

Likelihood theory: inferential results

(An overview)

N. Torelli, G. Di Credico, R. Macrì Demartino

Fall 2026

University of Trieste

Confidence intervals¹

Likelihood-based tests

Model selection

¹Agresti, Kateri: sec 5.7

Confidence intervals

Since the theory of MLE provides a general formula for standard errors, Wald-type confidence intervals for a parameter of interest ψ are generally available (here given for $1 - \alpha = 0.95$):

$$\hat{\psi} \pm 1.96 \text{SE}(\hat{\psi})$$

The asymptotic normality of the MLE justifies the usage of normal quantiles.

Actually, the availability of a general formula for $\text{SE}(\hat{\psi})$ when $\hat{\psi}$ is the MLE supports the widespread usage of this kind of confidence intervals.

Performance of Wald-type confidence intervals

The biggest issue with Wald-type confidence intervals is that **their accuracy depends on the chosen parametrization**.

Eventually, the MLE is approximately normally distributed, but for finite sample the parametrization matters.

(That's why methods which are invariant, such as percentile bootstrap confidence intervals, are preferable).

R lab: Wald-type CI for a variance

Let us assess the coverage probability for Wald-type intervals for σ^2 of a normal random sample.

```
M <- 100000; n <- 20; mat.ci <- matrix(NA, nrow = M, ncol = 2)
for(i in 1:M) { y <- rnorm(n, s=sqrt(2)); s2 <- var(y) * (n-1)/n
                se_s2 <- sqrt(2/n) * s2 * qnorm(0.975)
                mat.ci[i,] <- s2 + se_s2 * c(-1, 1)}
mean(mat.ci[,1] < 2 & mat.ci[,2] > 2)

## [1] 0.87048
```

```
M <- 100000; n <- 100; mat.ci <- matrix(NA, nrow = M, ncol = 2)
for(i in 1:M) { y <- rnorm(n, s=sqrt(2)); s2 <- var(y) * (n-1)/n
                se_s2 <- sqrt(2/n) * s2 * qnorm(0.975)
                mat.ci[i,] <- s2 + se_s2 * c(-1, 1)}
mean(mat.ci[,1] < 2 & mat.ci[,2] > 2)

## [1] 0.93269
```

R lab: Wald-type CI for σ^2

Things get better, given these parameter values, if we choose $\psi = \sigma$ and then re-transform the intervals:

```
M <- 100000; n <- 20; mat.ci <- matrix(NA, nrow = M, ncol = 2)
for(i in 1:M) { y <- rnorm(n, s=sqrt(2)); s2 <- var(y) * (n-1)/n
                se_s <- sqrt(s2 / (n * 2)) * qnorm(0.975)
                mat.ci[i,] <- (sqrt(s2) + se_s * c(-1, 1))^2}
mean(mat.ci[,1] < 2 & mat.ci[,2] > 2)
```

```
## [1] 0.89849
```

```
M <- 100000; n <- 100; mat.ci <- matrix(NA, nrow = M, ncol = 2)
for(i in 1:M) { y <- rnorm(n, s=sqrt(2)); s2 <- var(y) * (n-1)/n
                se_s <- sqrt(s2 / (n * 2)) * qnorm(0.975)
                mat.ci[i,] <- (sqrt(s2) + se_s * c(-1, 1))^2}
mean(mat.ci[,1] < 2 & mat.ci[,2] > 2)
```

```
## [1] 0.93817
```

There are also other approaches for confidence intervals based on the likelihood function.

They are based on **likelihood-based test statistics**, taking advantage of the relation existing between tests and confidence intervals, which is a general fact.

Likelihood-based tests

The likelihood ratio test

We saw that the likelihood ratio makes possible to choose between different parameter values. Therefore, it is not strange that the likelihood ratio can be used as test statistic, being in some sense the optimal choice, as supported by the **Neyman-Pearson lemma**.

Formally, the lemma is valid for choosing between two *simple hypotheses* $H_0 : \theta = \theta_0$ vs $H_1 : \theta = \theta_1$, for any pair of parameter values θ_0 and θ_1 .

The **likelihood ratio test statistic** is given by

$$\lambda(\mathbf{y}) = \frac{L(\theta_1)}{L(\theta_0)} = \frac{f_{\theta_1}(\mathbf{y})}{f_{\theta_0}(\mathbf{y})}$$

with rejection region

$$\mathcal{R}_\alpha = \{\mathbf{y} : \lambda(\mathbf{y}) \geq k_\alpha\},$$

being the test's *power* $\Pr_{\theta_1}\{\lambda(\mathbf{Y}) \geq k_\alpha\}$.

Three likelihood-based tests

We first focus on a simple one-parameter model, and on the problem of testing $H_0 : \theta = \theta_0$ vs $H_1 : \theta \neq \theta_0$.

