Workshop 6

- Wishart Distribution
  - Joint distribution of eigenvalues
  - Diagonal elements of a Wishart random matrix
- Generalised Variance
  - Interpreting the generalised variance
  - Distribution of sample GV using classic theory
    - Uncorrelated and low dimensional ($p = 3$)
    - Correlated and low dimensional ($p = 3$)
    - Uncorrelated and higher dimensional ($p = 10$)
    - Uncorrelated and higher dimensional ($p = 30$)
- Distribution of sample GV in high-dimensional setting
  - Uncorrelated and higher dimensional ($p = 30$)

Wishart Distribution

Joint distribution of eigenvalues

First we create a function to calculate the eigenvalues of a matrix `x`. We need to `sample` as `eigen` always returns the eigenvalues in *decreasing* order. If you don't randomly shuffle them then you don't see any interesting patterns.

```
eigenvalues <- function(x) {
  sample(eigen(x, only.values=T)$values, nrow(x))
}
```

Define our covariance matrix Σ.

```
Sigma <- matrix(c(1,1/3,1/3,1), ncol=2)
Sigma
```

```
##           [,1]      [,2]
## [1,] 1.0000000 0.3333333
## [2,] 0.3333333 1.0000000
```

The function `rWishart` generates n random matrices, distributed according to the Wishart distribution with parameters Σ and $N = 10$ degrees of freedom $W_p(N, \Sigma)$. It returns a numeric array, say `R`, of dimension $p \times p \times n$, where each `R[,,i]` is a positive definite matrix, a realization of the Wishart distribution $W_p(N, \Sigma)$.

```
rWishart(2, 10, Sigma)
```

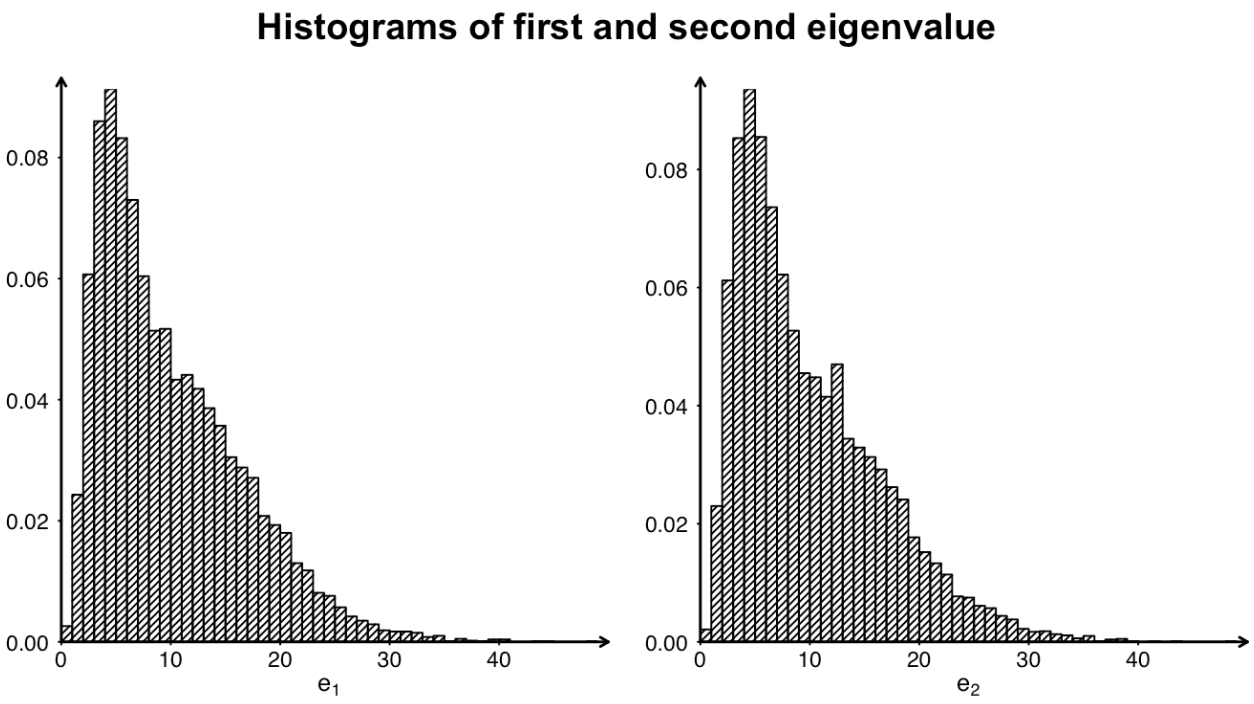
```
## , , 1
##
##          [,1]     [,2]
## [1,] 6.978381 1.668078
## [2,] 1.668078 7.183889
##
## , , 2
##
##           [,1]      [,2]
## [1,]  8.683242 -2.204733
## [2,] -2.204733 14.872851
```

Now we sample $n = 10^4$ Wishart matrices with covariance matrix Σ and calculate the eigenvalues. We use the function `apply` to apply the `eigenvalues` function (above) to the 3rd margin of the numeric array returned by `rWishart`.

```
Ls <- apply(rWishart(10^4, 10, Sigma), 3, eigenvalues)
```

We can now plot the histograms of the marginals.

