Projects 2023

Thierry Denoeux

2023-10-21

Logistic regression

Question 1

Let us first write a function that computes the log-likelihood:

```
loglik<-function(beta,X,y){
  p <- 1/(1+exp(-X %*% beta))
  l <- sum(y*log(p)+(1-y)*log(1-p))
  return(l)
}
```

To find the MLE, we can either use function glm in package stats, or maximize loglik directly. Let us try both approaches.

Approach 1:

```
library(aplore3)
fit<-glm(chd~age,data=chdage,family='binomial')
betah<-fit$coefficients
print(betah)
```

```
## (Intercept)         age
##  -5.3094534   0.1109211
```

Approach 2:

```
X<-model.matrix(chd~age,data=chdage)
y<-as.numeric(chdage$chd)-1
opt<-optim(par=c(0,0),fn=loglik,
           method="BFGS",
           control=list(trace=5,fnscale=-1,maxit=1000),
           X=X,y=y)
```

```
## initial  value 69.314718
## final  value 53.676547
## converged
```

```
betah<-opt$par
print(betah)
```

```
## [1] -5.3109895  0.1109536
```

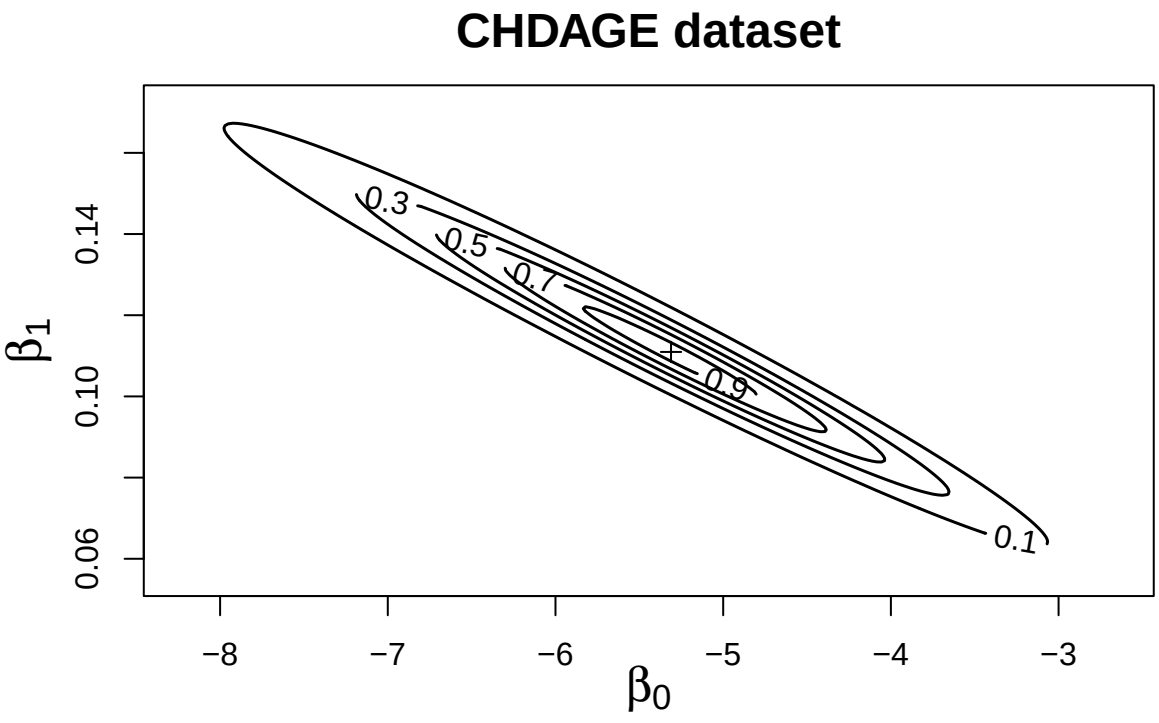
```
ellmax<- loglik(betah,X,y)
```

We can now write a function that computes the relative loglikelihood:

```r
# Relative likelihood
rellik<-function(beta,X,y,ellmax){
  l<-loglik(beta,X,y)-ellmax
  return(exp(l))
}
```

Let us plot the contours of this function for the `chdage` dataset:

```r
N<-200
f<-2
xmin<-betah[1]-1.1*abs(betah[1])/f
xmax<-betah[1]+abs(betah[1])/f
ymin<-betah[2]-abs(betah[2])/f
ymax<-betah[2]+1.1*abs(betah[2])/f
xx<-seq(xmin,xmax,(xmax-xmin)/(N-1))
yy<-seq(ymin,ymax,(ymax-ymin)/(N-1))
L<-matrix(0,N,N)
for(i in 1:N) for(j in 1:N) L[i,j]<-rellik(c(xx[i],yy[j]),X,y,ellmax)
contour(xx,yy,L,xlab="",ylab="",levels=seq(0.1,0.9,0.2),
        lwd=1.5,labcex = 1)
title(ylab=expression(beta[1]), line=2.2, cex.lab=1.5)
title(xlab=expression(beta[0]), line=2.2, cex.lab=1.5)
title(main="CHDAGE dataset", line=1, cex.main=1.5)
points(betah[1],betah[2],pch=3)
```



