Thank you Senthilvel.

The links were not very helpful, unfortunately.

But I think I solved the problem with help of the book "In all Likelihood" (Yudi Pawitan, Oxfort University Press, 2013). On page 94 there is in fact an example of getting the profile likelihood of a gamma sample (example 4.9). I tried this in R and get the same results.

I think one of my problems was a wrong parametrization of the gamma density.

Then I noticed that the confidence interval of the glm in R is based on the inversion of the likelihood-ratio test and, for mu, simply finds the values of the likelihood (at phi = phi.hat [= the MLE for phi]), not of the profile likelihood. This gives areas under the normalized (profile)likelihood that are lower than the given confidence level (i tried levels of for instance 0.9 and got areas of 0.83). This (the difference) seems to be the result of the chi-quare approximation (the larger the sample size the better is the match between desired conf.level and area under the norm. likelihood). I would be greatful if an expert could comment on this (am I right or wrong? how wrong? wrong in what way?...)

Here is the R code reproducing the results from example 4.9 in the book, showing the likelihood contour, the likelihood function for mu at phi.hat and the profile likelihood, plus the confidence limits obtained by glm:

y = c(23.91, 27.33, 0.15, 3.65, 5.99, 0.88, 0.93,0.53,0.17,14.17,6.18,0.05,3.89,0.24,0.08)

n = length(y)

# re-parametrized gamma desnity with mu and phi  
# scale (s) = phi\*mu  
# shape (a) = 1/phi  
# f(x, shape=a, scale=s) = 1/(s^a Gamma(a)) x^(a-1) e^-(x/s)  
# --> mu = scale/phi = scale\*shape

dgamma2 = function(x,mu,phi,...) dgamma(x, shape=1/phi, scale=phi\*mu, ...)

# fit the parameters of the distribution to get MLEs

fitted = MASS::fitdistr(y,dgamma2,start=list(mu=mean(y),phi=sd(y)/sqrt(n)))  
mu.hat = fitted$estimate["mu"]  
phi.hat = fitted$estimate["phi"]

# logLikelihood function

logL = function(mu,phi) sum(dgamma2(y, mu, phi, log=TRUE))  
logL = Vectorize(logL)

# Relative logLikelihood function

maxLogL = logL(mu.hat, phi.hat)  
logR = function(mu,phi) logL(mu,phi)-maxLogL  
logR = Vectorize(logR)

# Find the ranges for mu and phi to plot

mu.range = c(  
uniroot(function(mu) logR(mu, phi.hat)+4, c(min(y)/10, mu.hat))$root,  
uniroot(function(mu) logR(mu, phi.hat)+4, c(mu.hat,10\*max(y)))$root)  
phi.range = c(  
uniroot(function(phi) logR(mu.hat, phi)+4, c(1E-8, phi.hat))$root,  
uniroot(function(phi) logR(mu.hat, phi)+4, c(phi.hat,sd(y)))$root)

# relative Likelihood contour plot

mu = seq(mu.range[1], mu.range[2], len=100)  
phi = seq(phi.range[1], phi.range[2], len=100)  
mat = outer(mu,phi,logR)  
contour(mu,phi, exp(mat), xlab=bquote(mu), ylab=bquote(phi))  
abline(h=phi.hat, col=2, lty=3)  
abline(v=mu.hat, col=2, lty=3)

# profile logL

profileLogL = Vectorize(function(mu) {  
phi = optimize(function(phi) logL(mu,phi), phi.range, maximum=TRUE)$maximum  
logL(mu, phi)  
})  
  
# compare logL for phi=phi.hat and profileLogL

plot(function(mu) logL(mu,phi.hat), xlim=mu.range, xlab=bquote(mu), ylab="log relative Likelihood")  
plot(function(mu) profileLogL(mu), xlim=mu.range, col=2, add=TRUE)

# GLM

conf = 0.9  
model = glm(y~1, family=Gamma("identity"))  
glm.ci = confint(model, level=conf)  
abline(v=exp(coef(model)),lty=3)  
abline(v=glm.ci)  
abline(h= maxLogL-0.5\*qchisq(conf,1), lty=2) # cuts logL, not profileLogL

 Dec