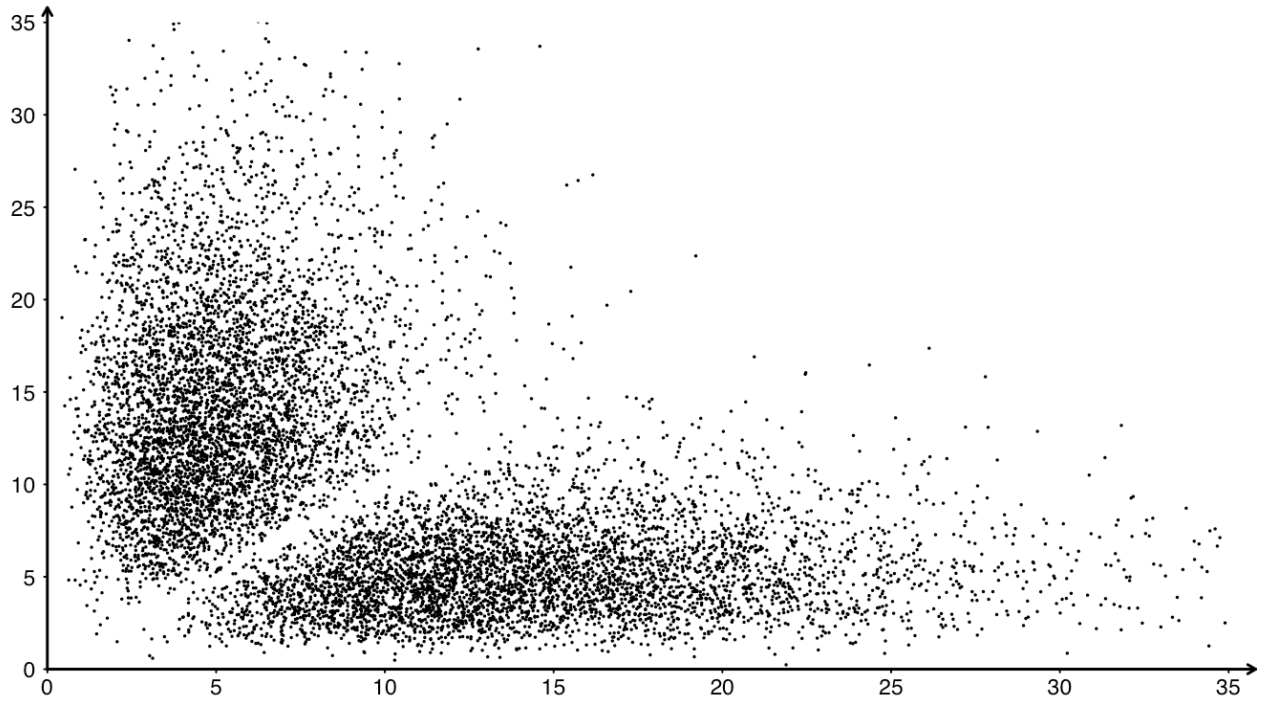
```
par(mfrow=c(1,2),
    oma = c(1,0,2,0) + 0.1,
    mar = c(0,0,2,2) + 0.1) -> opar

hist(Ls[1,], xlab=expression(e[1]))
hist(Ls[2,], xlab=expression(e[2]))

title(main="Histograms of first and second eigenvalue", outer=T, line=1)
```




```
par(opar)
```

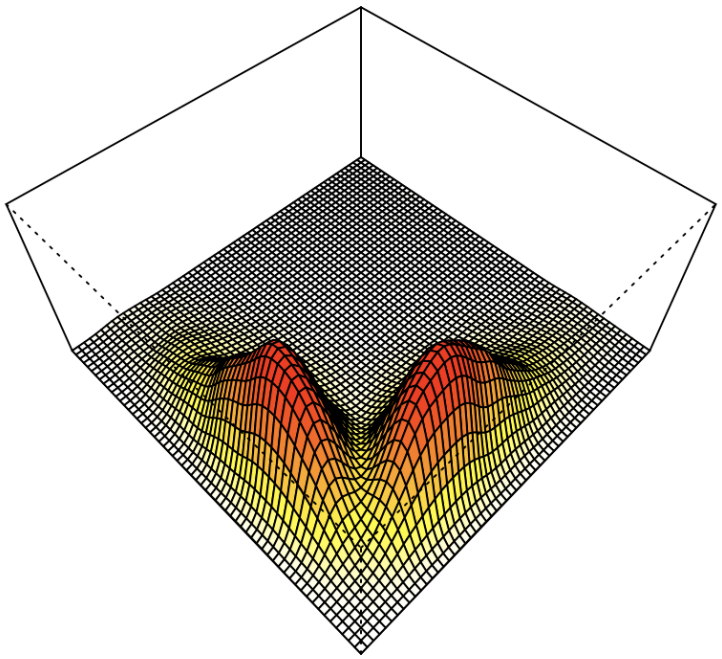
Scatter plot of the eigenvalues.

```
plot(Ls[1,], Ls[2,], xlim=c(0,35), ylim=c(0,35))
```



Or if we are feeling fancy, generate a nice 3D plot using a kernel density estimate.

```
require(MASS)

f <- kde2d(Ls[1,], Ls[2,], n=60, lims=c(-1,35,-1,35))

# set margins
par(mai=c(0.1,0.1,0.1,0.1)) -> opar

# extract data
e1 <- f$x
e2 <- f$y
z <- f$z

# generate colors
nb.col <- 256
color <- heat.colors(nb.col)
nrz <- nrow(z)
ncz <- ncol(z)
facet <- -(z[-1, -1] + z[-1, -ncz] + z[-nrz, -1] + z[-nrz, -ncz])
facetcol <- cut(facet, nb.col)

# plot
persp(e1, e2, z, phi = 50, theta = -45,
      expand=0.5, col=color[facetcol],
      ticktype="detailed", axes=F)
```



```
par(opar)
```

Diagonal elements of a Wishart random matrix

We are now going to look at the diagonal elements of a Wishart matrix. First, we sample $n = 10^4$ Wishart matrices and extract their diagonals. First we setup our Σ and our degrees of freedom $N = 10$

```
N <- 10
Sigma <- matrix(c(1,4/5,4/5,1), ncol=2)
Sigma
```

```
##      [,1] [,2]
## [1,]  1.0  0.8
## [2,]  0.8  1.0
```

Now we sample and extract the diagonals using `apply` .

```
Ds <- apply(rWishart(10^4, N, Sigma), 3, diag)
```

