FNU ANIRUDH

S670

ASSIGNMENT I

**z = f(x; y) = cos(x)sin(y)**

**1. Create a function in R which will produce either an image or perspective plot of the bi-variate function f(x; y) over a speci\_ed x and y range. Hint: You can use the functionexpand.grid(a; b) in R to create a grid of (x; y) values for which you can create the plot**.

plot\_image=function(x,y)

{

value\_range=expand.grid(x,y)

z\_matrix=matrix(cos(value\_range$Var1)\*sin(value\_range$Var2),nrow=length(x))

image(x,y,z\_matrix)

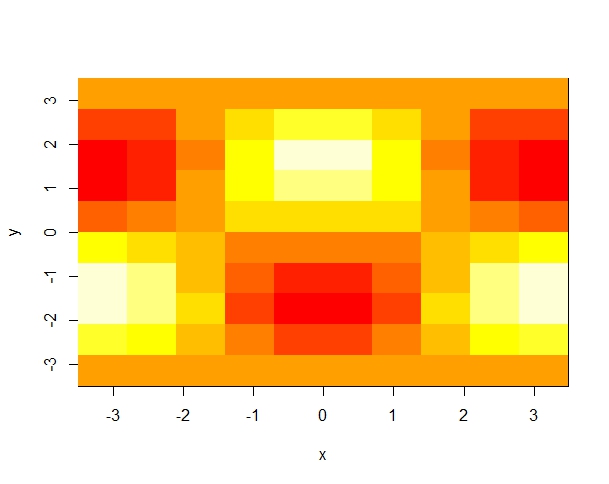
}

x=seq(-pi,pi,length = 10)

y=x

plot\_image(x,y)

**Output:-**



**2. Now create some R code which would allow the user to overlay some arrows on your image plot which point in the direction of the gradient of f(x; y) for a sequence of test points f(x1; y1); (x2; y2); : : : (xn; yn)g For this exercise you can use the deriv() function in R along with the expand.grid(a; b) function.**

#Function to plot a bivarient function and display the gradient

#for required points.

#List of Arguments:-

#1.expr:-Expression to be plotted.

#2.plotx:-X values to be plotted

#3.ploty:-Y values to be plotted

#4.arrowsx:-Values of x for which the gradient is to be plotted

#5.arrowsy:-values of y for which the gradient is to be plotted

plot\_image = function(expr,plotx, ploty,arrowsx, arrowsy)

{

# the Derivative function

f = deriv(expr, c("x", "y"), function(x, y){})

# Grid values of X and Y

grid = expand.grid(plotx, ploty)

# get the values of function for given X and Y

z = f(grid[,1], grid[,2])

# Matrix Form

mat = matrix(z, nrow = length(plotx), ncol = length(ploty))

#Image Plot

image(plotx, ploty, mat, xlab = "X-AXIS", ylab = "Y-AXIS",

main = expr)

#Contour plot on the image plot itself

contour(plotx, ploty, mat, add = TRUE)

# grid values to plot the gradient

arrowsgrid = expand.grid(arrowsx, arrowsy)

# call the func passing the arrow grid value

G = f(arrowsgrid[,1],arrowsgrid[,2])

# get the gradient values

grad = attr(G,"gradient")

grad\_tip = arrowsgrid + grad

# get the X and Y gradient

xgrad = grad[,1]

ygrad = grad[,2]

# Add Arrows to the plot

arrows(arrowsgrid[,1], arrowsgrid[,2], grad\_tip[, 1],

grad\_tip[,2], lty = 5)

}

# define range for X

x = seq(-2\*pi, 2\*pi, length.out = 20)

# make Y equal to X

y = x

# define the X range values for plotting arrows

arrows\_x = seq(-1.5\*pi,1.5\*pi, length.out = 10)