Question 2

Let $pl(\beta)$ denote the contour function (relative likelihood) of β. The marginal contour functions (relative profile likelihoods) of β_0 and β_0 are, respectively,

$$pl(\beta_0) = \sup_{\beta_1} pl(\beta) \quad \text{and} \quad pl(\beta_1) = \sup_{\beta_0} pl(\beta).$$

To compute these functions, we thus need to solve optimization problems. This function computes $pl(\beta)$ with the component `MARGIN` of β fixed at `beta1`:

```
rellik_fixed <- function(beta2,beta1,MARGIN,X,y,ellmax){
  beta<-c(0,0)
  beta[MARGIN]<- beta1
  beta[-MARGIN]<-beta2
  return(rellik(beta,X,y,ellmax))
}
```

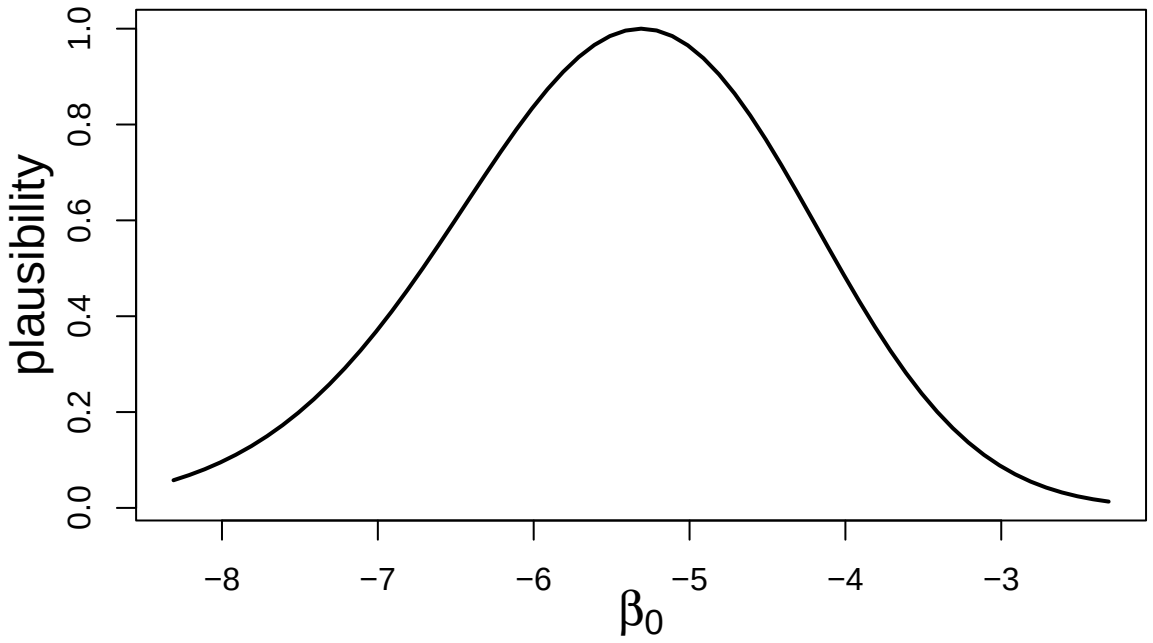
This function then compute the relative likelihood $pl(\beta_0)$ if `MARGIN=1` and $pl(\beta_1)$ if `MARGIN=2`:

```
rellik_profile <- function(beta1,MARGIN,X,y,ellmax,beta20){
  # beta1 must be a scalar
  opt<-optim(par=beta20,fn=rellik_fixed,method="BFGS",
             control=list(fnscale=-1,maxit=10000),
             beta1=beta1,MARGIN=MARGIN,X=X,y=y,ellmax=ellmax)
  return(opt)
}
```

We note that an initial value `beta20` for the parameter that is varied must be provided.

