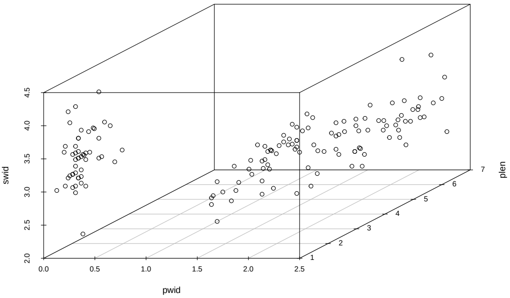
Lecture 17. (Interactive) 3D Plots

R and Data Visualization

BIG2006, Hanyang University, Fall 2022

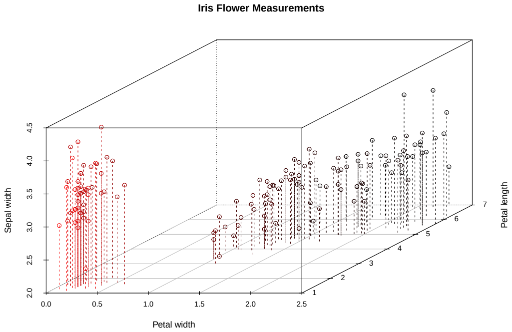
3D Scatterplots

```r
#library(scatterplot3d)
pwid <- iris$Petal.Width; plen <- iris$Petal.Length
swid <- iris$Sepal.Width; slen <- iris$Sepal.Length
scatterplot3d(x=pwid,y=plen,z=swid)
```



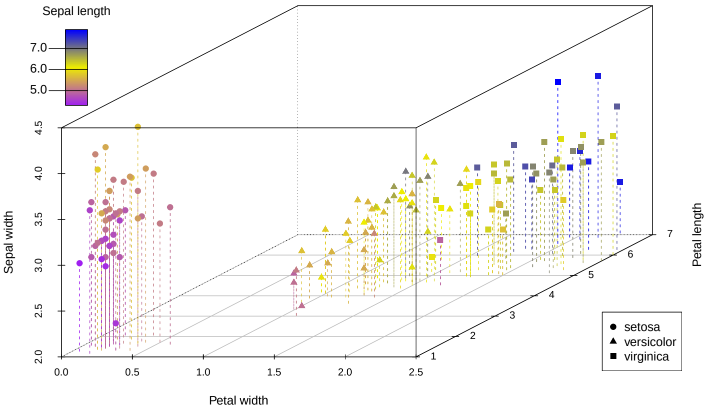
Visual Enhancements

```
scatterplot3d(x=pwid,y=plen,z=swid,highlight.3d=TRUE,type="h",
              lty.hplot=2,lty.hide=3,xlab="Petal width",
              ylab="Petal length",zlab="Sepal width",
              main="Iris Flower Measurements")
```



```r
normalize <- function(datavec){
  lo <- min(datavec,na.rm=TRUE)
  up <- max(datavec,na.rm=TRUE)
  datanorm <- (datavec-lo)/(up-lo)
  return(datanorm)
}

keycols <- c("purple","yellow2","blue")
slen.pal <- colorRampPalette(keycols)
slen.pal2 <- colorRamp(keycols)
slen.cols <- rgb(slen.pal2(normalize(slen)),maxColorValue=255)

scatterplot3d(x=pwid,y=plen,z=swid,color=slen.cols,
              pch=c(19,17,15)[as.numeric(iris$Species)],type="h",
              lty.hplot=2,lty.hide=3,xlab="Petal width",
              ylab="Petal length",zlab="Sepal width",
              main="Iris Flower Measurements")

legend("bottomright",legend=levels(iris$Species),pch=c(19,17,15))

colorlegend(slen.pal(200),zlim=range(slen),zval=5:7,digit=1,
            posx=c(0.1,0.13),posy=c(0.7,0.9),
            left=TRUE,main="Sepal length")
```

