

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN
DE AREQUIPA



ESCUELA PROFESIONAL DE CIENCIA DE LA
COMPUTACIÓN

Física Computacional

ECUACIÓN DE ONDA
(PRACTICA 07)

Villanueva Sanchez, Fernando Thomas

supervised by
Professor Edwin Llamoca Requena

15 de noviembre del 2020

Índice

1. Solución aproximada de la ecuación de Onda	2
1.1. Analice el mismo problema, cambiando h y k y verifique sus resultados.	2
1.2. Verifique que si no cumple $r \leq 1$, como se verán los resultados ?.	7
2. Problema Desafío	8
2.1. Use $a = 1$, $b = 1$, $v = 1$, $g(x) = 0$	8
2.2. Haga una secuencia de evolución paso a paso y que se grafique en forma independiente, ejemplo. Utilice la instrucción subplot(10,10,x). Subplot(10,10,1) para $j = 1$, Subplot(10,10,2) para $j = 2$, etc.	9

1. Solución aproximada de la ecuación de Onda

- 1.1. Analice el mismo problema, cambiando h y k y verifique sus resultados.

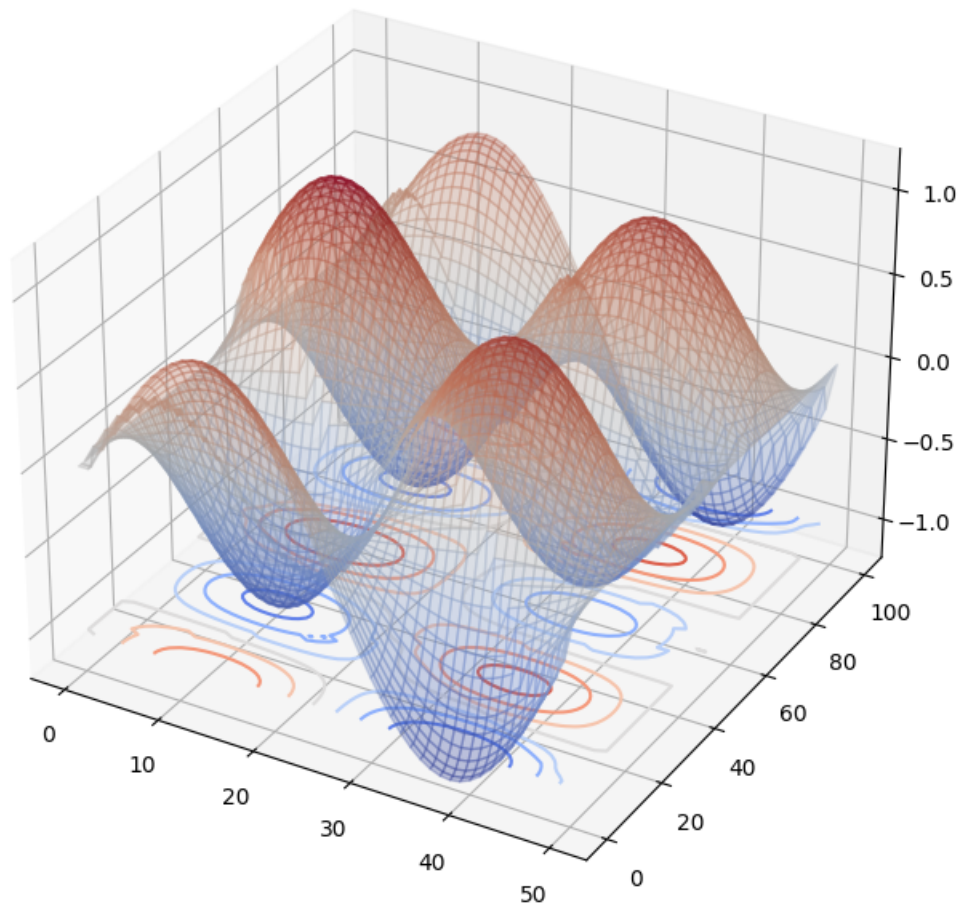


Figura 1: parámetros para la función `wave('f','g',1,1,2,0.02,0.01)`

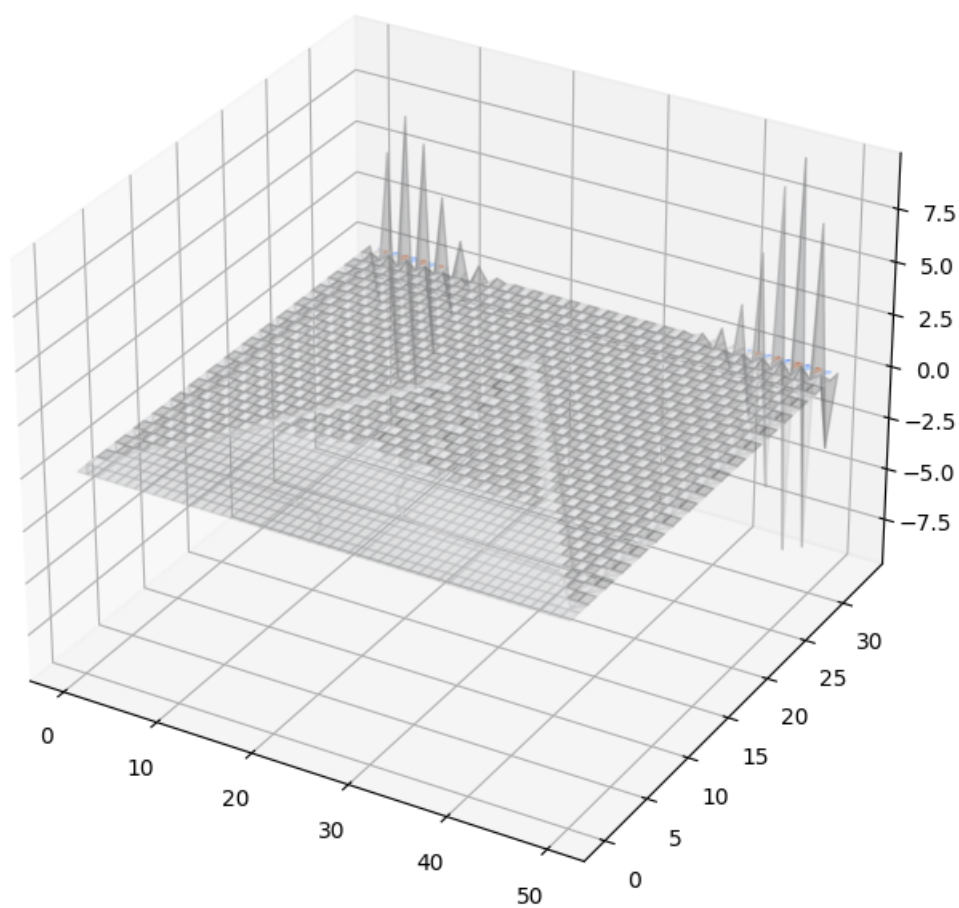


Figura 2: parámetros para la función $\text{wave}('f','g',1,1,2,0.02,0.03)$

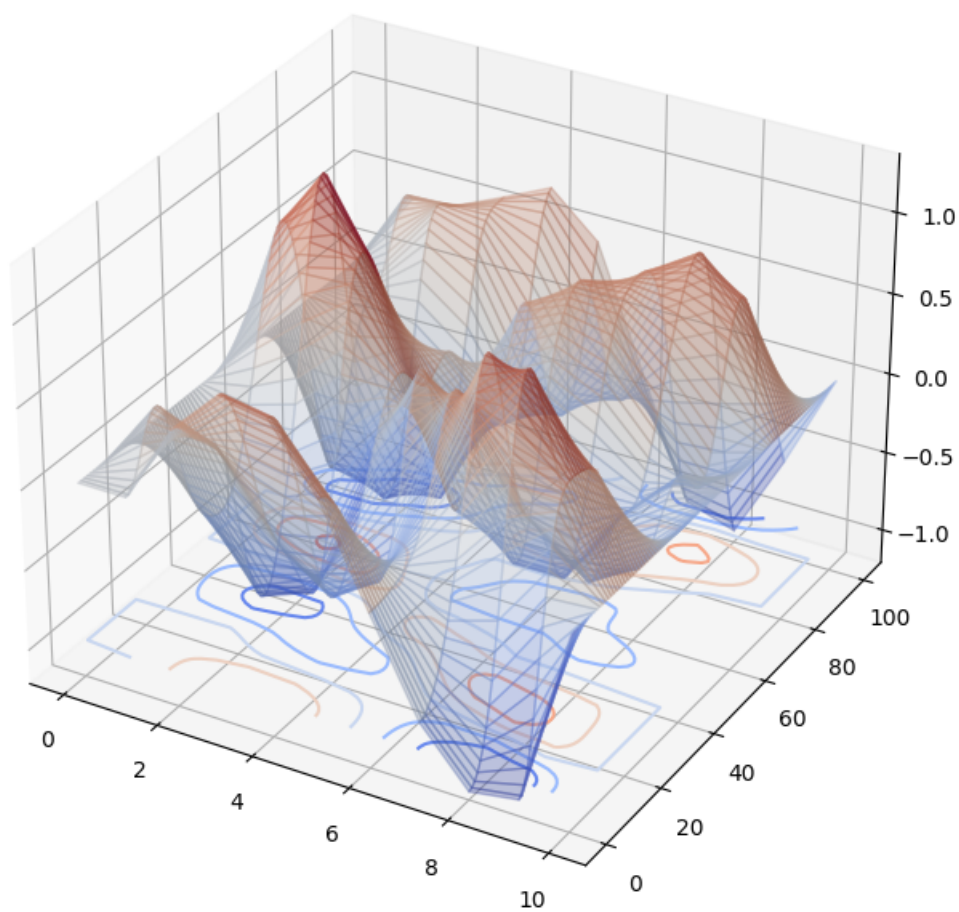


Figura 3: parámetros para la función `wave('f','g',1,1,2,0.1,0.01)`

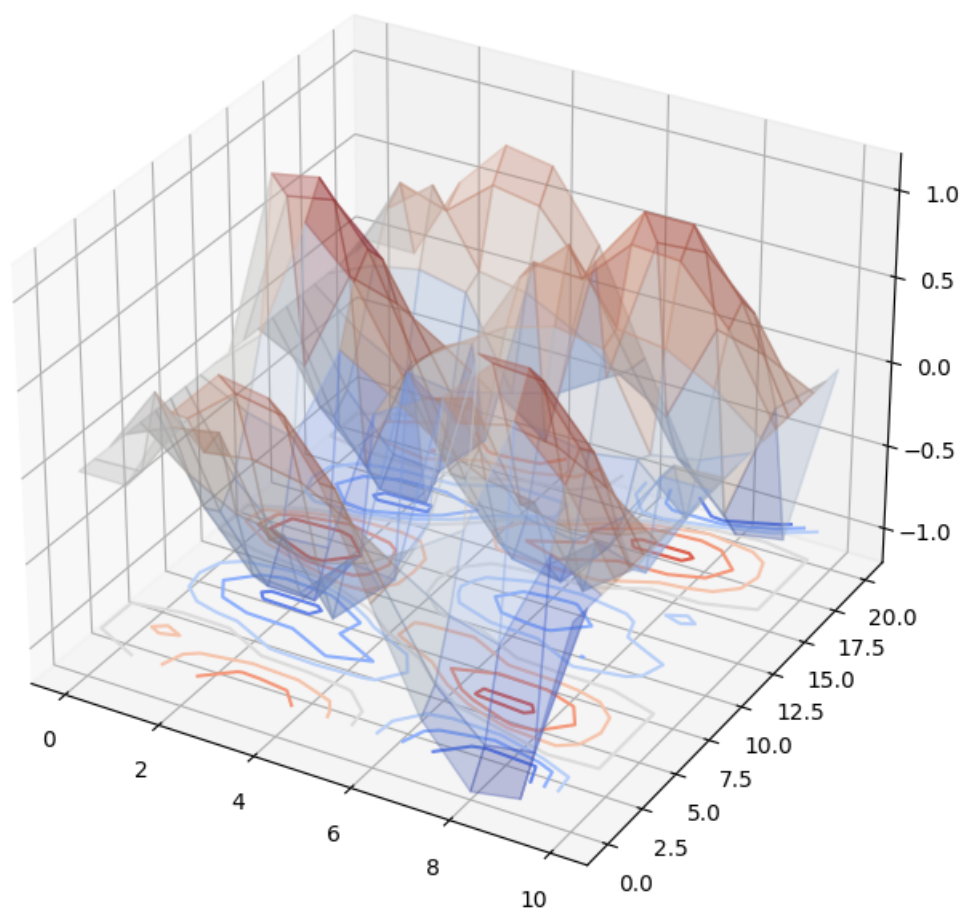


Figura 4: parámetros para la función `wave('f','g',1,1,2,0.1,0.05)`

```

import numpy as np
from matplotlib.colors import Normalize
from matplotlib import cm
import matplotlib.pyplot as plt