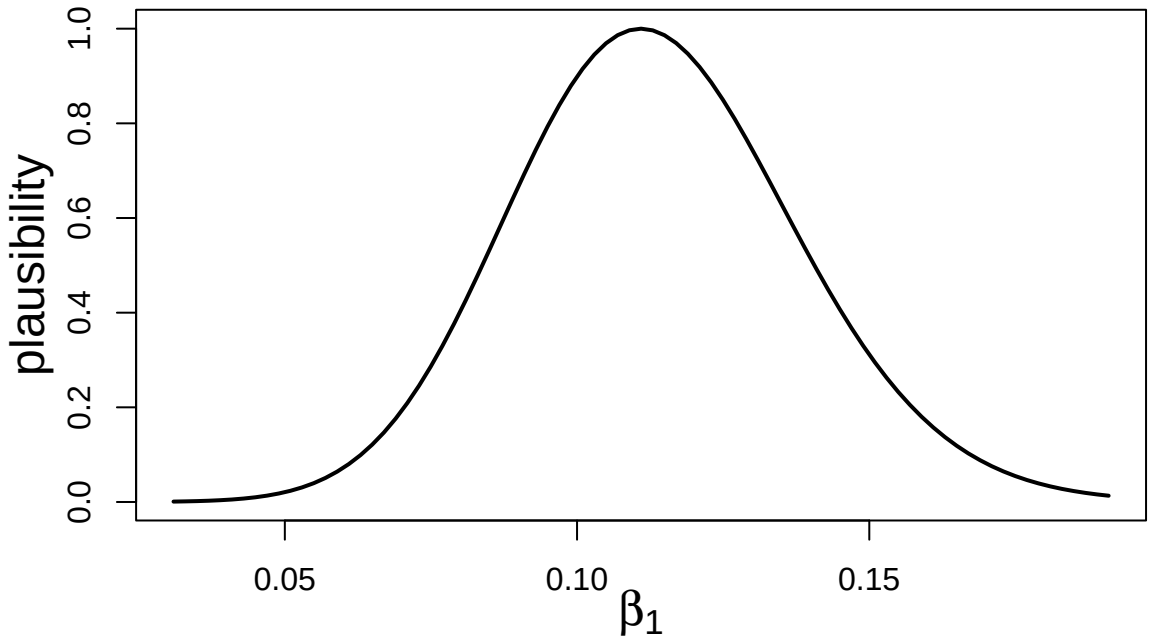
Let us now plot the marginal contour of β_0:

```
Beta0<-seq(betah[1]-3,betah[1]+3,0.1)
N<-length(Beta0)
pl <- rep(0,N)
i0<-(N+1)/2
pl[i0]<-1
beta20<-betah[2]
for(i in (i0+1):N){
  opt<-rellik_profile(beta1=Beta0[i],1,X,y,ellmax,beta20)
  pl[i]<-opt$value
  beta20<-opt$par
}
beta20<-betah[2]
for(i in (i0-1):1){
  opt<-rellik_profile(beta1=Beta0[i],1,X,y,ellmax,beta20)
  pl[i]<-opt$value
  beta20<-opt$par
}
plot(Beta0,pl,type="l",lwd=2,xlab="",ylab="")
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(beta[0]), line=2.2, cex.lab=1.5)
```



And similarly for β_1:

```
Beta1<-seq(betah[2]-0.08,betah[2]+0.08,0.002)
N<-length(Beta1)
pl <- rep(0,N)
i0<-(N+1)/2
pl[i0]<-1
beta20<-betah[1]
for(i in (i0+1):N){
  opt<-rellik_profile(beta1=Beta1[i],2,X,y,ellmax,beta20)
  pl[i]<-opt$value
  beta20<-opt$par
}
beta20<-betah[1]
for(i in (i0-1):1){
  opt<-rellik_profile(beta1=Beta1[i],2,X,y,ellmax,beta20)
  pl[i]<-opt$value
  beta20<-opt$par
}
plot(Beta1,pl,type="l",lwd=2,xlab="",ylab="")
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(beta[1]), line=2.2, cex.lab=1.5)
```



Question 3

We have

$$P(x) = \frac{1}{1+\exp(-\beta_0 - \beta_1 x)}.$$

Consequently,

$$\begin{aligned} pl(p) &= \sup\left\{pl(\beta) : \frac{1}{1+\exp(-\beta_0-\beta_1 x)} = p\right\} \\ &= \sup\left\{pl(\beta) : \beta_0 = \ln\frac{p}{1-p} - \beta_1 x\right\} \\ &= \sup_{\beta_1} pl\left(\ln\frac{p}{1-p} - \beta_1 x, \beta_1\right). \end{aligned}$$

We first write a function that computes $pl\left(\ln\frac{p}{1-p} - \beta_1 x, \beta_1\right)$:

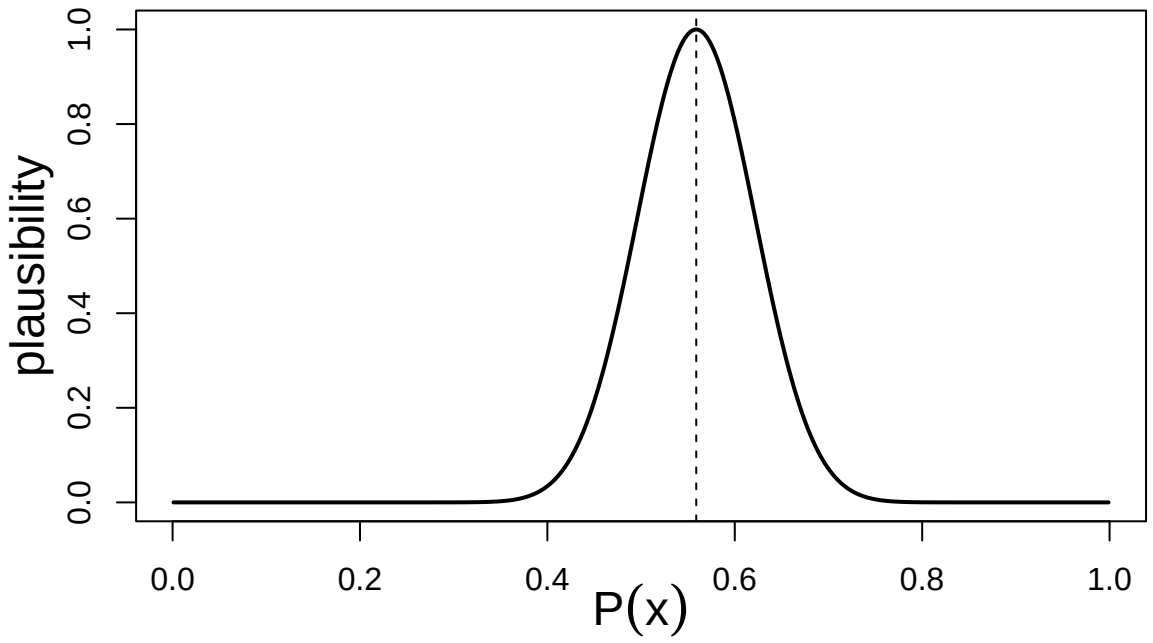
```
fonc_loglik <- function(beta1,x0,P,X,y,ellmax)
  rellik(beta=c(-log(1/P-1)-beta1%*%x0[-1],beta1),X,y,ellmax)
```

