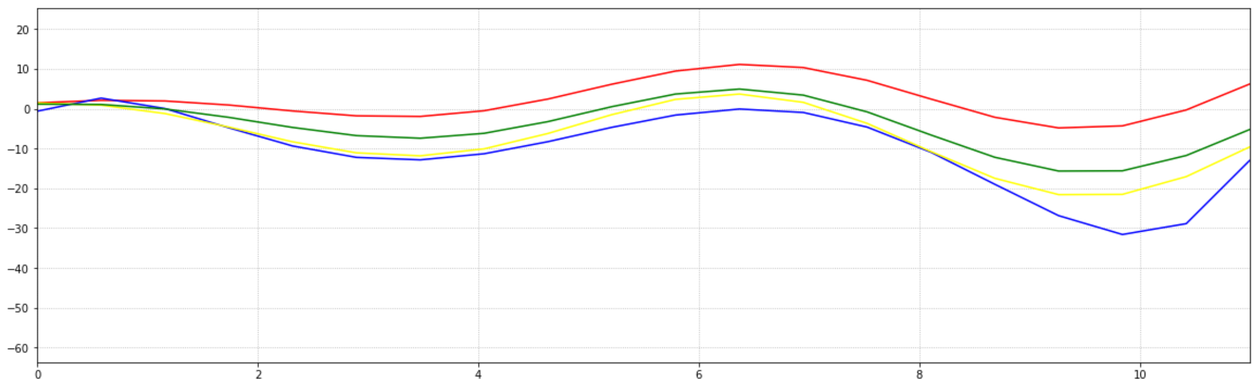


In [6]:
```
fig = make_subplots(rows=1, cols=2,
                    subplot_titles=("P(x,y,N=12)", "P(x,y,N=18)"),
                    specs=[[{'type': 'surface'}, {'type': 'surface'}]])
fig.add_trace(go.Surface(x=x,y=y,z=P12),row=1, col=1)
fig.add_trace(go.Surface(x=x,y=y,z=P18),row=1, col=2)
fig.show()
```



Срез графиков по плоскости zx красный-исходная функция
синий -график на 6 узлах
желтый -график на 12 узлах
зеленый -график на 18 узлах

In [7]:
```
x,y=np.meshgrid(np.linspace(0,11,20),np.linspace(0,11,20))
plt.figure(figsize=(20, 6))
plt.contour(x,f(x,y),y,[1], colors='red')
plt.contour(x,P6,y,[3],colors='blue')
plt.contour(x,P12,y,[4],colors='yellow')
plt.contour(x,P18,y,[2],colors='green')
plt.grid(ls=':')
```



Бикубический спалйн на прямоугольнике по сеткам 6 × 6, 12 × 12, 18 × 18 равноотстоящих узлов

$$S_i(x, y) = \alpha_i(y) + \beta_i(y)(x - x_i) + \frac{\gamma_i(y)}{2}(x - x_i)^2 + \frac{\delta_i(y)}{6}(x - x_i)^3$$

In [8]:
```
def method_vstr_prog(n, a, b, c, f, m=0):
        alpha, beta, mu, nu = ([0] * n for _ in range(4))
        alpha[0], beta[0] = b[0]/c[0], f[0]/c[0]
        mu[n-1],nu[n-1] = a[n-2]/c[n-1],f[n-1]/c[n-1]

        for i in range(n-2, m-1, -1):
            denom = c[i] - b[i] * mu[i+1]
            mu[i] = a[i-1] / denom
            nu[i] = (f[i] - b[i] * nu[i+1]) / denom

        y = [0] * n
        y[m] = (nu[m] - mu[m] * beta[m - 1]) / (1 - mu[m] * alpha[m-1])
        for i in range(m-1, -1, -1): y[i] = beta[i] - alpha[i] * y[i+1]
        for i in range(m, n-1): y[i+1] = nu[i+1] - mu[i+1] * y[i]

        return y
```

In [9]:
```
def coeffs(x, y):
    h = x[1] - x[0]
    b, g, d = [np.zeros(N) for _ in range(3)]
    c, e = [np.array([.5]*(N-3)) for _ in range(2)]
    b_ = 3*((y[2:] + y[:-2] - 2*y[1:-1])/h**2)

    g[1:-1] = method_vstr_prog(N-2, c, e, [2]*(N-2), b_)
    d[1:] = (g[1:] - g[:-1])/h
    b[1:] = (y[1:] - y[:-1])/h + (2*g[1:] + g[:-1]) / 6 * h
    return y, b, g, d
```

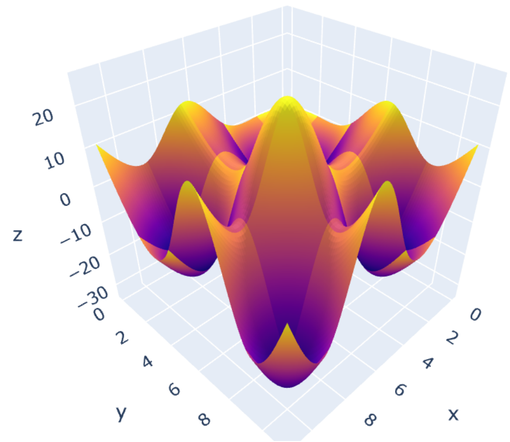
In [10]:
```
def S(yi, j, a, b, g, d):
    return a[j] + b[j] * (yi - Y[j]) + g[j]/2 * (yi - Y[j])**2  + d[j]/6 * (yi - Y[j])**3
```