The following three tests are available:

- The **likelihood ratio test (LRT)**

$$W(\theta_0) = 2 \{ \ell(\hat{\theta}) - \ell(\theta_0) \}$$

- The **Wald test**

$$W_e(\theta_0) = (\hat{\theta} - \theta_0)^2 J(\hat{\theta}) = \frac{(\hat{\theta} - \theta_0)^2}{\text{SE}(\hat{\theta})^2}$$

- The **score test**

$$W_u(\theta_0) = \frac{U(\theta_0)^2}{\mathcal{I}(\theta_0)}$$

In all the three cases, we reject H_0 for large values of the statistic, so that the p -value is (for instance) $p = \Pr_{\theta_0}(W \geq w_{obs})$.

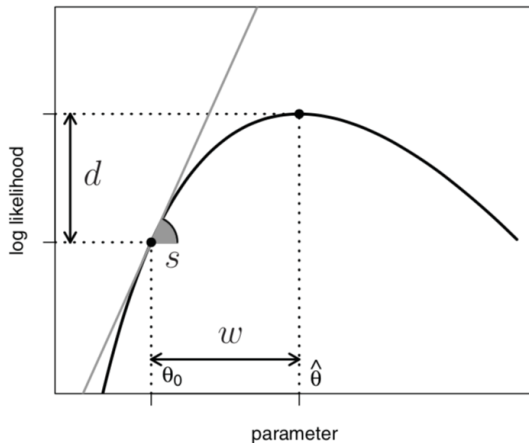


Figure 1. Comparing the three test statistics according to the traditional plot: Likelihood ratio is reported on the y scale, Wald on the x scale, and the score on the first derivative scale. The different scales do not favor understanding of the underlying connections.

Three likelihood-based tests: comments

- Whenever available, the exact distribution of these tests can be employed.
- Which one is preferable? The likelihood ratio test is clearly an obvious choice, but for large samples the three statistics are equivalent: this fact can be proved by a Taylor expansion of $U(\hat{\theta})$ around θ_0 .
- From the asymptotic distribution of the MLE, it readily follows that the null distribution of W_e is approximately

$$W_e(\theta_0) \dot{\sim} \chi_1^2$$

and since the two other tests are equivalent in large samples, the same result holds also for them.

- For one-sided alternatives such as $H_1 : \theta > \theta_0$, the signed squared-root versions of the test should be used, namely (for the LRT)

$$R(\theta_0) = \text{sgn}(\hat{\theta} - \theta_0) \sqrt{W(\theta_0)}$$

and, under H_0 , $R(\theta_0) \dot{\sim} \mathcal{N}(0, 1)$.

Confidence intervals based on W

The confidence interval based on W is particularly appealing: using the relation between confidence intervals and tests, it can be written as

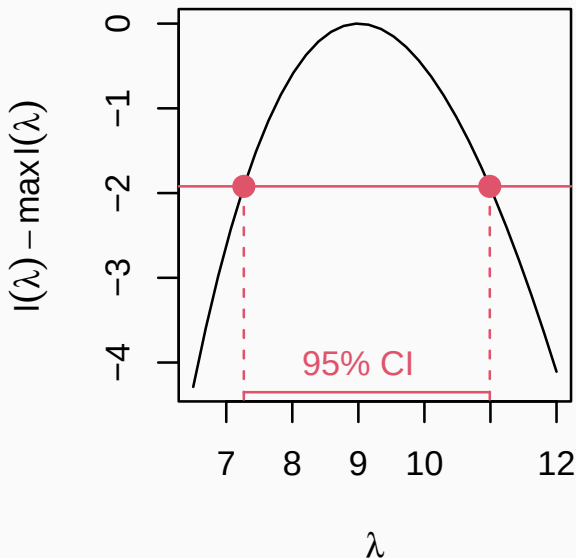
$$\{\theta : W(\theta) \leq \chi_{1;1-\alpha}^2\} = \left\{ \theta : \ell(\theta) \geq \ell(\hat{\theta}) - \frac{\chi_{1;1-\alpha}^2}{2} \right\}$$

so that the interval (which could actually be a union of intervals for multi-modal log likelihoods) includes all the parameter values with large log likelihood, i.e. the set of values most supported by the data.

The result does not depend on the parameterization (the chosen scale), differently from the Wald test.

R lab: visualizing the confidence interval based on the LRT

Back to the Poisson example (with $n = 10$ and $\sum_i y_i = 90$):



Parameter of interest and nuisance parameters

The three tests introduced readily generalize to hypotheses on the entire p -dimensional parameter θ . For instance, the LRT would become

$$W(\theta_0) = 2 \{ \ell(\hat{\theta}) - \ell(\theta_0) \}$$

with asymptotic null distribution given by χ_p^2 .