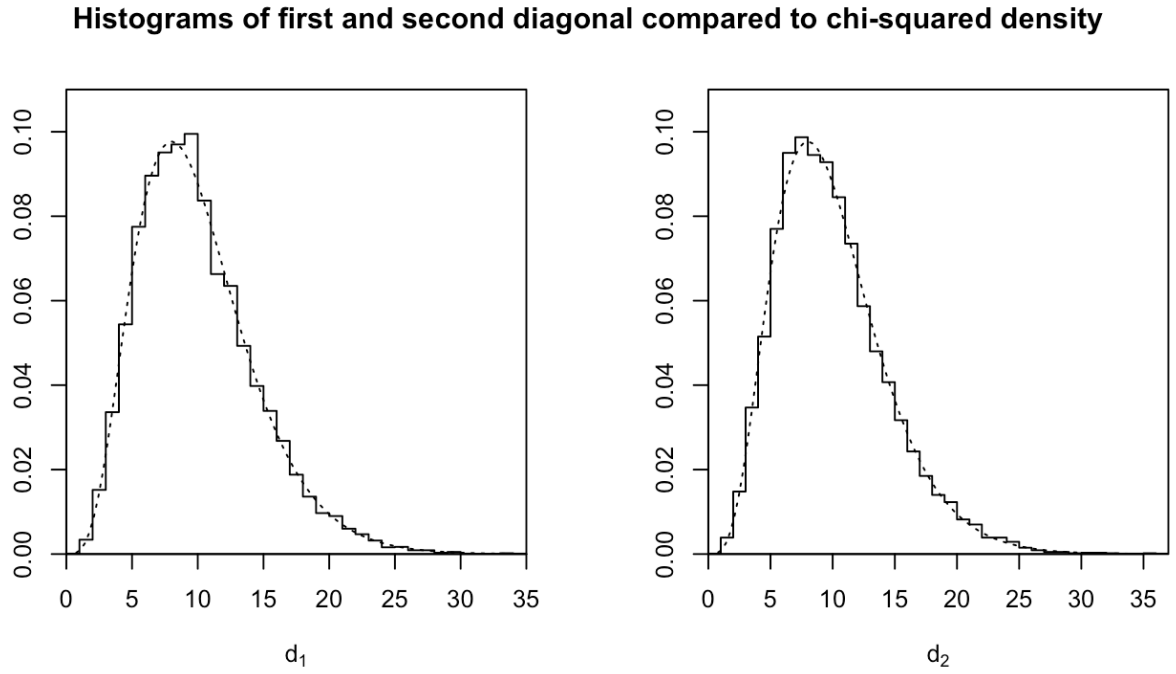
We plot histogram of marginals and compare the density against the density of a χ^2 distribution with appropriate degrees of freedom.

Create a histogram of each marginal and store it.

```
hist(Ds[1,], breaks=30, plot=F) -> h1
hist(Ds[2,], breaks=30, plot=F) -> h2
```

Create a custom histogram plot that looks nice.

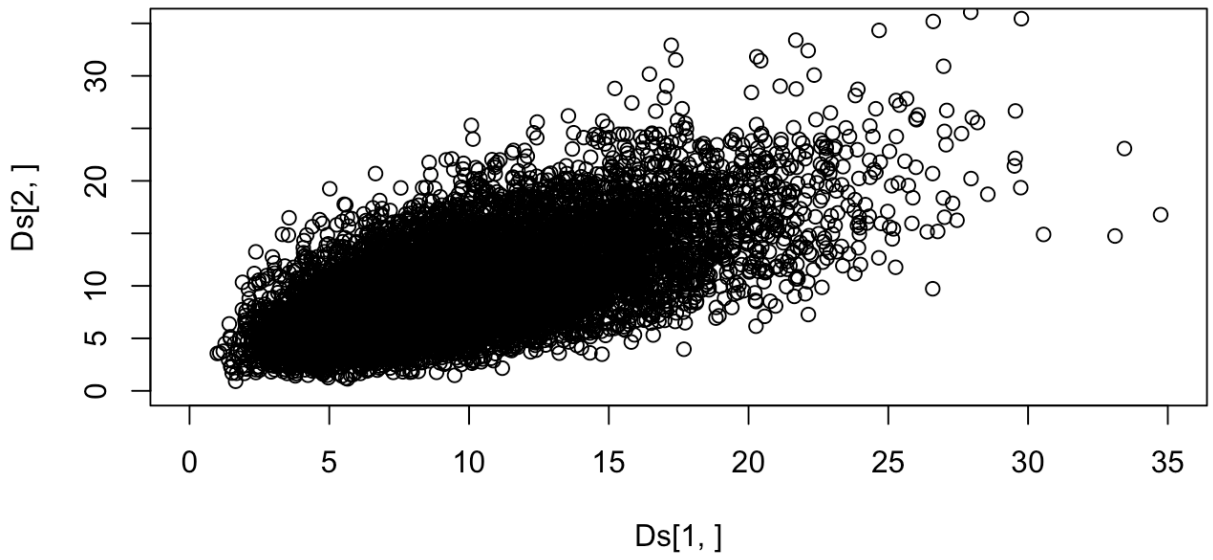
```
par(mfrow=c(1,2), xaxs="i", yaxs="i", cex=0.8,
    cex.axis=1.0) -> opar

# density of marginal
f <- function(x) dchisq(x, df=N)

plot(h1$breaks, c(h1$density, 0), type="s",
     xlab=expression(d[1]), ylab="", ylim=c(0, 0.11))
curve(f, 0, 35, lwd=1, lty=3, add=TRUE)

plot(h2$breaks, c(h2$density, 0), type="s",
     xlab=expression(d[2]), ylab="", ylim=c(0, 0.11))
curve(f, 0, 35, lwd=1, lty=3, add=TRUE)

title(main="Histograms of first and second diagonal compared to chi-squared den
sity", outer=T, line=-2)
```




```
par(opar)
```

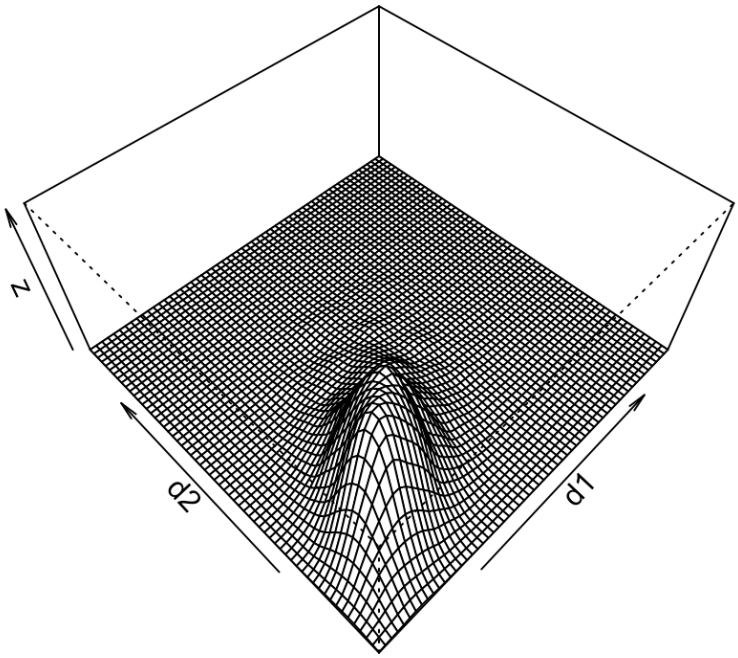
Generating a scatter plot of diagonal elements shows that they are not independent.

```
plot(Ds[1,], Ds[2,], xlim=c(0,35), ylim=c(0,35))
```