We then write a function that computes $pl(p)$, and its vectorized version:

```
pl_P1<-function(P,x,X,y,betah,delta=1){
  if((P==0)|(P==1)) pl<-0 else{
    opt<-optimize(f=fonc_loglik,interval=c(betah[2]-delta,betah[2]+delta),maximum=TRUE,
                  x0=x,P=P,X=X,y=y,ellmax=loglik(betah,X,y))
    pl<-opt$objective
  }
  return(pl)
}
pl_P<-Vectorize(pl_P1,vectorize.args="P")
```

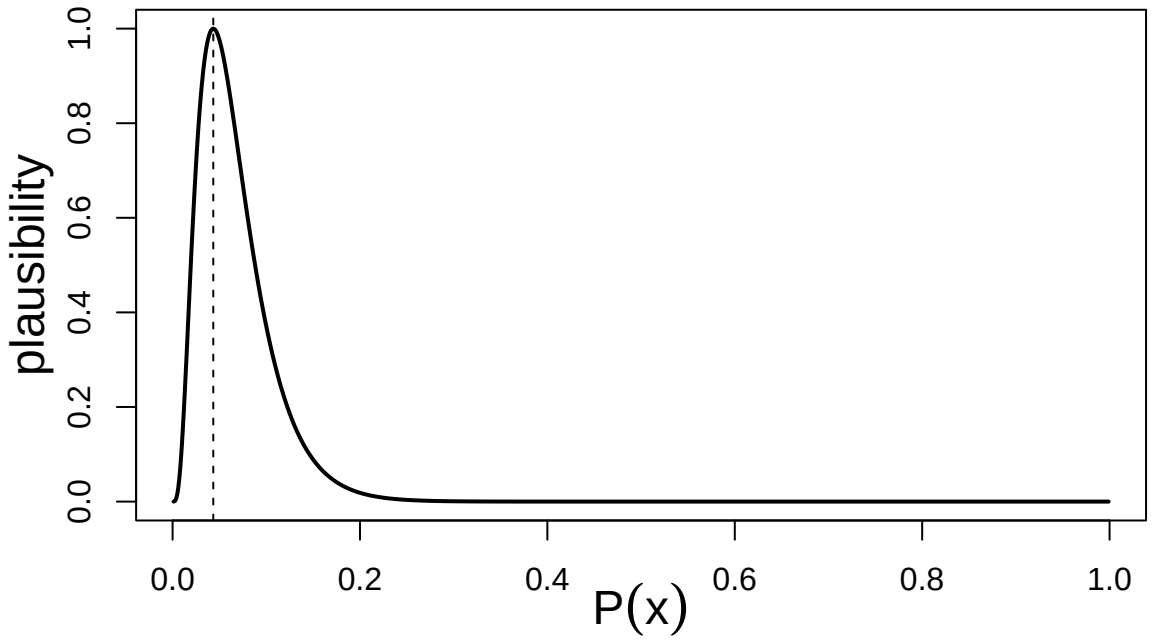
Let us plot $pl(p)$ as a function of p for $x = 50$:

```
x<-c(1,50)
P<-seq(0.001,0.999,0.001)
pl<-pl_P(P,x,X,y,betah)
plot(P,pl,type="l",xlab="",ylab="",lwd=2)
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(P(x)), line=2.2, cex.lab=1.5)
abline(v=1/(1+exp(-t(betah)%*%x)),lty=2)
```



Now, for $x = 20$:

```
x<-c(1,20)
pl<-pl_P(P,x,X,y,betah)
plot(P,pl,type="l",xlab="",ylab="",lwd=2)
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(P(x)), line=2.2, cex.lab=1.5)
abline(v=1/(1+exp(-t(betah)%*%x)),lty=2)
```