import matplotlib.pylab as p
import imageio

def feval(funcName, *args):
    return eval(funcName)(*args)

def meshgrid_of(A):
    x = range(np.shape(A)[1])
    y = range(np.shape(A)[0])
    xx, yy = p.meshgrid(x,y)
    return xx, yy

def surf(Z, colormap=cm.coolwarm):
    X, Y = meshgrid_of(Z)
    C = Z
    fig = plt.figure()
    scalarMap = cm.ScalarMappable(norm=Normalize(vmin=C.min(),
                                                    vmax=C.max()), cmap=colormap)
    C_colored = scalarMap.to_rgba(C, 0.3)
    ax = fig.gca(projection='3d')
    surf = ax.plot_surface(X, Y, Z, rstride=1, cstride=1,
                           facecolors=C_colored)
    cset = ax.contour(X,Y,Z, zdir='z', offset=-1, cmap=cm.coolwarm)

    plt.show()
    return surf

f = lambda x: x**2-x + np.sin(2*np.pi*x)
g = lambda x: 0

def wave(f, g, a, b, v, h, k):
    row = int(a/h+1)
    col = int(b/k+1)
    r = v*k/h
    r1 = r**2
    r2 = r**2/2
    s1 = 1-r**2
    s2 = 2*(1-r**2)

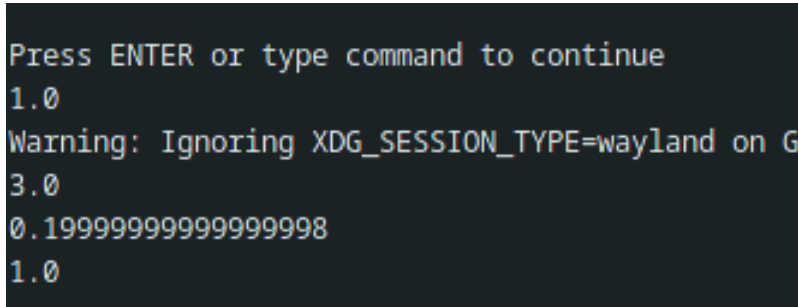
    U = np.zeros((row,col))
    for i in range(1,row-1):
        U[i,0] = feval(f, (i-1)*h)
        U[i,1] = s1*feval(f,h*(i-1)) + k*feval(g,h*(i-1)) +
                r2*(feval(f,h*i) + feval(f,h*(i-2)))

    for j in range(1,col-1):
        for k in range(1,row-1):
            U[k,j+1] = s2*U[k,j]+r1*(U[k-1,j]+U[k+1,j])-U[k,j-1]
    surf(np.transpose(U))

wave('f','g',1,1,2,0.02,0.01)
wave('f','g',1,1,2,0.02,0.03)
wave('f','g',1,1,2,0.1,0.01)
wave('f','g',1,1,2,0.1,0.05)