Or a nice 3D plot.

```
require(MASS)

f <- kde2d(Ds[1,], Ds[2,], n=60)

# set margins
par(mai=c(0.1,0.1,0.1,0.1)) -> opar

# extract data
d1 <- f$x
d2 <- f$y
z <- f$z

persp(d1, d2, z, phi = 50, theta = -45, expand=0.5)
```




```
par(opar)
```

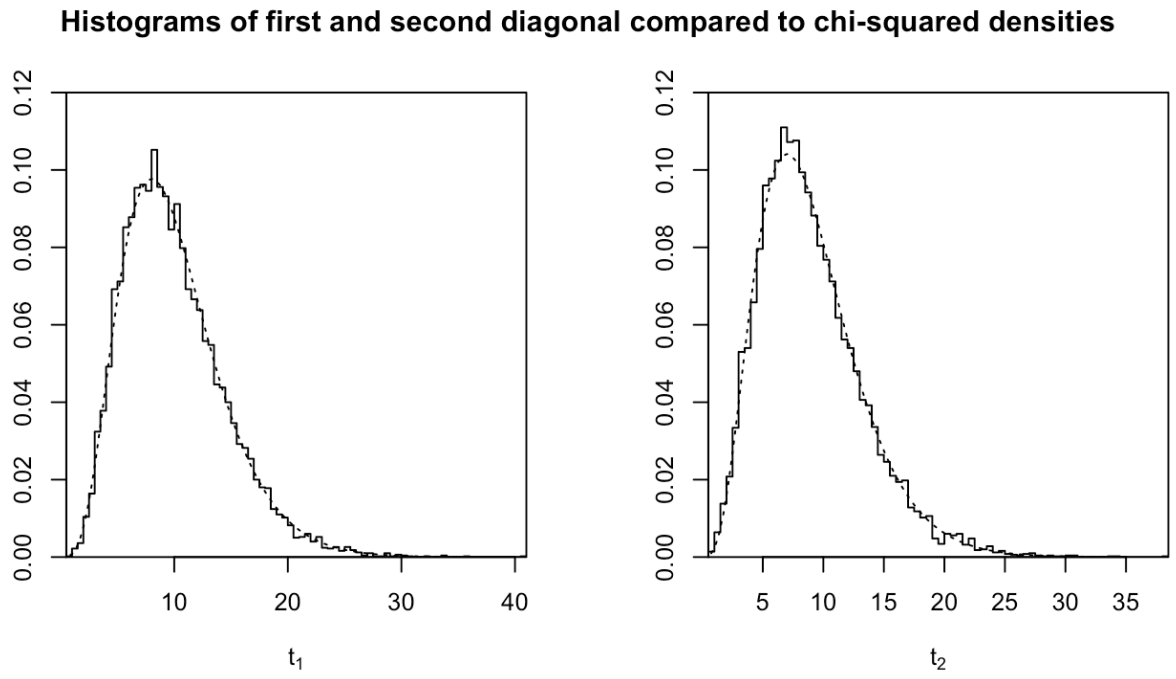
Looking at the diagonal case $\Sigma = I_p$ and performing the Cholesky decomposition.

```
Sigma <- diag(1, 2, 2)

tdiag <- function(A) {
  diag(chol(A))^2
}

Ts <- apply(rWishart(10^4, N, Sigma), 3, tdiag)
```

```
hist(Ts[1,], breaks="FD", plot=F) -> h1
hist(Ts[2,], breaks="FD", plot=F) -> h2
```

Create a custom histogram plot that looks nice.

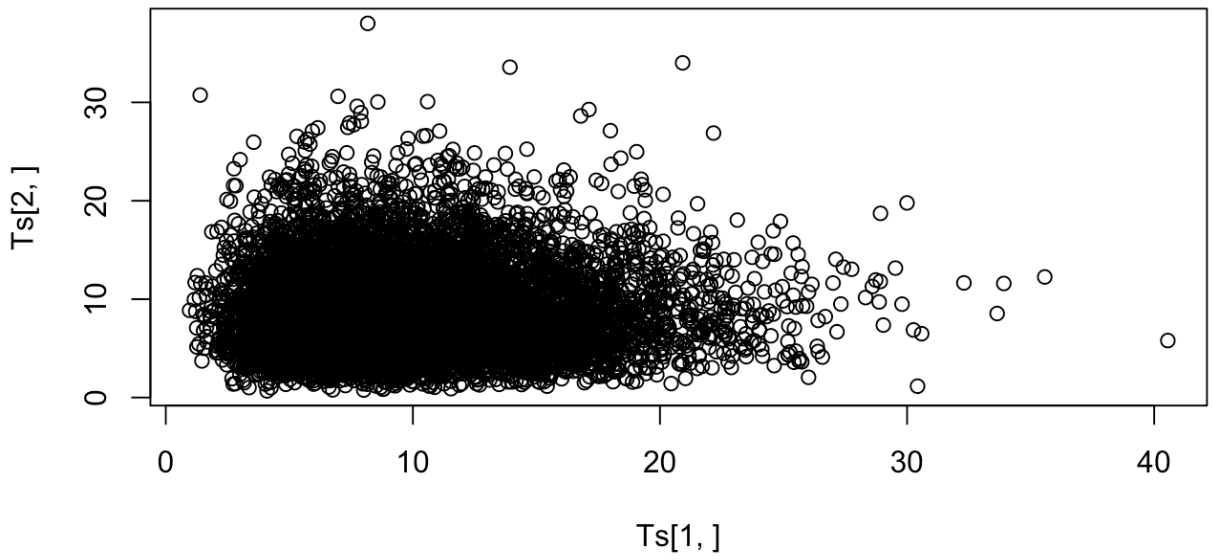
```
par(mfrow=c(1,2), xaxs="i", yaxs="i", cex=0.8,
    cex.axis=1.0) -> opar

# density of marginal
f <- function(x) dchisq(x, df=N)

plot(h1$breaks, c(h1$density, 0), type="s",
     xlab=expression(t[1]), ylab="", ylim=c(0, 0.12))
curve(f, 0, 35, lwd=1, lty=3, add=TRUE)

f <- function(x) dchisq(x, df=N-1)

plot(h2$breaks, c(h2$density, 0), type="s",
     xlab=expression(t[2]), ylab="", ylim=c(0, 0.12))
curve(f, 0, 35, lwd=1, lty=3, add=TRUE)

title(main="Histograms of first and second diagonal compared to chi-squared den
sities", outer=T, line=-2)
```




```
par(opar)
```

We see that they are uncorrelated.

```
plot(Ts[1,], Ts[2,])
```



Generalised Variance

Interpreting the generalised variance

First, we are going to show that the generalised variance does not properly capture some details about the covariance structure of the population. We shall do this by constructing three different distributions with exactly the same *population generalised variance*.