At any rate, the typical (and most interesting) situation is where we wish to test an hypothesis on a q -dimensional subset of θ , with $q < p$.

Following the CS book, we write $\theta^\top = (\psi^\top, \gamma^\top)$, with the null and alternative hypotheses given by $H_0 : \psi = \psi_0$ vs $H_1 : \psi \neq \psi_0$.

Here ψ is denoted as the **parameter of interest** and γ is the **nuisance parameter**.

The profile likelihood*

Likelihood theory handles nuisance parameters by introducing the **profile likelihood**.

Denoted by $\hat{\gamma}_{\psi}$ the MLE of γ for fixed value of ψ , namely

$$\hat{\gamma}_{\psi} = \operatorname{argmax}_{\gamma \in \Gamma} \ell(\psi, \gamma)$$

then we define the profile likelihood for ψ as

$$L_P(\psi) = L(\psi, \hat{\gamma}_{\psi}).$$

Note that the maximum of $L_P(\psi)$ is given by the MLE of ψ .

Inference based on the profile likelihood*

A crucial point is the large-sample properties of the profile likelihood are **those of a bona-fide likelihood function** for the parameter of interest only.

In particular, the profile likelihood LRT

$$W_P(\psi) = 2 \{ \ell_P(\hat{\psi}) - \ell_P(\psi_0) \}$$

the asymptotic null distribution is given by χ_q^2 .

Note, however, that if the dimension of γ is large, the large-sample results may be poor. In such cases, the parametric bootstrap is a more accurate route to obtain the p -value.

The t -test as a likelihood-based method*

Many noteworthy tests can be derived from the LRT based on the profile likelihood.

A very important instance is the t -test on μ for a normal random sample, $Y_i \sim \mathcal{N}(\mu, \sigma^2)$. With some simple algebra

$$\ell_P(\hat{\mu}) - \ell_P(\mu_0) = \frac{n}{2} \log(\hat{\sigma}^2) - \frac{n}{2} \log(\hat{\sigma}_{\mu_0}^2),$$

and since $\hat{\sigma}_{\mu}^2 = \hat{\sigma}^2 + (\hat{\mu} - \mu)^2$, it follows

$$r_P(\mu_0) = \text{sgn}(\hat{\mu} - \mu_0) \sqrt{n \log \left\{ 1 + \frac{(\hat{\mu} - \mu_0)^2}{\hat{\sigma}^2} \right\}}.$$

Further simple algebra shows that $R_P(\mu_0)$ is a monotonic increasing function of the T test statistic $T(\mu_0) = (\bar{y} - \mu_0)/\sqrt{s^2/n}$, so that, for instance, $\Pr_{H_0}\{R(\mu_0) \geq r_{obs}\} = \Pr_{H_0}\{T(\mu_0) \geq t_{obs}\}$.

Other notable instances

Several other tests can be derived as special cases of the LRT, such as the F test for one-way anova models, or exact tests employed in linear regression models.

Other famous tests are instead special cases of the score test. The most notable instance is the chi-squared test of independence for two-way contingency tables, and related tests. The underlying statistical model is the **multinomial distribution** for the observed frequencies.

The generalised likelihood ratio statistic

In broad generality, for testing $H_0 : \theta \in \Theta_0$ vs $H_1 : \theta \in \Theta_1$ the LRT is most natural resolution

$$W(H_0) = 2 \{ \ell(\hat{\theta}) - \ell(\hat{\theta}_{H_0}) \}.$$

In broad generality, parametric bootstrap is the most convenient approach to approximate the null distribution and compute the p -value.

An approximate (large-sample) null distribution exists when H_0 can be expressed as

$$H_0 : \mathbf{R}(\theta) = 0$$

where $\mathbf{R}$ is a vector-valued function of θ that imposes r restrictions on the parameter vector. In such case, under the null

$$W(H_0) \dot{\sim} \chi_r^2.$$

Model selection

Choosing the best model

Several statistical tests are applied for choosing between two alternative specifications of a statistical model. For this sort of problem, more suitable techniques are available, which can also be extended to settings where the two models are not *nested* (i.e. one model is a special instance of the other).

The **Akaike's Information Criterion (AIC)** is perhaps the most commonly used method for choosing the *best* model.

A useful starting point is the **Kullback-Leibler divergence** between the true model f_t and the model under consideration

$$K(f_{\hat{\theta}}, f_t) = \int_{\mathcal{Y}} \{\log f_t(\mathbf{y}) - \log f_{\hat{\theta}}(\mathbf{y})\} f_t(\mathbf{y}) d\mathbf{y}$$

Selecting models to minimize (an estimate of) the expected value of K is equivalent to selecting the model that has the lowest value of

$$\text{AIC} = -2 \ell(\hat{\boldsymbol{\theta}}) + 2p$$

with $p = \dim(\boldsymbol{\theta})$.