```

- 1.2. Verifique que si no cumple $r \leq 1$, como se verán los resultados ?.

A terminal window with a dark background and light-colored text. The text is as follows:

```
Press ENTER or type command to continue
1.0
Warning: Ignoring XDG_SESSION_TYPE=wayland on G
3.0
0.19999999999999998
1.0
```

Figura 5: La figura muestra la salida de la ejecución del código mostrado arriba. La imagen de salida es 2 y como se muestra aquí en la línea bajo el *warning* se aprecia que el r es 3.0 y es un dato que no debe mostrarse nunca, es decir un gráfico con propiedades físicas coherentes debe tener r menos o igual a 1.

2. Problema Desafío

2.1. Use $a = 1$, $b = 1$, $v = 1$, $g(x) = 0$.

$$f(x) = \begin{cases} 2x & 0 \leq x \leq 1/2 \\ 2 - 2x & 1/2 \leq x \leq 1 \end{cases}$$

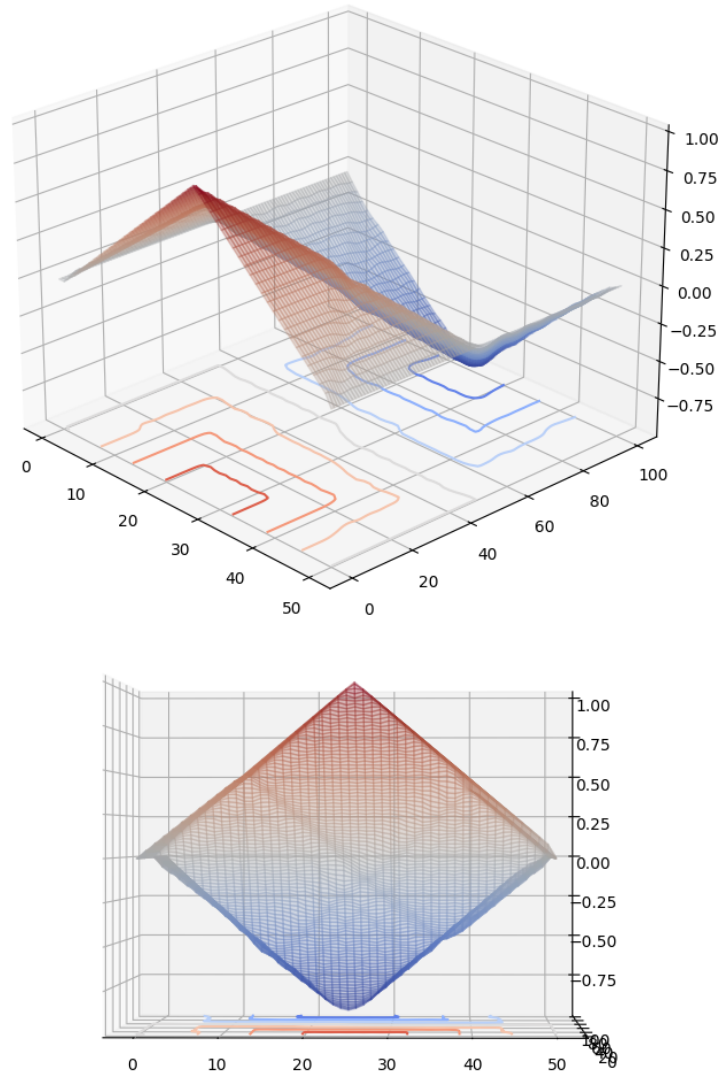


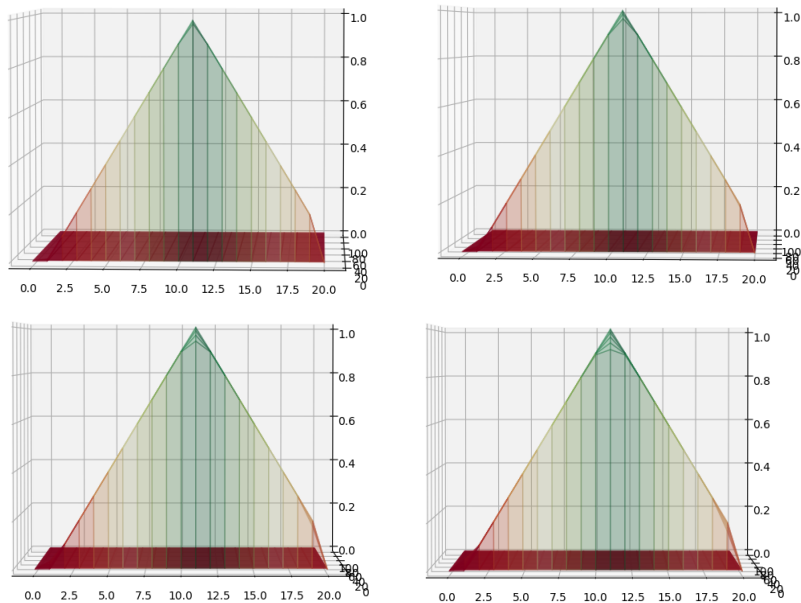
Figura 6: Imagen de salida del desafío, la imagen inferior se muestra desde el plano x,y

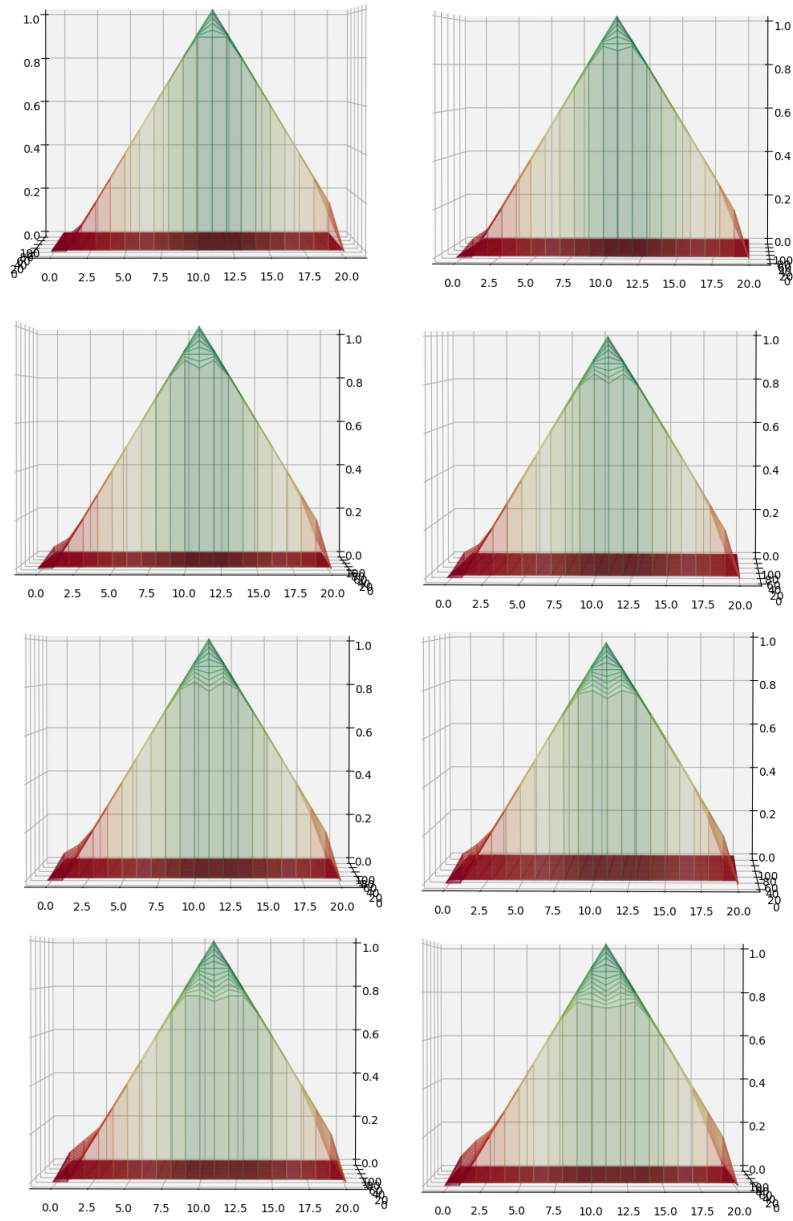
El código es el mismo, la diferencia que se implemento es la función lambda f, pues esta tiene que cumplir con las características de *even - add* de este problema. Esto se aprecia en el pedazo de código que se cambio.