Set the mean μ.

```
mu <- c(2, 1)
```

The covariance Σ_1.

```
Sigma1 <- matrix(c(5,4,4,5), ncol=2)
Sigma1
```

```
##      [,1] [,2]
## [1,]    5    4
## [2,]    4    5
```

The covariance Σ_2.

```
Sigma2 <- matrix(c(3,0,0,3), ncol=2)
Sigma2
```

```
##      [,1] [,2]
## [1,]    3    0
## [2,]    0    3
```

The covariance Σ_3.

```
Sigma3 <- matrix(c(5,-4,-4,5), ncol=2)
Sigma1
```

```
##      [,1] [,2]
## [1,]    5    4
## [2,]    4    5
```

We load the `mvtnorm` library for sampling from the multivariate normal distribution.

```
library(mvtnorm)
```

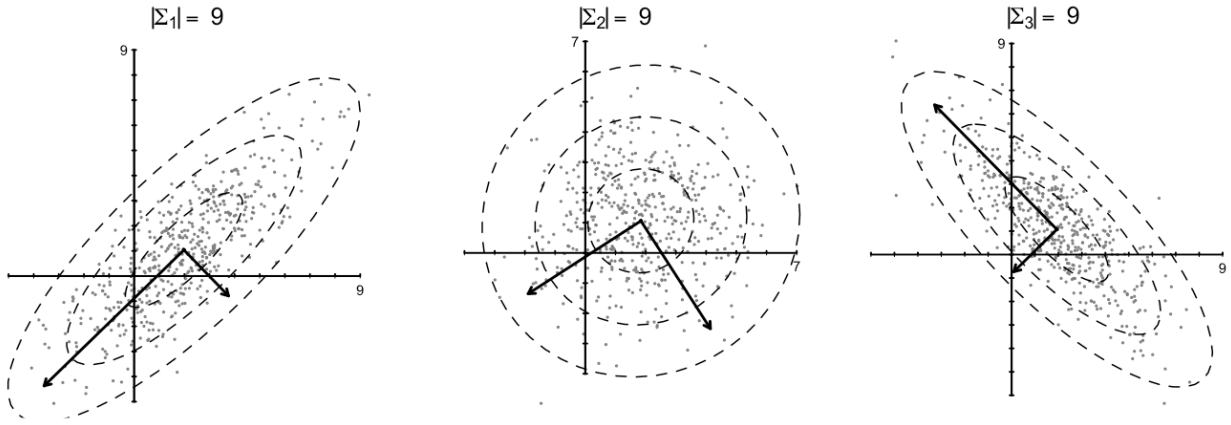
Now we create a function that generates a nice scatter plot for us that has the confidence ellipses and plots the eigenvectors. You may need to install the package `car` for the `ellipse` function.

```
SA <- function(X) {
  require(car)

  par(pty="s", bty="n") -> opar

  plot(X[,1], X[,2], pch=20, col="gray50", cex=0.2,
       axes=F, xlab="", ylab="", xlim=range(X),
       ylim=range(X))

  mr <- as.integer(min(range(X)))
  Mr <- as.integer(max(range(X)))

  lpts <- seq(mr, Mr, 1)
  tpts <- rep('', (Mr+1-mr))
  tpts[(Mr+1-mr)] <- Mr
  tck = 0.005
  axis(1, pos=0, tick=T, tck = tck, label=F, at=lpts)
  axis(2, pos=0, tick=T, tck = tck, label=F, at=lpts)
  axis(1, pos=0, tick=T, tck = -1*tck, label=tpts,
       at=lpts,  padj=-2.5, cex.axis=0.7, lwd=0.5)
  axis(2, pos=0, tick=T, tck = -1*tck, label=tpts,
       at=lpts, cex.axis=0.7, hadj=-1.3, las=1)

  mu <- colMeans(X)
  Sigma <- var(X)
  e <- eigen(Sigma)

  for (i in 1:3) {
    ellipse(mu, Sigma, i, xlim=range(X), ylim=range(X),
          col=1, center.pch=19, center.cex=0.2,
          lwd=0.8, lty=2)
  }

  arrows(mu[1], mu[2],
         mu[1]+e$vectors[1,1]*sqrt(e$values[1])*2.5,
         mu[2]+e$vectors[2,1]*sqrt(e$values[1])*2.5,
         length=.04,col=1,lwd=1.5)
  arrows(mu[1], mu[2],
         mu[1]+e$vectors[1,2]*sqrt(e$values[2])*2.5,
         mu[2]+e$vectors[2,2]*sqrt(e$values[2])*2.5,
         length=.04, col=1, lwd=1.5)

  par(opar)
}
```

We now sample from each distribution and show that although they look very different each

distribution as the same population generalised variance.

```
par(mfrow=c(1,3), mai=c(0,0.1,0.1,0.1)) -> opar

n <- 500

X <- rmvnorm(n, mean=mu, sigma=Sigma1)
Sn <- t(X) %*% X / n
pGV <- det(Sigma1)
SA(X)
title(bquote(abs(Sigma[1])== ~ .(pGV)))

X <- rmvnorm(n, mean=mu, sigma=Sigma2)
Sn <- t(X) %*% X / n
pGV <- det(Sigma2)
SA(X)
title(bquote(abs(Sigma[2])== ~ .(pGV)))

X <- rmvnorm(n, mean=mu, sigma=Sigma3)
Sn <- t(X) %*% X / n
pGV <- det(Sigma3)
SA(X)
title(bquote(abs(Sigma[3])== ~ .(pGV)))
```




```
par(opar)
```

Distribution of sample GV using classic theory

We look at the theorem that we proved in the lectures that shows

$$\sqrt{n}\left(\frac{|\mathbf{S}|}{|\Sigma|} - 1\right) \to N(0, 2p)$$

for fixed p and $n \to \infty$.