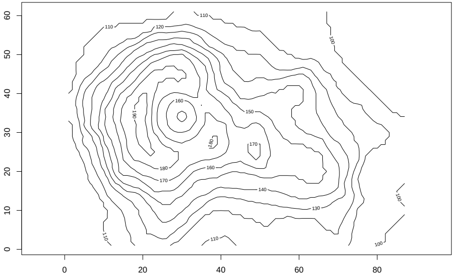
Iris Flower Measurements



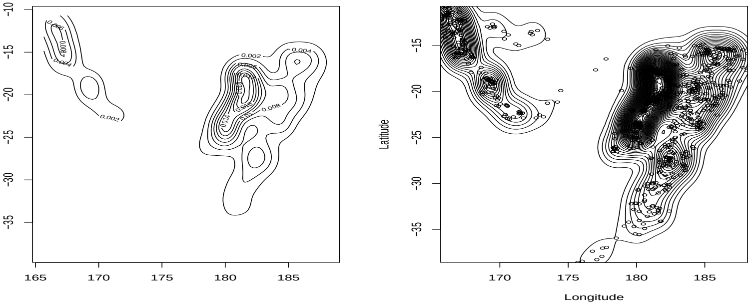
Contour Plots

One of the most common plots used to display a surface based on evaluation of a function over a grid of bivariate coordinates.

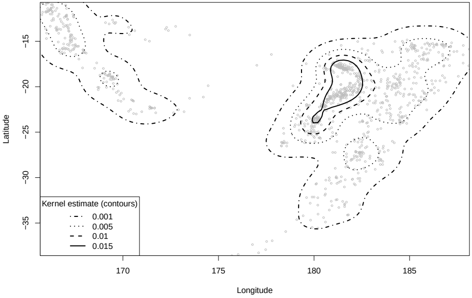
```
#dim(volcano) # 87 X 61
contour(x=1:nrow(volcano),y=1:ncol(volcano),z=volcano,asp=1)
```



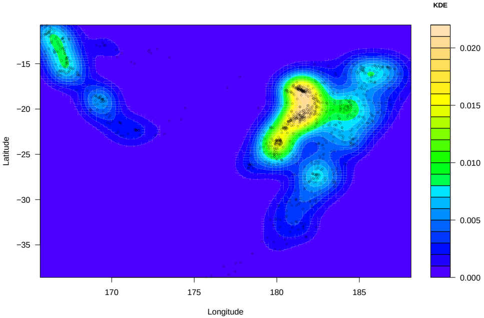
```r
# Nonparametric Bivariate Density Estimate
#library(MASS)
quak.dens <- kde2d(x=quakes$long,y=quakes$lat,n=100)
#dim(quak.dens$z) # 100 X 100
par(mfrow=c(1,2))
contour(quak.dens$x,quak.dens$y,quak.dens$z)
contour(quak.dens$x,quak.dens$y,quak.dens$z,nlevels=50,drawlabels=FALSE,
        xaxs="i",yaxs="i",xlab="Longitude",ylab="Latitude")
points(quakes$long,quakes$lat,cex=0.7)
```



```r
plot(quakes$long,quakes$lat,cex=0.5,col="gray",xaxs="i",yaxs="i",
     xlab="Longitude",ylab="Latitude")
quak.levs <- c(0.001,0.005,0.01,0.015)
contour(quak.dens$x,quak.dens$y,quak.dens$z,add=TRUE,levels=quak.levs,
        drawlabels=FALSE,lty=4:1,lwd=2)
legend("bottomleft",legend=quak.levs,lty=4:1,lwd=2,
       title="Kernel estimate (contours)")
```



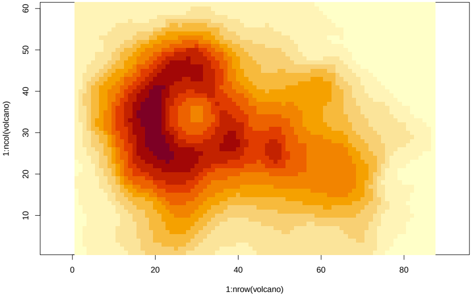
```r
filled.contour(x=quak.dens$x,y=quak.dens$y,z=quak.dens$z,
               color.palette=topo.colors,nlevels=30,xlab="Longitude",
               ylab="Latitude",key.title=title(main="KDE",cex.main=0.8),
               plot.axes={axis(1);axis(2);
                 points(quakes$long,quakes$lat,cex=0.5,
                        col=adjustcolor("black",alpha=0.3))})
```



Pixel Images

Its appearance is similar to a filled contour plot, but an image plot gives you more direct control over the display of each entry of the relevant z-matrix.