In [11]:
```
def Sij(i, j, x, y):
    res=[]
    coeff=[coeffs(Y, C[:,k][:,i]) for k in range(4)]
    for xi, yj in zip(x, y):
        a, b, g, d=[S(yj, j, *coeff[k]) for k in range(4)]
        res.append(a + b * (xi - X[i]) + g/2 *(xi - X[i])**2 + d/6 * (xi - X[i])**3)
    return res
```

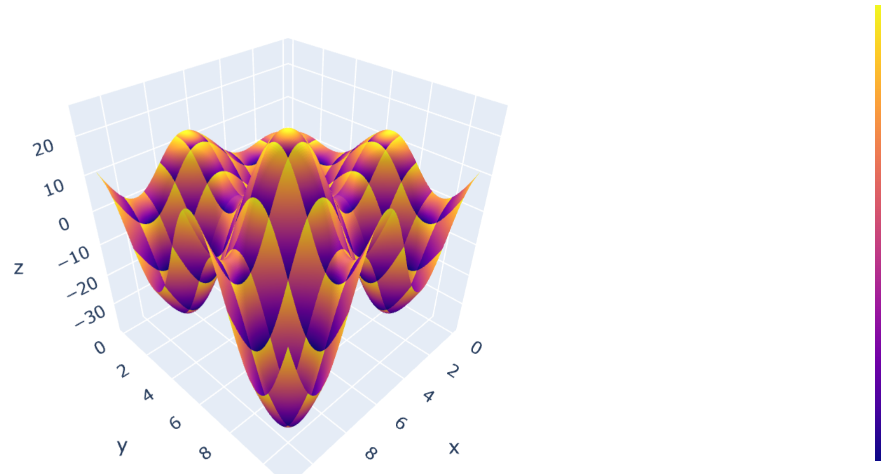
In [12]:
```
def buid_plot():
    data=[]
    start = time.time()
    for i in range(1,N):
        for j in range(1,N):
            x, y = np.meshgrid(np.linspace(X[i-1], X[i], 20),np.linspace(Y[j-1], Y[j], 20))
            data.append(go.Surface(x=x, y=y, z=Sij(i, j, x, y)))

    BicubickSpline.append(time.time()-start)
    return data
```

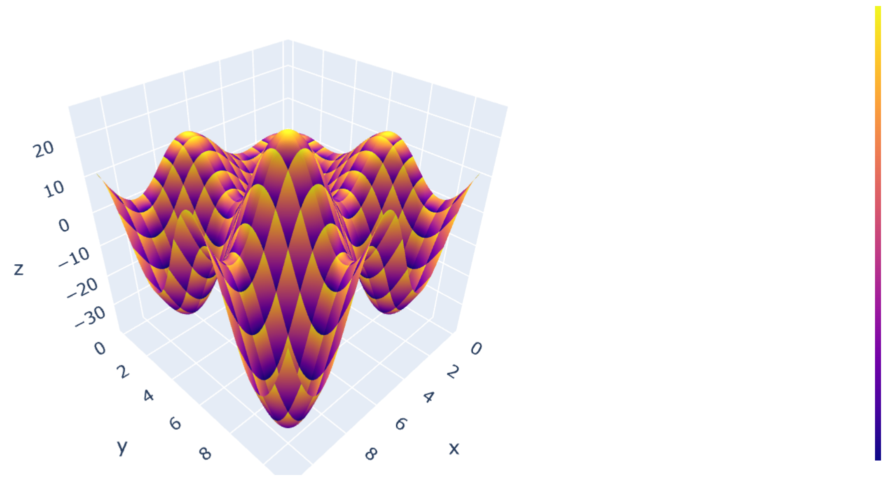
In [13]:
```
N=6
X, Y= np.linspace(0,11,N), np.linspace(0,11,N)
C = np.array([coeffs(X, f(X,y)) for y in Y])
go.Figure(buid_plot()).show()
```



In [14]:
```
N=12
X, Y= np.linspace(0,11,N), np.linspace(0,11,N)
C = np.array([coeffs(X, f(X,y)) for y in Y])
go.Figure(buid_plot()).show()
```



In [15]:
```
N=18
X, Y= np.linspace(0,11,N), np.linspace(0,11,N)
C = np.array([coeffs(X, f(X,y)) for y in Y])
go.Figure(buid_plot()).show()
```



In [16]:
```
df = pd.DataFrame([['Лагранж 2 переменных',*InterPolinomTime],
                  ['БиКубический Сплайн',*BicubickSpline]]
                 , columns=['Тип', '6','12','18'])
df
```

Out[16]:

	Тип	6	12	18
0	Лагранж 2 переменных	0.300869	1.318034	3.507402
1	БиКубический Сплайн	0.048999	0.237001	0.541998

Видно что бикубический сплайн выигрывает по времени у ИМ 2-х перменных
Все из-за сложности алгоритма
ИМ-$O(n^3)$
БиКуб Сплайн - $O(4n^2)$