# make Y range equal to X range

arrows\_y = arrows\_x

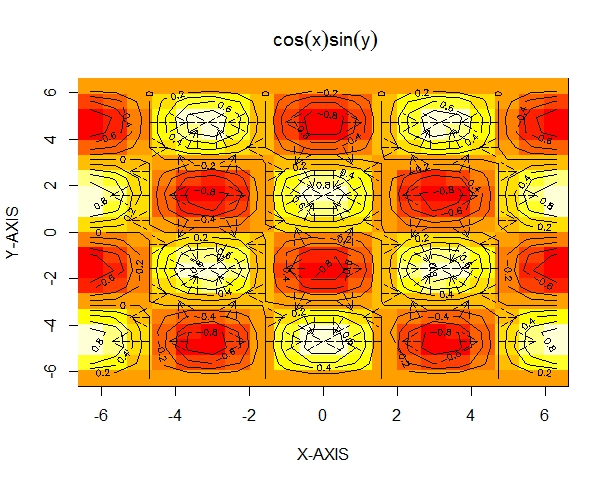
# Expression

bivarient\_expression = ~cos(x)\*sin(y)

# Call Function

plot\_image(bivarient\_expression, x, x, arrows\_x, arrows\_x)

**Output:-**



**3. For the next two problems use Data from UREDA page 31, problem 2(b), 2(c).**

**(a) Give stem-and-leaf displays for each of those batches of data. Choose one dataset to present your display by hand using two lines per stem (justify whether this is suggested by the rules), and the other dataset using R function using di\_erent values for \scale".**

**(b) Briefly describe the distributions from your displays (shape, symmetry, outliers, etc).**

a=c(0.12,0.15,0.15,0.10,0.13,0.15,0.14,

0.08,0.11,0.09,0.14,0.09,0.13,0.14,

0.12,0.16,0.15,0.13,0.12,0.12,0.09)

b=c(88,66,71,63,101,55,76,

49,63,38,91,79,41,36,

73,55,42,49,50,90,51)

stem(a,scale=2)

stem(a,scale=3)

stem(a,scale=4)

**Output:-**

The decimal point is 2 digit(s) to the left of the |

8 | 0

9 | 000

10 | 0

11 | 0

12 | 0000

13 | 000

14 | 000

15 | 0000

16 | 0

> stem(a,scale=3)

The decimal point is 2 digit(s) to the left of the |

8 | 0

9 | 000

10 | 0

11 | 0

12 | 0000

13 | 000

14 | 000

15 | 0000

16 | 0

> stem(a,scale=4)

The decimal point is 2 digit(s) to the left of the |

8 | 0

8 |

9 | 000

9 |

10 | 0

10 |

11 | 0

11 |

12 | 0000

12 |

13 | 000

13 |

14 | 000

14 |

15 | 0000

15 |

16 | 0

(b) leaf unit: 1

n: 21

3. | 68

4\*| 12

4. | 99

5\*| 01

5. | 55

6\*| 33

6. | 6

7\*| 13

7. | 69

8\*|

8. | 8

9\*| 01

9. |

10\*|

**4. Generate 1000 data from Gamma distributions (R: rgamma()) with two di\_erent sets of parameter values picked by you. For each of the generated datasets:-**

**(a) Produce a plot of the exact density function using the curve() function in R.**

**(b) Produce QQ plots of the generated data and interpret. Your interpretation might include symmetry, shape, heavy/light tail(s), location of the median, etc.**

**(c) Produce histogram plots of the data.**

**(d) Produce non-parametric density plots using the density() function for three different kernel functions. Which kernel function do you think best represents the data?**

alpha = 3

beta = 2

# Density function

par(mfrow = c(2, 3))

curve((((beta^alpha)/gamma(alpha))\*x^(alpha - 1)\*exp(-beta\*x)), 0, 20)

# Data generated from RGamma

a = rgamma(1000, shape = alpha,rate = beta)

# QQ Plot for a

qqnorm(a)

# Histogram

hist(a)

plot(density(a, kernel = c("gaussian")))

plot(density(a, kernel = c("rectangular")))

plot(density(a, kernel = c("cosine")))

**Output:-**