Uncorrelated and low dimensional ($p = 3$)

We first try a simple example with $\Sigma = I_p$ and $p = 3$.

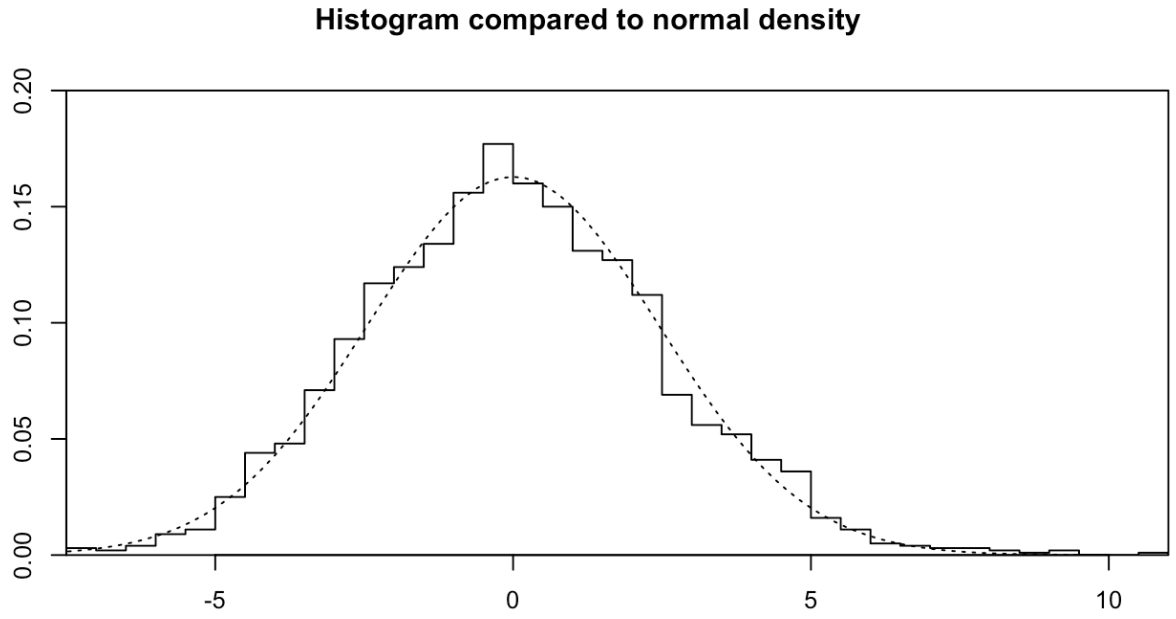
```
p <- 3
Sigma <- diag(1, p, p)
mu <- rep(0, p)

M <- 2000 # number of MC simulations
n <- 2000 # number of observations

MC <- numeric(M)
for (i in 1:M) {
  X <- rmvnorm(n, mean=mu, sigma=Sigma)
  Sn <- t(X) %*% X / (n-1)
  MC[i] <- sqrt(n)*(det(Sn)/det(Sigma) - 1)
}
```

Generate histogram of MC simulations.

```
hist(MC, breaks="FD", plot=F) -> h
```

Create a custom histogram plot that looks nice.

```
par(xaxs="i", yaxs="i", cex=0.8, cex.axis=1.0) -> opar

# theoretical density
f <- function(x) dnorm(x, mean=0, sd=sqrt(2*p))

plot(h$breaks, c(h$density, 0), type="s",
     xlab="", ylab="", ylim=c(0, 0.2))
curve(f, -10, 10, lwd=1, lty=3, add=TRUE)

title(main="Histogram compared to normal density", outer=T, line=-2)
```




```
par(opar)
```

Correlated and low dimensional ($p = 3$)

To create some test covariance matrices, I'll use the function from a previous workshop that generates this "power correlation matrix" (AR1) for an arbitrary p.

```
pcor <- function(rho, p) {
  Tn <- matrix(0, p, p)
  for (i in 1:p) {
    for (j in 1:p) {
      Tn[i,j] <- rho^abs(i-j)
    }
  }
  return(Tn)
}
```

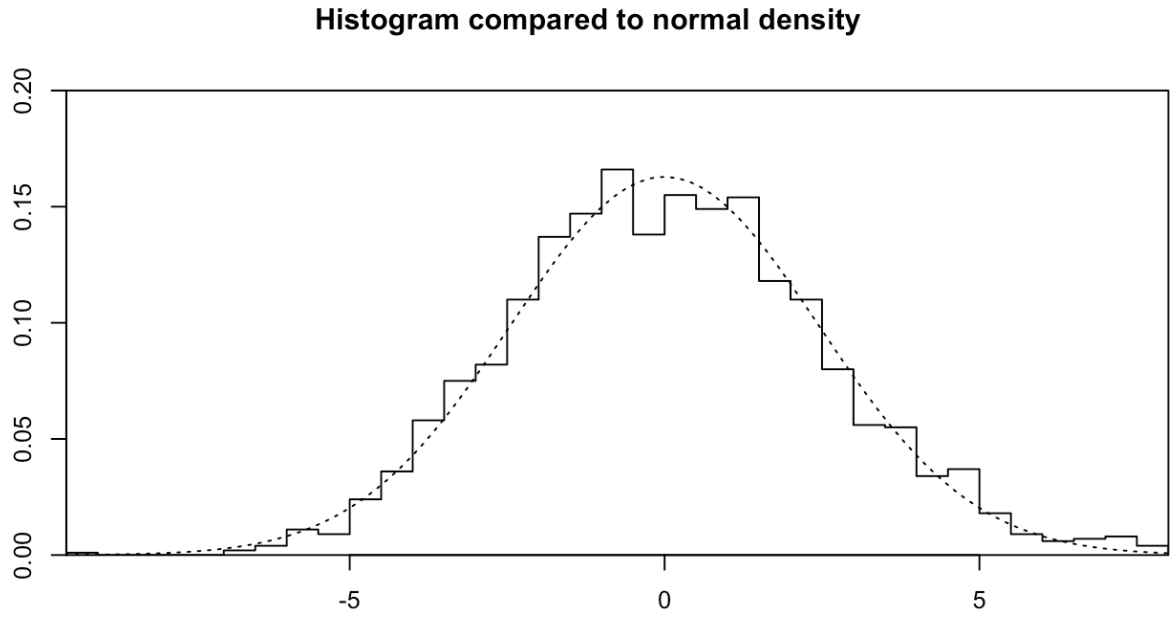
```
p <- 3
Sigma <- pcor(0.5, p)
mu <- rep(0, p)

M <- 2000 # number of MC simulations
n <- 2000 # number of observations

MC <- numeric(M)
for (i in 1:M) {
  X <- rmvnorm(n, mean=mu, sigma=Sigma)
  Sn <- t(X) %*% X / (n-1)
  MC[i] <- sqrt(n)*(det(Sn)/det(Sigma) - 1)
}
```

