STAT 5170: Applied Time Series

Course notes for part B of learning unit 4

Section 4B.1: Markov chain Monte Carlo simulation.

We have seen that a Bayesian approach to inference, especially when implemented by numerical simulation algorithms, offers notable flexibility for insightful data-analysis. We have seen how a Gibbs sampling strategy offers an added advantage for managing multi-parameter models. However, in all of the examples thus far discussed, values are simulated directly from a canonical distribution, such as the beta, normal, or χ^2 distribution. A necessity to simulate from canonical distributions could be a limitation for Bayesian methods, if not for the availability of a much more general strategy, which, in theory, is capable of generating a simulated, representative sample from any distribution, using only the form of its density. That is, to sample from, say, a posterior distribution with density $p(\boldsymbol{\theta}|\boldsymbol{x})$, the form of this density needs only to be known up to proportional equivalence; a simulated sample may be generated using only the numerator of Bayes's formula, $p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}|\boldsymbol{x})$.

The general sampling strategy alluded to above is called the *Metropolis-Hastings algorithm*. It is an example of what is known as a *Markov chain Monte Carlo* or *MCMC algorithm*. The "Monte Carlo" portion of this label is the name of a city that is famous for gambling, and serves only as a fancy metaphor for simulation. The "Markov chain" portion, however, presents an interesting twist on simulation algorithms. A Markov chain is, in fact, a type of time series model. It is characterized by the property that the dependency of any one value on previous values in the Markov chain is determined by its dependency on the value one time lag before. Manifested as an iterative sampling algorithm, it works out in the following way:

Suppose the algorithm aims to simulate a sample from a posterior distribution with density $p(\boldsymbol{\theta}|\boldsymbol{x})$, and is generating values $\boldsymbol{\theta}^{(0)}, \boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots$, where $\boldsymbol{\theta}^{(0)}$ denotes the starting value and $\boldsymbol{\theta}^{(k)}$ is the simulated value at iteration k. The Markov chain property translates to mean that the simulated value $\boldsymbol{\theta}^{(k)}$ may depend on the values simulated in past iterations, $\boldsymbol{\theta}^{(0)}, \ldots, \boldsymbol{\theta}^{(k-1)}$, only through the value simulated in the iteration immediately before, $\boldsymbol{\theta}^{(k-1)}$.

Take note of the following ideas in particular:

- An MCMC algorithm does not generate a simulated sample of independent and identically distributed values. However, if it is designed well and implemented with care, it will generate a simulated sample of *dependent, but approximately identically distributed* values.

- The values simulated by a (well designed and carefully implemented) MCMC algorithm follow the usual relative-frequency properties that are familiar to us from working with independent and identically distributed random samples. For example, if $\boldsymbol{\theta}^{(0)}, \ldots, \boldsymbol{\theta}^{(m-1)}$ is a sample that is generated from an MCMC algorithm designed to simulate representative samples from a posterior distribution with density $p(\boldsymbol{\theta}|\boldsymbol{x})$, then a numerical approximation to the posterior expectation of the function $g(\boldsymbol{\theta})$ is

$$E[g(\boldsymbol{\theta})|\boldsymbol{x}] \approx \frac{1}{m}\sum_{k=0}^{m-1} g(\boldsymbol{\theta}^{(k)}).$$

This implies that all such quantities as posterior expectations, probabilities, and quantiles are approximated from their sample analogues calculated on the simulated sample.

One important difference from the independent and identically distributed sampling case is that the accuracy of approximation is reduced in the presence of dependency, meaning that longer sequence of values needs to be simulated in an MCMC algorithm to produce the same level of accuracy.

- In order to reduce the impact of the algorithm's starting value, $\boldsymbol{\theta}^{(0)}$, which can be substantial if the value is not carefully selected, a common strategy is to discard the $\boldsymbol{\theta}^{(k)}$ simulated in an initial set of iterations that comprise a *burn-in* phase of the algorithm. For example, if the burn-in phase ends at iteration $b-1$, then the numerical approximation above is modified to

$$E[g(\boldsymbol{\theta})|\boldsymbol{x}] \approx \frac{1}{m-b}\sum_{k=b}^{m-1} g(\boldsymbol{\theta}^{(k)}).$$

- The duration of the burn-in period would be determined by *monitoring* the sequence of values that are generated by the MCMC algorithm. Using any of a variety of graphical and numerical *monitoring diagnostics*, the burn-in phase would be declared complete once the random patterns in the sequence begin to settle into an equilibrium. Specialized monitoring diagnostics are also available to check the MCMC algorithm for deficiencies that would reduce its efficiency or prevent it from producing reliable numerical approximations.