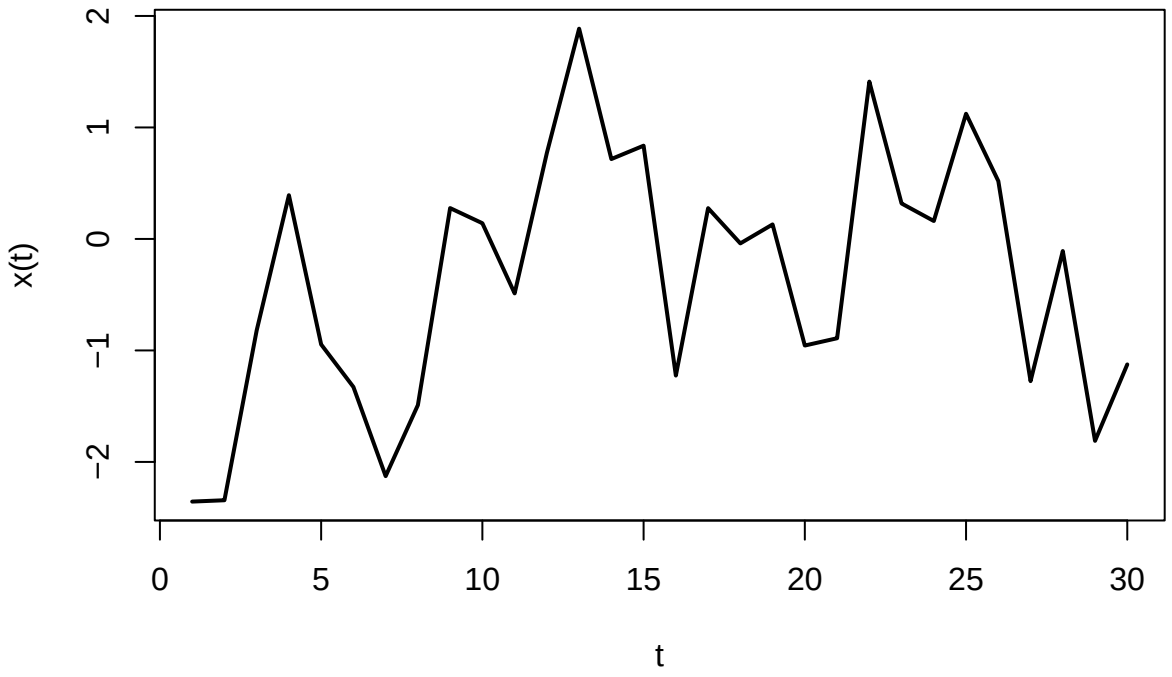
Question 4

Order-1 autoregressive process

Question 1

Let us generate a series of length $T = 30$ with $sigma = 1$ and $\rho = 0.7$:

```
set.seed(20231021)
sig<-1
rho <- 0.7
theta<-c(rho,sig)
T<-30
X<-rep(0,T)
X[1]<-rnorm(1,mean=0,sd=sig/sqrt(1-rho^2))
for(t in (2:T)) X[t]<-rnorm(1,mean=rho*X[t-1],sd=sig)
plot(X[1:30],type="l",xlab="t",ylab='x(t)', lwd=2)
```



Question 2

Given that the marginal distribution of X_1 is

$$X_1 \sim N\left(0, \frac{\sigma^2}{1-\rho^2}\right)$$

and the conditional distribution of X_t given $X_{t-1} = x_{t-1}$ is $X_t|x_{t-1} \sim N\left(\rho x_{t-1}, \sigma^2\right)$, the likelihood function after observing the first T terms of the sequence can easily be computed as

$$L(\theta; x_{1:T}) = \phi\left(x_1; 0, \frac{\sigma}{\sqrt{1-\rho^2}}\right) \prod_{t=2}^{T} \phi\left(x_t; \rho x_{t-1}, \sigma\right), \tag{1}$$

where $\theta = (\rho, \sigma)$ and $\phi(\cdot; \mu, \sigma)$ is the normal probability density function with mean μ and standard deviation σ. Here a function that computes the log-likelihood $\ell(\theta)$

```
loglik<-function(theta,x){
  rho<-theta[1]
  sig<-theta[2]
  T<-length(x)
  ell <- log(dnorm(x[1],0,sig/sqrt(1-rho^2)))+
         sum(log(dnorm(x[2:T],rho*x[1:(T-1)],sig)))
  return(ell)
}
```

To compute the relative likelihood, we first compute the MLE of θ:

```
theta0<-c(0,sd(X))
opt<-optim(theta0,loglik,method="L-BFGS-B",lower=c(-0.999,1e-6),
           upper=c(0.999,Inf),x=X,control=list(fnscale = -1))
thetah<-opt$par
ellmax<-opt$value
print(thetah)
```