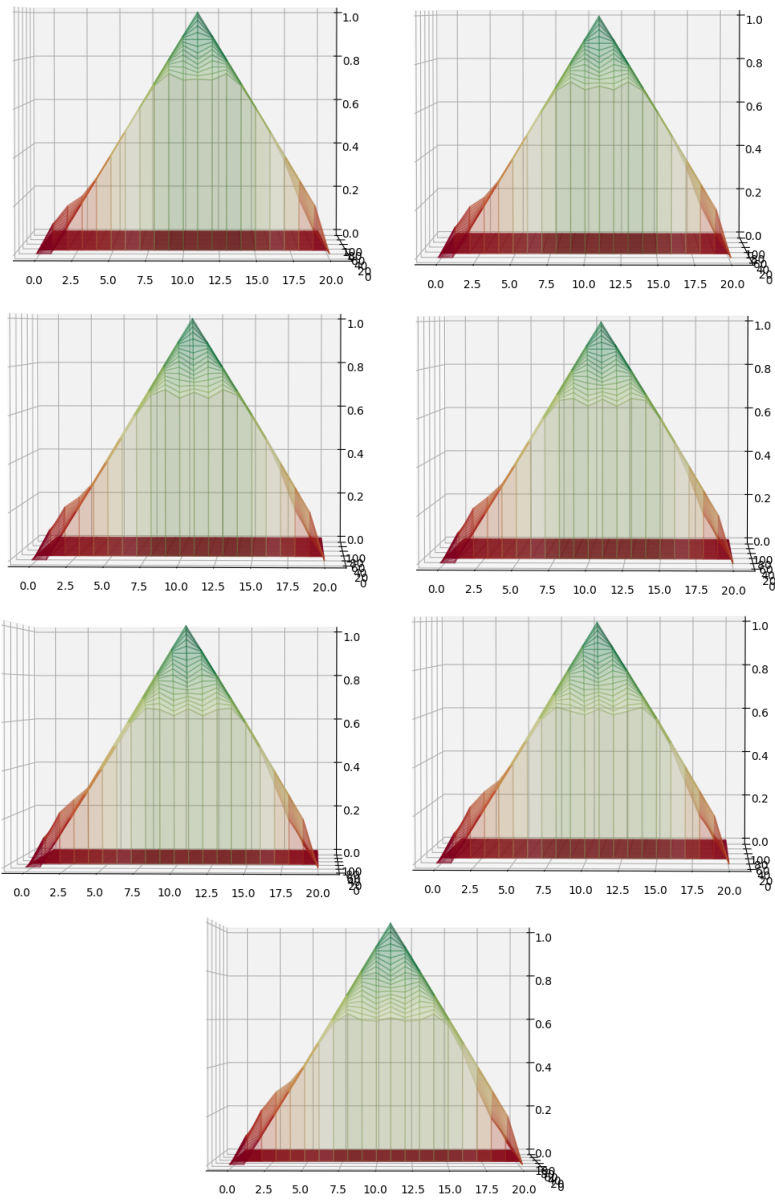
```
#
#funcion lambda para el problema desafio 1
#
f = lambda x: 2*x if 0<=x<=0.5 else (2-2*x if 0.5<=x<=1 else 0)
g = lambda x: 0
#
```

```
#la llamada a la fucion wave se hace con los siguiintes parametros:
#main
#
wave('f','g',1,1,1,0.02,0.01)
```