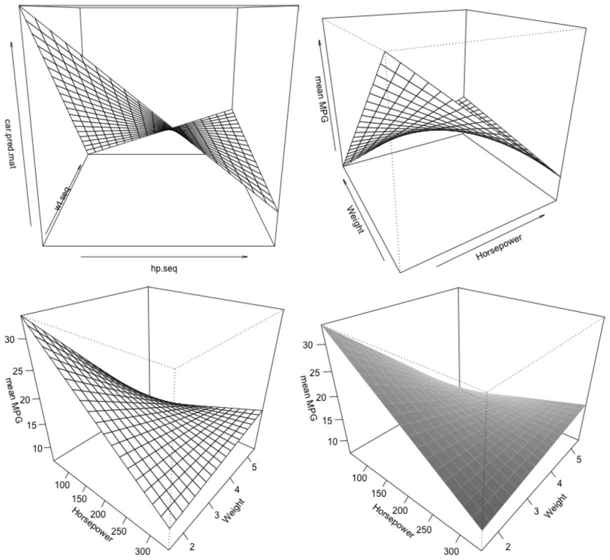
```
image(x=1:nrow(volcano),y=1:ncol(volcano),z=volcano,asp=1)
```



Pespective Plots (wireframe)

Unlike contour plots and pixel images, where fluctuations in the surface are emphasized with line pattern and/or colors, a perspective plot uses a physical third dimension against which the z value is plotted.

```
car.fit <- lm(mpg~hp*wt,data=mtcars); len <- 20
hp.seq <- seq(min(mtcars$hp),max(mtcars$hp),length=len)
wt.seq <- seq(min(mtcars$wt),max(mtcars$wt),length=len)
hp.wt <- expand.grid(hp=hp.seq,wt=wt.seq)
car.pred.mat <- matrix(predict(car.fit,newdata=hp.wt),nrow=len,ncol=len)
par(mfrow=c(2,2))
persp(x=hp.seq,y=wt.seq,z=car.pred.mat)
persp(x=hp.seq,y=wt.seq,z=car.pred.mat,theta=-30,phi=23,
      xlab="Horsepower",ylab="Weight",zlab="mean MPG")
persp(x=hp.seq,y=wt.seq,z=car.pred.mat,theta=40,phi=30,ticktype="detailed",
      xlab="Horsepower",ylab="Weight",zlab="mean MPG")
persp(x=hp.seq,y=wt.seq,z=car.pred.mat,theta=40,phi=30,ticktype="detailed",
      shade=0.6,border=NA,expand=0.8,
      xlab="Horsepower",ylab="Weight",zlab="mean MPG")
```


car.pred.mat
wt.seq
hp.seq
mean MPG
Weight
Horsepower
30
25
20
15
10
100
150
200
250
300
2
3
4
5

Interactive 3D Plots

When it comes to 3D plots, it's important to be able to view them from different angles to interpret the function or surface that's been displayed.

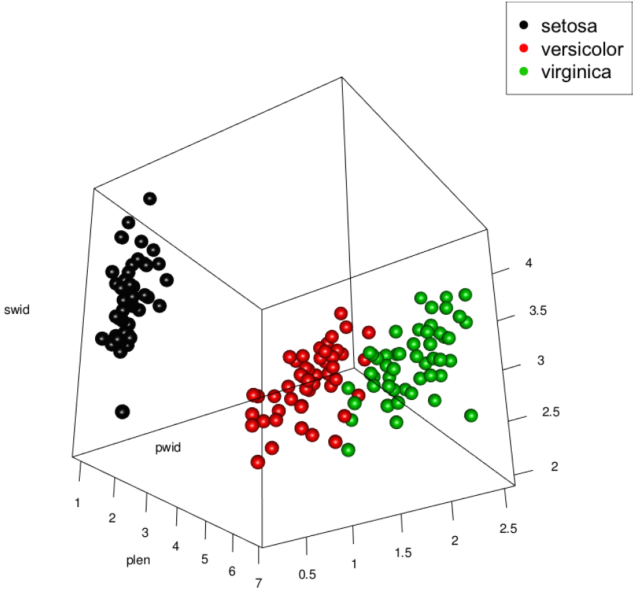
The `rgl` package offers some fantastic, simple-to-use R functions that allow you to rotate and zoom in on three-dimensional plots.