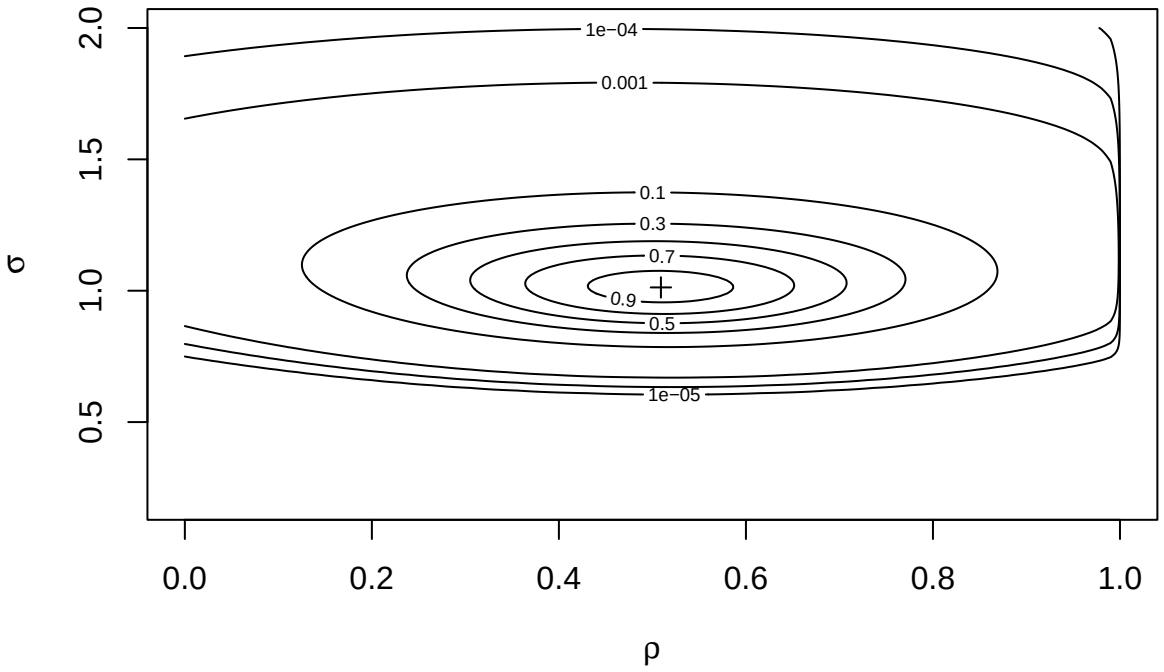
```
## [1] 0.5090863 1.0126383
```

The relative likelihhod can then be computed by the following function:

```
pl_lik<-function(theta,x,ellmax) exp(loglik(theta,x)-ellmax)
```

Let us plot the contours of this function:

```
xx<-seq(0,1,0.01) # rho
yy<-seq(0.2,2,0.01) # sigma
nx=length(xx)
ny=length(yy)
z12 <- matrix(0,nrow=nx,ncol=ny)
for(i in 1:nx) for(j in 1:ny) z12[i,j]=pl_lik(c(xx[i],yy[j]),X,opt$value)
contour(x=xx,y=yy,z12,xlab=expression(rho),ylab=expression(sigma),
        level=c(1e-5,1e-4,1e-3,seq(0.1,0.9,0.2)))
points(thetah[1],thetah[2],pch=3)
```



Question 3

Let $pl(\theta)$ denote the contour function (relative likelihood) of θ. The marginal contour functions (relative profile likelihoods) of ρ and σ are, respectively,

$$pl(\rho) = \sup_{\sigma} pl(\theta) \quad \text{and} \quad pl(\sigma) = \sup_{\rho} pl(\theta).$$

To compute these functions, we thus need to solve optimization problems. This function computes $pl(\theta)$ with the component `MARGIN` of β fixed at `theta1`:

```
rellik_fixed <- function(theta2,theta1,MARGIN,x,ellmax){
  theta<-c(0,0)
  theta[MARGIN]<- theta1
  theta[-MARGIN]<-theta2
  return(pl_lik(theta,x,ellmax))
}
```

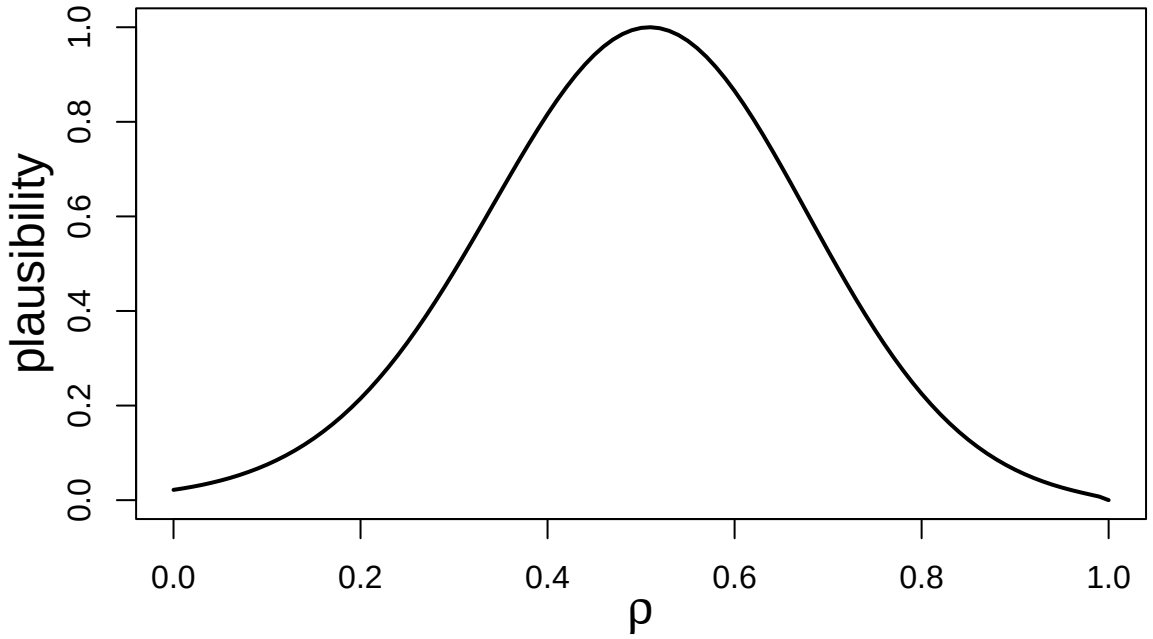
This function then computes the relative likelihood $pl(\rho)$ if `MARGIN=1` and $pl(\sigma)$ if `MARGIN=2`:

```
rellik_profile <- function(theta1,MARGIN,x,ellmax,theta20){
  # beta1 must be a scalar
  opt<-optim(par=theta20,fn=rellik_fixed,method="BFGS",
             control=list(fnscale=-1,maxit=10000),
             theta1=theta1,MARGIN=MARGIN,x=x,ellmax=ellmax)
  return(opt)
}
```

We note that an initial value `theta20` for the parameter that is varied must be provided.

