Akaike Information Criterion

$$AIC = 2k - 2\ln L$$

k - number of model parameters

L - likelihood function of the estimated model.

Given a set of candidate models, the one with minimized AIC is the preferred one.

1) Penalizes model complexity
2) Rewards "goodness of fit" (maximizing likelihood).

It is founded in the information theory.

Let data be generated by a process f, and let g_1 and g_2 be two competing models.

Information lost by using g_i is evaluated by the Kullback-Leibler's divergence $D_{KL}(f, g_i)$

We should choose the model that minimizes the information loss. In 1974, Akaike showed that "how much extra information" is lost when we use one model or the other can be obtained using AIC.

Let us have R models and their AIC criteria evaluate to $AIC_1, AIC_2, \dots, AIC_R$, and let minimal of those be AIC_{min}. The corresponding model can be seen as

pp. 2

$$e^{(AIC_i - AIC_{min})/2}$$

times more likely model than model i.

Let's have models evaluating to

$$AIC_1 = 100; \quad AIC_2 = 102; \quad AIC_3 = 110$$

Second model is $e^{-\frac{(102-100)}{2}} = 0.368$ times as likely to generate the data as model 1.

Third model is $e^{-\frac{110-100}{2}} = 0.007$ times as likely to generate the data than model 1.

One can use this to even Bayesian-ly mix models if their AIC-s are relatively close.

Please note this is all valid asymptotically.

Note 1) Founded in information theory

Note 2) Only differences are meaningful

Note 3) For Gaussian processes, ln L is related to the residual sum of squares

$$AIC \sim 2k + \chi^2$$

Original paper

H. Akaike, "A new look at the statistical model identification", IEEE Tr. on Automatic Control, Vol 19, No. 6, 716-723, 1974

When we only have a residual sum of squares, the AIC becomes:

$$AIC = n \cdot \ln \frac{RSS}{n} + 2k + \underset{\uparrow}{C}$$

not important

Obtained by reducing the Likelihood function to

$$\ln L = C_1 - \frac{n}{2} \ln \frac{RSS}{n}$$

pp. 3.

n - number of samples.

Estimation of Model Parameters

(rough outline → not pursued in detail since we do not want to get into optimization)

X_t-s given $\xrightarrow{?}$ ϕ-s, θ-s & $\sigma_a^2 = ?$

If we have AR(n) model, things are easy

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \dots + \phi_n X_{t-n} + a_t \qquad a_t \sim NIID(0, \sigma_a^2)$$

$$\underset{\underline{Y}}{\begin{bmatrix} X_{n+1} \\ X_{n+2} \\ \vdots \\ X_N \end{bmatrix}} = \underbrace{\begin{bmatrix} X_n & X_{n-1} & \cdots & X_1 \\ X_{n+1} & X_n & \cdots & X_2 \\ - & - & - & - \\ X_{N-1} & \cdots & & X_{N-n} \end{bmatrix}}_{\underline{X}} \underset{\vec{\phi}}{\begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{bmatrix}} + \underset{\vec{\varepsilon}}{\begin{bmatrix} a_{n+1} \\ a_{n+2} \\ \vdots \\ a_N \end{bmatrix}}$$

$$\hat{\phi} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{Y}$$

$$\hat{\sigma}_a^2 = \frac{1}{N-2n} \sum_{t=n+1}^{N} (X_t - \hat{\phi}_1 X_{t-1} - \dots - \hat{\phi}_n X_{t-n})$$

unbiased

$\frac{1}{N-n}$ ← biased, but min variance!

– ARMA (n, n-1) model

$$X_t - \phi_1 X_{t-1} - \dots - \phi_n X_{t-n} = a_t - \theta_1 a_{t-1} - \dots - \theta_{n-1} a_{t-n+1}$$

We need to find ϕ_s^* and θ-s that will minimise residual sum of squares

$$a_t = X_t - \phi_1 X_{t-1} - \dots - \phi_n X_{t-n} + \theta_1 a_{t-1} + \dots + \theta_{n-1} a_{t-n+1}$$

$$a_{t-1} = X_{t-1} - \phi_1 X_{t-2} - \dots - \phi_n X_{t-n-1} + \theta_1 a_{t-2} + \dots + \theta_{n-1} a_{t-n}$$

$$a_{t-2} = \dots$$

Substitution of $a_{t-1}, a_{t-2}, \dots, a_0$ into (*) gives us a non-linear optimiz. problem

Basically, each combination of $\vec{\phi}$-s and $\vec{\theta}$-s will give us some $RSS(\vec{\phi}, \vec{\theta})$. We need to find $\hat{\vec{\phi}}, \hat{\vec{\theta}}$ that will minimize the RSS

$$RSS(\vec{\phi}, \vec{\theta}) = \sum_{t=1}^{N} a_t^2$$

(usually you assign $a_0 = a_1 = \dots = a_{n-1} = 0$ or you can include them into optimization)

Marquard's method, Steepest descent, etc → all can be used to descend down the RSS curve!

This is a multi-modal problem & getting close to the solution is very important!

"armax" command from Matlab executes optimization used in my code "PostulateARMA"

$\hat{\vec{\phi}} \pm \Delta\phi \quad \hat{\vec{\theta}} \pm \Delta\vec{\theta}$ is given, where $\pm\Delta$-s come from local linear approximations of the problem (Jacobians take over the role of $\underline{X}$).

It is possible to define a problem like

$$(\mathring{X}_t - \mu) - \phi_1(\mathring{X}_{t-1} - \mu) - \dots - \phi_n(X_{t-n} - \mu) =$$

$$= a_t - \theta_1 a_{t-1} - \dots - \theta_{n-1} a_{t-n+1}$$

and estimate μ-s, θ-s and ϕ-s, but ARMAX doesn't do it $\rightarrow$ you need to estimate $\mu = \bar{X}_t$ and subtract it.

<u>How to get "close" to a solution?</u>

<u>Initial Guess</u>

One way (analytically tractable & easy to understand) can be to use I.F. approximation.