Generate histogram of MC simulations.

```
hist(MC, breaks="FD", plot=F) -> h
```

Create a custom histogram plot that looks nice.

```
par(xaxs="i", yaxs="i", cex=0.8, cex.axis=1.0) -> opar

# theoretical density
f <- function(x) dnorm(x, mean=0, sd=sqrt(2*p))

plot(h$breaks, c(h$density, 0), type="s",
     xlab="", ylab="", ylim=c(0, 0.2))
curve(f, -10, 10, lwd=1, lty=3, add=TRUE)

title(main="Histogram compared to normal density", outer=T, line=-2)
```




```
par(opar)
```

Uncorrelated and higher dimensional ($p = 10$)

We take with $\Sigma = I_p$ and $p = 10$.

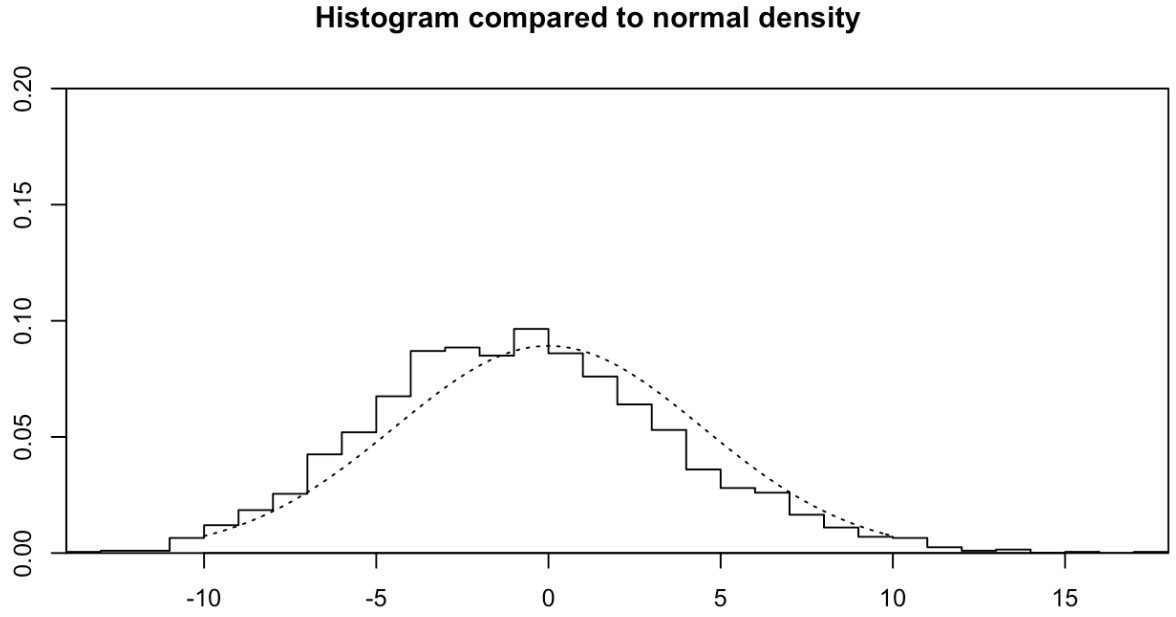
```
p <- 10
Sigma <- diag(1, p, p)
mu <- rep(0, p)

M <- 2000 # number of MC simulations
n <- 2000 # number of observations

MC <- numeric(M)
for (i in 1:M) {
  X <- rmvnorm(n, mean=mu, sigma=Sigma)
  Sn <- t(X) %*% X / (n-1)
  MC[i] <- sqrt(n)*(det(Sn)/det(Sigma) - 1)
}
```

Generate histogram of MC simulations.

```
hist(MC, breaks="FD", plot=F) -> h
```

Create a custom histogram plot that looks nice.

```
par(xaxs="i", yaxs="i", cex=0.8, cex.axis=1.0) -> opar

# theoretical density
f <- function(x) dnorm(x, mean=0, sd=sqrt(2*p))

plot(h$breaks, c(h$density, 0), type="s",
     xlab="", ylab="", ylim=c(0, 0.2))
curve(f, -10, 10, lwd=1, lty=3, add=TRUE)

title(main="Histogram compared to normal density", outer=T, line=-2)
```




```
par(opar)
```

Uncorrelated and higher dimensional ($p = 30$)

We take with $\Sigma = I_p$ and $p = 30$.