2.2. Haga una secuencia de evolución paso a paso y que se grafique en forma independiente, ejemplo. Utilice la instrucción subplot(10,10,x). Subplot(10,10,1) para $j = 1$, Subplot(10,10,2) para $j = 2$, etc.







```

import numpy as np
from matplotlib.colors import Normalize
from matplotlib import cm
import matplotlib.pyplot as plt
import matplotlib.pylab as p
import imageio

def feval(funcName, *args):
    return eval(funcName)(*args)

def meshgrid_of(A):
    x = range(np.shape(A)[1])
    y = range(np.shape(A)[0])
    xx, yy = p.meshgrid(x,y)
    return xx, yy

def surf(Z, colormap=cm.RdYlGn):
    X, Y = meshgrid_of(Z)
    C = Z

    fig = plt.figure()
    scalarMap = cm.ScalarMappable(norm=Normalize(vmin=C.min(), vmax=C.max()), cm
    C_colored = scalarMap.to_rgba(C,0.3)

    ax = fig.gca(projection='3d')
    surf = ax.plot_surface(X, Y, Z,rstride=1,cstride=1,facecolors=C_colored)
    cset = ax.contour(X,Y,Z,zdir='z',offset=-1,cmap=cm.RdYlGn)
    #fig.colorbar(surf,shrink=0.5,aspect=5)

    plt.show()
    return surf

#f = lambda x: x**2-x + np.sin(2*np.pi*x)
#g = lambda x: 0
f = lambda x: 2*x if 0<=x<=0.5 else (2-2*x if 0.5<=x<=1 else 0)
g = lambda x: 0

def wave(f, g, a, b, v, h, k):
    row = int(a/h+1)
    col = int(b/k+1)
    r = v*k/h
    print(r)
    r1 = r**2
    r2 = r**2/2
    s1 = 1-r**2
    s2 = 2*(1-r**2)

    U = np.zeros((row,col),np.float)
    #print(U)
    #print(row,col)
    #arr=[]
    for i in range(1,row-1):
        U[i,0] = feval(f,(i-1)*h)
        U[i,1] = s1*feval(f,h*(i-1)) + k*feval(g,h*(i-1)) + r2*(feval(f,h*i) + f

    #print(U)

    for j in range(1,col-1):
        for k in range(1,row-1):
            #U[k][j+1] = s2 * U[k][j] + r1 * (U[k-1][j] + U[k+1][j]) - U[k][j-2]
            U[k,j+1] = s2*U[k,j]+r1*(U[k-1,j]+U[k+1,j])-U[k,j-1]

```

```
surf(np.transpose(U))

#main
"""
r=0.9
h=0.05
k=0.01
v=r*h/k
"""
wave('f','g',1,1,1,0.05,0.01)
```

Referencias

- [1] Repositorio en Github que contiene todos los ejercicios implementados en python.
- [2] Enlace hacia el codigo LaTeX en Overleaf.