Suppose we wish to simulate a sample from a posterior distribution with density $p(\boldsymbol{\theta}|\boldsymbol{x})$, and we only know, $f(\boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$, the numerator of Bayes's formula, for which $f(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}|\boldsymbol{x})$. To generate the required sample, the Metropolis-Hastings algorithm operates according to the following steps:

Step 1: At iteration k the value simulated in the previous iteration is $\boldsymbol{\theta}^{(k-1)}$. A candidate value $\boldsymbol{\gamma}$, which may eventually be assigned to $\boldsymbol{\theta}^{(k)}$, is generated by directly sampling from a distribution from which it is easy to directly sample. This distribution would depend on the previous value $\boldsymbol{\theta}^{(k-1)}$, making it a conditional distribution. It is called a *proposal distribution*, and its conditional density function is denoted $q(\boldsymbol{\gamma}|\boldsymbol{\theta})$.

Step 2: It is to be determine whether the candidate value $\boldsymbol{\gamma}$ is assigned to $\boldsymbol{\theta}^{(k)}$. To make this determination, start by evaluating the *acceptance probability formula*,

$$\alpha = \min\left\{1, \frac{f(\boldsymbol{\gamma})q(\boldsymbol{\theta}|\boldsymbol{\gamma})}{f(\boldsymbol{\theta})q(\boldsymbol{\gamma}|\boldsymbol{\theta})}\right\},$$

having written $\boldsymbol{\theta}$ for $\boldsymbol{\theta}^{(k-1)}$ out of convenience. A little intuition about this formula is provided below.

STEP 3: Simulate a value u from a uniform distribution on the unit interval (*i.e.*, $0 < u < 1$). If $u \leq \alpha$, then assign $\boldsymbol{\theta}^{(k)} = \boldsymbol{\gamma}$; otherwise, discard $\boldsymbol{\gamma}$ and assign $\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)}$, in which case the simulated parameter at iteration k is a duplicate of that at iteration $k-1$.

For intuition into the sensibility of the acceptance probability formula, consider the following. In some scenarios the proposal distribution is symmetric in the sense that $q(\boldsymbol{\gamma}|\boldsymbol{\theta}) = q(\boldsymbol{\theta}|\boldsymbol{\gamma})$, which implies that the acceptance probability formula simplifies to $\alpha = \min\{1, f(\boldsymbol{\gamma})/f(\boldsymbol{\theta})\}$. Observe that when $\boldsymbol{\gamma}$ falls into a region of higher posterior density, relative to the region occupied by $\boldsymbol{\theta}$, then $f(\boldsymbol{\gamma}) > f(\boldsymbol{\theta})$ and $\boldsymbol{\gamma}$ would be assigned to $\boldsymbol{\theta}^{(k)}$ than $\boldsymbol{\theta}$ with probability one. On the other hand, a poor candidate value would have $f(\boldsymbol{\gamma}) < f(\boldsymbol{\theta})$ and would possibly be discarded rather than assigned. A rigorous proof that the Metropolis-Hastings algorithm produces a sample from the desired distribution requires the use of mathematical concepts well beyond the scope of the course.

A demonstration of the Metropolis-Hastings algorithm, and introduction to the issue of tuning the algorithm for efficient implementation, is provided in the next example.

Example: Binomial-beta models

Recall a previous example in which the data-generating distribution is $X|\theta \sim \text{binomial}(n, \theta)$ and the prior distribution is $\theta \sim \text{beta}(\alpha, \beta)$. In our previous exploration of this set up, we drew on mathematical results identifying the posterior distribution as $\theta|x \sim \text{beta}(\alpha + x, \beta + n - x)$. By turning to functionality built-in to our computer-software that allows us to directly simulate from the beta distributions, we observed that it is straightforward to simulate a sample from the posterior distribution, and calculate a variety of posterior summaries.