Model selection based on AIC: comments

- The point is that we cannot select the model based on the log likelihood only, since it always selects the more complex model. AIC overcomes this problem by a penalty for adding parameters.
- The **AIC is not consistent**: as $n \rightarrow \infty$, the probability of selecting the correct model does not tend to 1. Indeed, at least for nested models, twice the drop in the maximized log likelihood between an overly complex model and the true model follows (approximately) a χ_r^2 distribution. Since neither χ_r^2 nor $p/2$ depends on n , the probability of selecting the overly complex model by AIC is nonzero and independent of n (for n large).
- The practical implications of the previous point are less serious than it may seem: if all the models under consideration are wrong, then we will tend to select increasingly complex specifications as the sample size increases and the predictive disadvantages of complexity diminish.

R lab: annual mean temperatures in New Haven

We return on the example employed to introduce Statistical Models, and compute the AIC for the four proposed models.

```
y <- (nhtemp - 32) / 1.8; x <- 1912:1971-1
AIC.vals <- rep(NA, 4)
mle1 <- fitdistr(y, "normal")
AIC.vals[1] <- -2 * mle1$loglik + 2 * 2
mle2 <- fitdistr(y, "t", df = 5)
AIC.vals[2] <- -2 * mle2$loglik + 2 * 2
mle3 <- lm(y ~ x)
AIC.vals[3] <- AIC(mle3)
mle4 <- arima(y, xreg=x, order=c(1, 0, 0))
AIC.vals[4] <- AIC(mle4)
AIC.vals

## [1] 130.9961 130.3981 114.9645 116.2789
```

AIC as an alternative to Cross Validation

As stated in the CS book, an alternative approach starts from observing that the KL divergence only depends on the model via
– $\int_{\mathbf{y}} \log f_{\hat{\theta}}(\mathbf{y}) f_t(\mathbf{y}) d\mathbf{y}$, where the expectation is taken over data not used to estimate $\hat{\theta}$.

An obvious direct estimator of this is the **cross-validation score**

$$\text{CV} = - \sum_i \log f_{\hat{\theta}^{[-i]}}(y_i)$$

where $\hat{\theta}^{[-i]}$ is the MLE based on the data with y_i omitted. We might multiply it by 2 to obtain something on the same scale of the AIC.

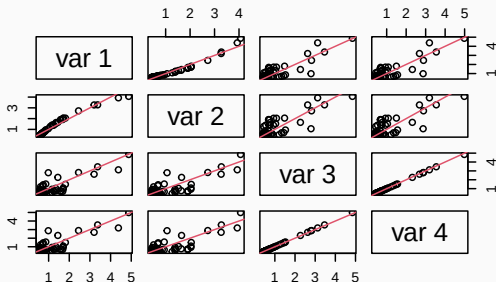
This estimates directly the **predictive accuracy** of the model, and it is a central quantity of *statistical learning methods*. Variants exist where more than one data point at a time are omitting from fitting, with 5 – 10 groups (*folds*) being a common choice. Clearly, the AIC is a much faster alternative.

R lab: CV scores for the example

```
n <- length(y); mat.CV1 <- matrix(0, nrow=n, ncol=4)
for(i in 1:n){
  mle1 <- fitdistr(y[-i], "normal")
  mat.CV1[i,1] <- -log(dnorm(y[i], mle1$est[1], mle1$est[2]))
  mle2 <- fitdistr(y[-i], "t", df = 5)
  mat.CV1[i,2] <- -log(dt((y[i] - mle2$est[1]) / mle2$est[2],
                        df = 5)) + log(mle2$est[2]))
  mle3 <- lm(y[-i] ~ x[-i])
  mui <- mle3$coef[1] + mle3$coef[2] * x[i]
  si <- summary(mle3)$sigma
  mat.CV1[i,3] <- -log(dnorm(y[i], mui, si))
  mle4 <- arima(y[-i], xreg = x[-i], order = c(1, 0, 0))
  mui <- mle4$coef[2] + mle4$coef[3] * x[i]
  si <- sqrt(mle4$sigma2 / (1 - mle4$coef[1]^2))
  mat.CV1[i,4] <- -log(dnorm(y[i], mui, si))
}
```

R lab: CV scores for the example

```
my_line <- function(x,y){points(x,y); abline(a=0, b=1, col=2)}  
pairs(mat.CV1, panel = my_line)
```



```
colSums(mat.CV1) * 2
```

```
## [1] 131.9519 130.5076 115.9256 116.1472
```