Point Clouds

Basic 3D Cloud (scatterplot of three continuous variables)

- `plot3d`: displays an interactive 3D cloud of points
- `legend3d`: inserts a static, unmovable legend
- `bg3d`: resets the background to its default white canvas

```
#library(rgl)
pwid <- iris$Petal.Width; plen <- iris$Petal.Length
swid <- iris$Sepal.Width; slen <- iris$Sepal.Length
#plot3d(x=pwid,y=plen,z=swid)
plot3d(x=pwid,y=plen,z=swid,size=1.5,type="s",
       col=c(1,2,3)[as.numeric(iris$Species)])
legend3d("topright",col=1:3,legend=levels(iris$Species),pch=16,cex=2)
bg3d(color="white")
```

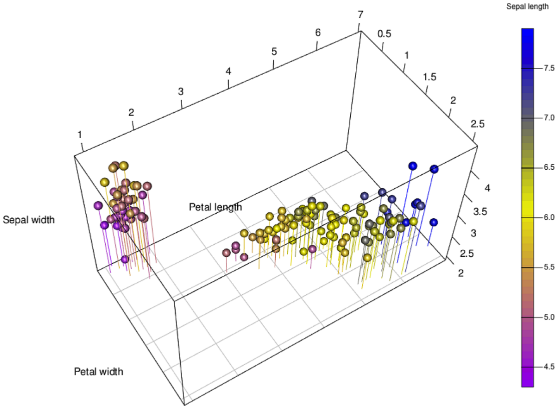

setosa
versicolor
virginica
swid
pwid
plen
1
2
3
4
5
6
7
0.5
1
1.5
2
2.5
2
2.5
3
3.5
4

Adding Further 3D Components

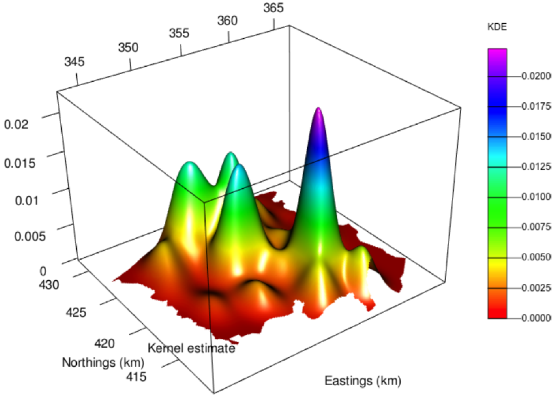
- `points3d`, `lines3d`, and `segments3d`: adds new points, lines, and segments to a current 3D plot

```
slen.pal <- colorRampPalette(c("purple","yellow2","blue"))
cols <- slen.pal(50)
slen.cols <- cut(slen,breaks=seq(min(slen),max(slen),length=51),
  include.lowest=TRUE)
plot3d(x=pwid,y=plen,z=swid,type="s",size=1.5,col=cols[slen.cols],
       aspect=c(1,1.75,1),xlab="Petal width",ylab="Petal length",
       zlab="Sepal width")
# draw the vertical lines
xfromto <- rep(pwid,each=2)
yfromto <- rep(plen,each=2)
zfromto <- rep(min(swid),times=2*nrow(iris))
zfromto[seq(2,length(zfromto),2)] <- swid
segments3d(x=xfromto,y=yfromto,z=zfromto,col=rep(cols[slen.cols],each=2))
# places a reference grid over the lower x-y plane
grid3d(side="z-")
```

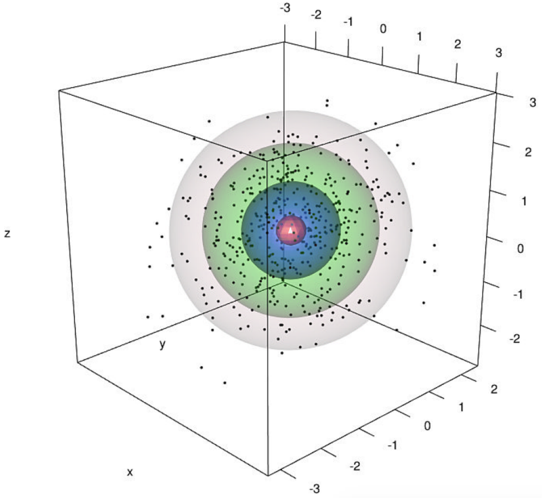
```r
#library(shape)
# add a color legend to the plot
bgplot3d({plot.new();colorlegend(slen.pal(50),zlim=range(slen),
                                   zval=seq(4.5,7.5,0.5),digit=1,
                                   posx=c(0.91,0.93),posy=c(0.1,0.9),
                                   main="Sepal length")})
```



Bivariate Surfaces (`persp3d`)