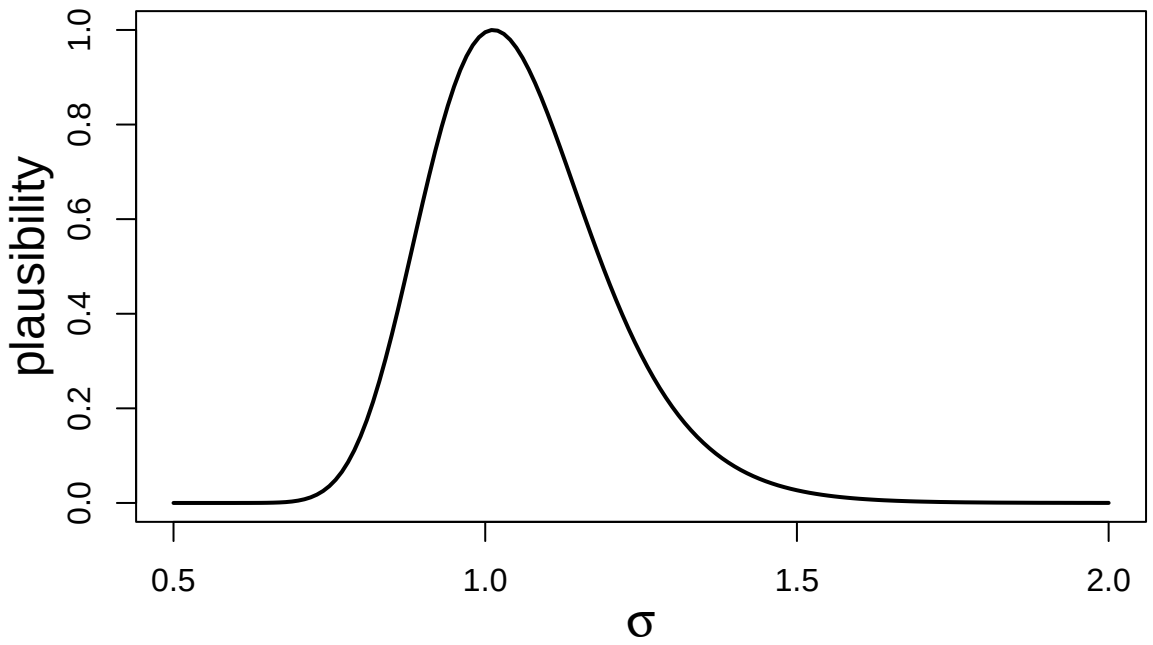
Let us now plot the marginal contour of ρ:

```
Theta0<-seq(0,1,0.01)
N<-length(Theta0)
pl <- rep(0,N)
i0<-(N+1)/2
theta20<-thetah[2]
for(i in i0:N){
  opt<-rellik_profile(theta1=Theta0[i],1,X,ellmax,theta20)
  pl[i]<-opt$value
  theta20<-opt$par
}
theta20<-thetah[2]
for(i in (i0-1):1){
  opt<-rellik_profile(theta1=Theta0[i],1,X,ellmax,theta20)
  pl[i]<-opt$value
  theta20<-opt$par
}
plot(Theta0,pl,type="l",lwd=2,xlab="",ylab="")
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(rho), line=2.2, cex.lab=1.5)
```



And the marginal contour of σ

```r
Theta1<-seq(0.5,2,0.01)
N<-length(Theta1)
pl <- rep(0,N)
i0<-(N+1)/2
theta20<-thetah[1]
for(i in i0:N){
  opt<-rellik_profile(theta1=Theta1[i],2,X,ellmax,theta20)
  pl[i]<-opt$value
  theta20<-opt$par
}
theta20<-thetah[1]
for(i in (i0-1):1){
  opt<-rellik_profile(theta1=Theta1[i],2,X,ellmax,theta20)
  pl[i]<-opt$value
  theta20<-opt$par
}
plot(Theta1,pl,type="l",lwd=2,xlab="",ylab="")
title(ylab="plausibility", line=2.2, cex.lab=1.5)
title(xlab=expression(sigma), line=2.2, cex.lab=1.5)
```



Question 4

To predict X_{T+1}, having observed $X_{1:T} = x_{1:T}$, we introduce the following φ-equation:

$$Y = \rho x_T + \sigma \Phi^{-1}(U) = \varphi_{x_{1:T}}(\theta, U),$$

where $Y \sim N(\rho x_T, \sigma^2)$ has the same distribution as X_{T+1} given $X_{1:T} = x_{1:T}$ and U has a standard uniform distribution.

Let $\Gamma(s) = \{\theta : pl(\theta) \geq s\}$. The predictive belief function on Y is defined y the random set

$$\varphi_{x_{1:T}}(\Gamma(S), U)$$

where (S, U) has a standard uniform distribution on $[0,1]^2$. It is a random interval $[V, W]$. To generate realizations of this random interval, we generate pairs (s, u) and solve the following constrained optimization problems:

$$\min_{\rho,\sigma} \rho x_T + \sigma \Phi^{-1}(u)$$

and

$$\max_{\rho,\sigma} \rho x_T + \sigma \Phi^{-1}(u)$$

such that $pl(\theta) \geq s$, $\rho \in (0,1)$ and $\sigma > 0$.