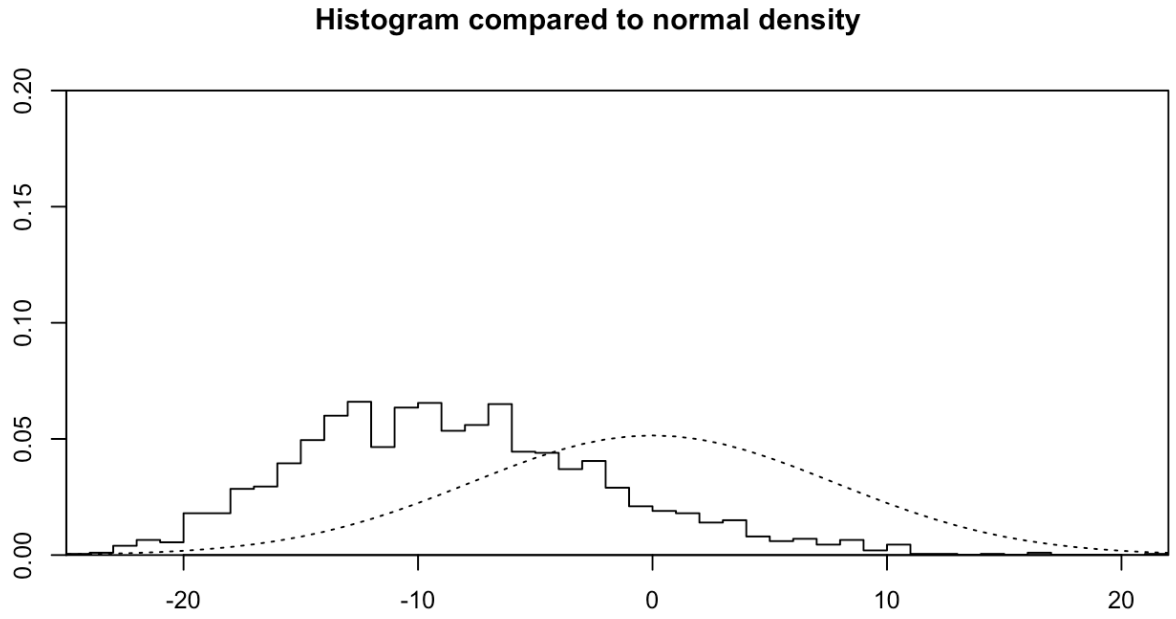
```
p <- 30
Sigma <- diag(1, p, p)
mu <- rep(0, p)

M <- 2000 # number of MC simulations
n <- 2000 # number of observations

MC <- numeric(M)
for (i in 1:M) {
  X <- rmvnorm(n, mean=mu, sigma=Sigma)
  Sn <- t(X) %*% X / (n-1)
  MC[i] <- sqrt(n)*(det(Sn)/det(Sigma) - 1)
}
```

Generate histogram of MC simulations.

```
hist(MC, breaks="FD", plot=F) -> h
```

Create a custom histogram plot that looks nice.

```
par(xaxs="i", yaxs="i", cex=0.8, cex.axis=1.0) -> opar

# theoretical density
f <- function(x) dnorm(x, mean=0, sd=sqrt(2*p))

plot(h$breaks, c(h$density, 0), type="s",
     xlab="", ylab="", ylim=c(0, 0.2))
curve(f, -50, 50, lwd=1, lty=3, add=TRUE)

title(main="Histogram compared to normal density", outer=T, line=-2)
```




```
par(opar)
```

Distribution of sample GV in high-dimensional setting

```
d <- function(u) 1 + (1-u)/u * log(1-u)
```

Uncorrelated and higher dimensional ($p = 30$)

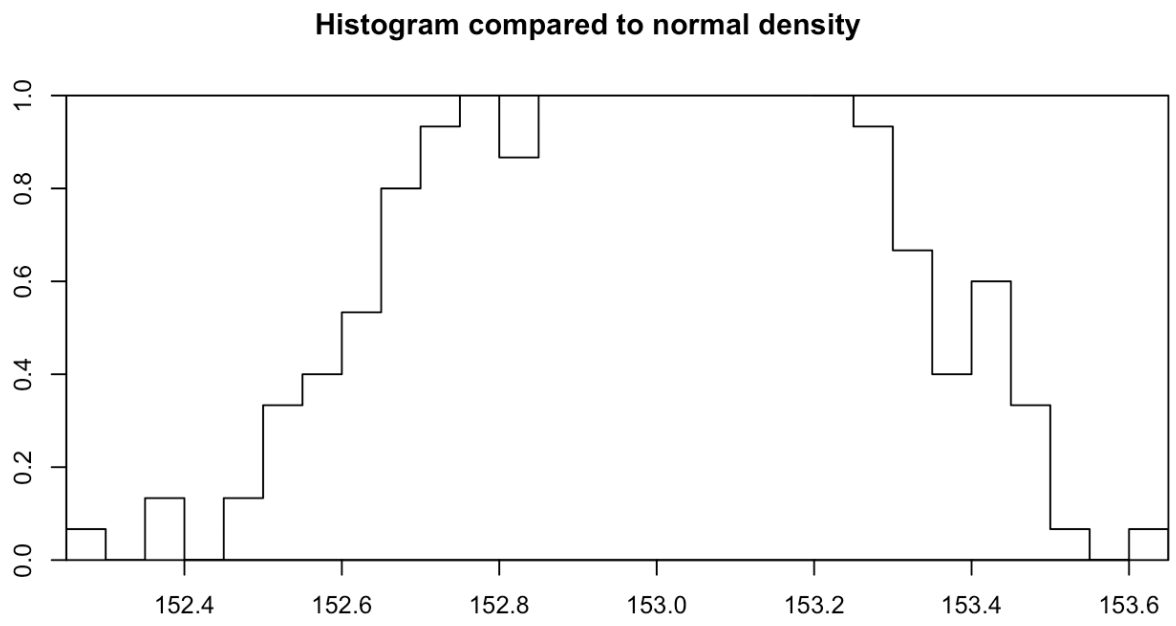
```
Sigma <- diag(1, p, p)
mu <- rep(0, p)

M <- 300 # number of MC simulations
n <- 1000 # number of observations
p <- 500

yn <- p/n

MC <- numeric(M)
for (i in 1:M) {
  X <- rmvnorm(n, mean=mu, sigma=Sigma)
  Sn <- t(X) %*% X / (n-1)
  MC[i] <- log(det(Sn)/det(Sigma)) + p*d(yn) - log(1-yn)/yn/p
}
```

Generate histogram of MC simulations.

```
hist(MC, breaks=30, plot=F) -> h
```

Create a custom histogram plot that looks nice.

```
par(xaxs="i", yaxs="i", cex=0.8, cex.axis=1.0) -> opar

# theoretical density
f <- function(x) dnorm(x, mean=0, sd=sqrt(-2*log(1-yn)))

plot(h$breaks, c(h$density, 0), type="s",
     xlab="", ylab="", ylim=c(0, 1.0))
curve(f, -5, 5, lwd=1, lty=3, add=TRUE)

title(main="Histogram compared to normal density", outer=T, line=-2)
```


Histogram compared to normal density
0.0
0.2
0.4
0.6
0.8
1.0
152.4
152.6
152.8
153.0
153.2
153.4
153.6

```
par(opar)
```