Trivariate Surfaces (`contour3d`)

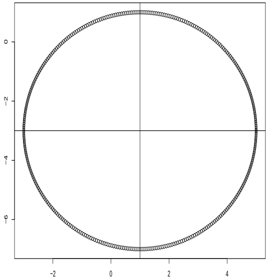


Handling Parametric Equations

2D Circle

$$x = a + r\cos(\theta) \text{ and } y = b + r\sin(\theta)$$

(a, b): center, $r > 0$: radius, $0 \leq \theta < 2\pi$: angle

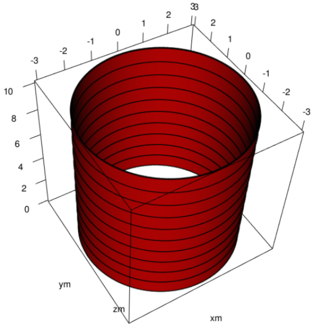


```r
radius <- 4
a <- 1
b <- -3
angle <- 0:360*(pi/180)

x <- a+radius*cos(angle)
y <- b+radius*sin(angle)

plot(x,y,ann=FALSE)
abline(v=a)
abline(h=b)
```

3D Cylinder

$$x = r\cos(\theta), y = r\sin(\theta), \text{ and } z = z$$

r: radius, $0 \le \theta < 2\pi$: angle, $0 \le z \le h$

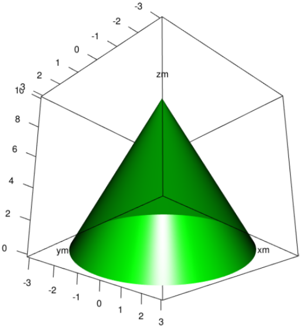


```r
r <- 3
h <- 10
zseq <- 0:h
theta <- 0:360*(pi/180)
#ztheta <- expand.grid(zseq,theta)

# nrow(ztheta) # 3971
xm <- outer(zseq,theta,function(z,t) r*cos(t))
ym <- outer(zseq,theta,function(z,t) r*sin(t))
zm <- outer(zseq,theta,function(z,t) z)

persp3d(x=xm,y=ym,z=zm,col="red")
points3d(x=xm,y=ym,z=zm)
```

3D Cone

$$x = \frac{h-z}{h} r\cos(\theta), y = \frac{h-z}{h} r\sin(\theta), \text{ and } z = z$$

r: radius, h: maximum height, θ: angle

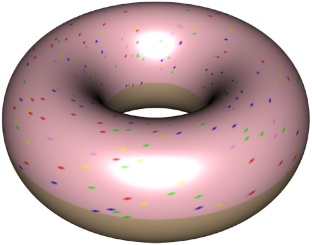


```r
r <- 3
h <- 10
zseq <- 0:h
theta <- 0:360*(pi/180)
#ztheta <- expand.grid(zseq,theta)

# nrow(ztheta) # 3971
xm <- outer(zseq,theta,function(z,t) (h-z)/h*r*cos(t))
ym <- outer(zseq,theta,function(z,t) (h-z)/h*r*sin(t))
zm <- outer(zseq,theta,function(z,t) z)

persp3d(x=xm,y=ym,z=zm,col="green")
```

Torus

$x = \beta + \alpha\cos(\theta_2)\cos(\theta_1), y = \beta + \alpha\cos(\theta_2)\sin(\theta_1),$ and $z = \alpha\sin(\theta_2)$

$0 \leq \theta_1, \theta_2 < 2\pi$: angles, α, β: radius of the "tube"



```r
res <- 200
theta <- 0:360*(pi/180)
alpha <- 1
beta <- 2
xm <- outer(theta,theta,function(t1,t2) (beta+alpha*cos(t2))*cos(t1))
ym <- outer(theta,theta,function(t1,t2) (beta+alpha*cos(t2))*sin(t1))
zm <- outer(theta,theta,function(t1,t2) alpha*sin(t2))

donutcols <- rep("tan",res^2)
donutcols[as.vector(zm)>0] <- "pink"
sprinkles <- c("blue","green","red","violet","yellow")
donutcols[sample(x=which(as.vector(zm)>0),size=300)] <- sprinkles

persp3d(xm,ym,zm,col=donutcols,aspect=c(1,1,0.4),axes=FALSE,
        xlab="",ylab="",zlab="")
```

Reference

- Davies, T. M. The Book of R. No Starch Press. Chapters 25 and 26.