To solve these optimization problems, we will use function `constrOptim.nl` in package `alabama`. We first write R functions to compute the objective function and the constraints:

```
fun_pred<- function(theta){
  return(theta[1]*X[T]+theta[2]*qnorm(u))
}
logpl_cstr <- function(theta){
  L<-loglik(theta,X)
  return(c(L-ellmax-log(s),theta[2]-1e-6,1-abs(theta[1])-1e-6))
}
```

The following code generates $N = 100$ realization of the random interval $[V, W]$. (We should rather use $N = 1000$ or more to get a good approximation of the predictive belief function, but we choose a small value of N to reduce computation time). Note that we used a Halton sequence instead of random draws from the uniform distribution, for faster convergence:

```
library('alabama')
```

```
## Loading required package: numDeriv
```

```
library('randtoolbox')
```

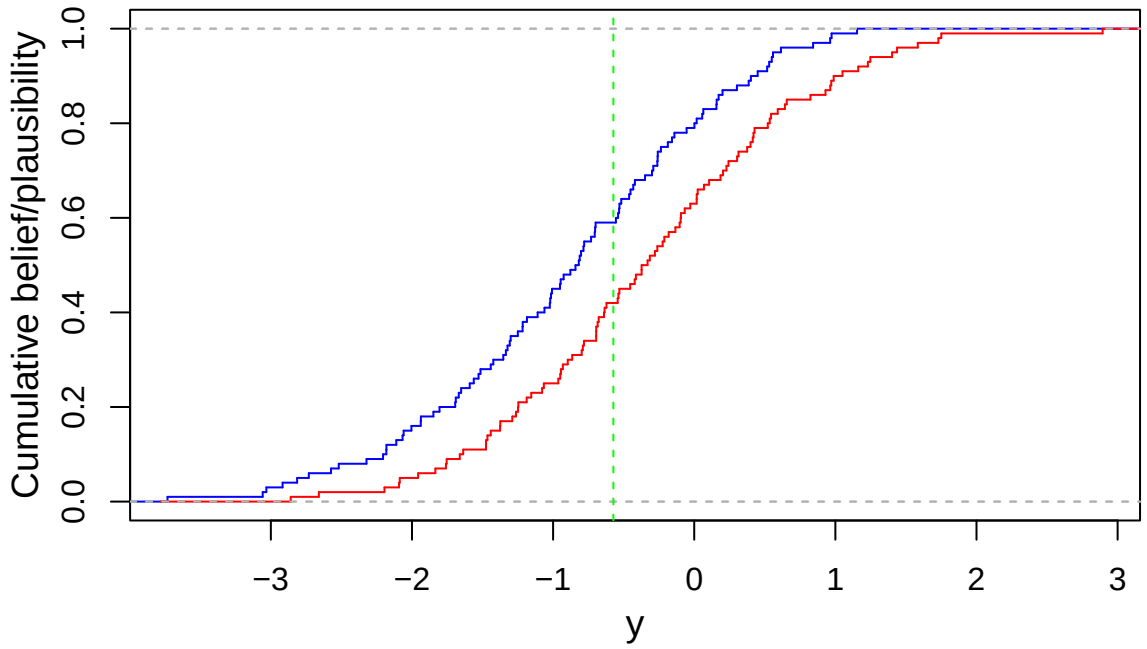
```
## Loading required package: rngWELL
```

```
## This is randtoolbox. For an overview, type 'help("randtoolbox")'.
```

```
N<-100
SW<-halton(n=N, dim = 2)
B<-matrix( nrow = N, ncol = 2)
for(i in (1:N)){
  s<- SW[i,1]
  u<- SW[i,2]
  opt_min<-constrOptim.nl(par=thetah,fn=fun_pred,hin =logpl_cstr,
                          control.outer=list(trace=0))
  opt_max<-constrOptim.nl(par=thetah,fn=fun_pred,hin =logpl_cstr,
                          control.outer=list(trace=0),
                          control.optim = list(fnscale=-1))
  B[i,]<-c(opt_min$value, opt_max$value)
}
```

We can now plot the lower and upper cdfs as:

```
plot(ecdf(B[,1]),do.points=FALSE, verticals=TRUE,xlab="",xlim=range(B),
     ylab="",main="",col="blue")
title(xlab="y",ylab="Cumulative belief/plausibility", line=2.2, cex.lab=1.2)
lines(ecdf(B[,2]),do.points=FALSE, verticals=TRUE,col="red")#,col="blue")
abline(v = thetah[1]*X[T],col="green",lty=2)
```



We can also plot the contour function as

```
y<-seq(min(B),max(B),0.01)
ny<-length(y)
pl<-rep(0,ny)
for(i in 1:ny) pl[i]<-mean((B[,1]<=y[i]) & (y[i]<=B[,2]))
plot(smooth.spline(y,pl),type="l",xlab="",ylab="")
title(xlab="y",ylab="plausibility", line=2.2, cex.lab=1.2)
abline(v = thetah[1]*X[T],col="red",lty=2)
```