Now we revisit this example to consider how that simulation could have been carried out (less efficiently) using a Metropolis-Hastings algorithm. As indicated above, in complicated problems a potential advantage of using this algorithm is that we need only know the posterior density up to proportional equivalence. Recall that the form of the posterior density is

$$p(\theta|x) \propto p(x|\theta)p(\theta) \propto \theta^{\alpha+x-1}(1-\theta)^{\beta+n-x-1},$$

which suggests we may set $f(\theta) = \theta^{\alpha+x-1}(1-\theta)^{\beta+n-x-1}$ in the acceptance probability formula.

Use of the Metropolis-Hastings algorithm also requires that we select a proposal distribution. The requirements of this distribution are (*i.*) that it is easy to simulate from, and (*ii.*) that it generates candidate values for γ within the range of the possible parameter values, which is the unit interval, $0 < \gamma < 1$. We will discuss *preferred* choices of the proposal distribution later, but these are the requirements.

A routine for simulating a candidate value, γ, that satisfies $0 < \gamma < 1$ by way of a symmetric proposal distribution is as follows.

Step 1: At iteration k, denote by θ the parameter value simulated in the previous iteration. Fix a positive constant $c < 1$, and simulate a value u from a uniform distribution between $-c$ and c (*i.e.*, $-c < u < c$).

Step 2: Assign the candidate parameter value according to

$$\gamma = \begin{cases} -(\theta + u) & \text{if } \theta + u < 0 \\ \theta + u & \text{if } 0 \leq \theta + u \leq 1 \\ 1 - (\theta + u - 1) & \text{if } \theta + u > 1 \end{cases}$$

Interpreting these steps, the algorithm works by simulating a symmetric shift of θ, followed by checking whether the value falls outside the unit interval; if it does, the value is reflected back into the interval by the same distance it fell outside of it. This type of proposal distribution is sometimes called a *uniform, symmetric shift with reflecting boundaries*. Its density is guaranteed to satisfy $q(\gamma|\theta) = q(\theta|\gamma)$, which implies that the acceptance probability formula simplifies by cancellation of these terms. Incidentally, when the parameter space has just one boundary, the same strategy of reflecting a shifted value that falls outside of the space back in by the same distance it fell out also yields a symmetric proposal distribution.

The quantity c is called a *tuning parameter.* The value selected for c is irrelevant to the theoretical capacity of the Metropolis-Hastings algorithm to generate a sample that is representative of the posterior distribution; any choice of c will do. Nevertheless, that choice is very influential to the performance of the algorithm itself. A poor choice of c would require that the algorithm execute over a greater number of iteration to produce the same numerical accuracy as would be required for a better choice of c.

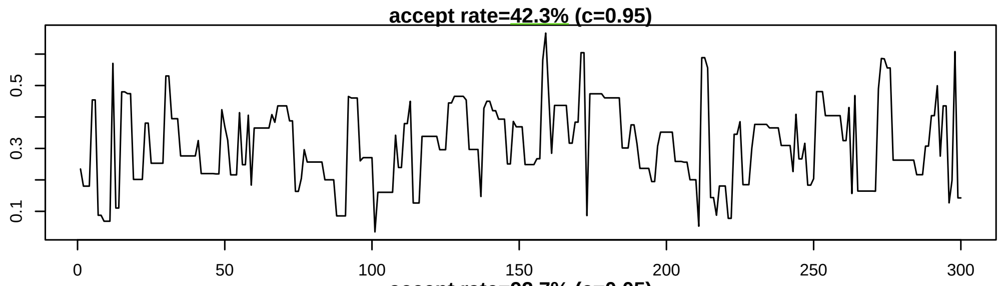
There is no precise formula for optimal choice of c that would maximize the algorithm's efficiency. However, a way to check check whether the algorithm is running at a suitable level of efficiency is to monitor the relative frequency at which candidate values, γ, are accepted as new values of θ in the current iteration. A commonly applied informal rule is that c is to be selected so the candidate values are accepted somewhere between 25% and 40% of the time. If the acceptance rate is too small, it would seem that the proposal distribution tends to produce candidate values that fall outside a region of high posterior probability, suggesting that c is too big. If the acceptance rate is too big, it would seem that the proposal distribution tends to produce candidate values that barely differ from the current parameter value, suggesting that c is too small.

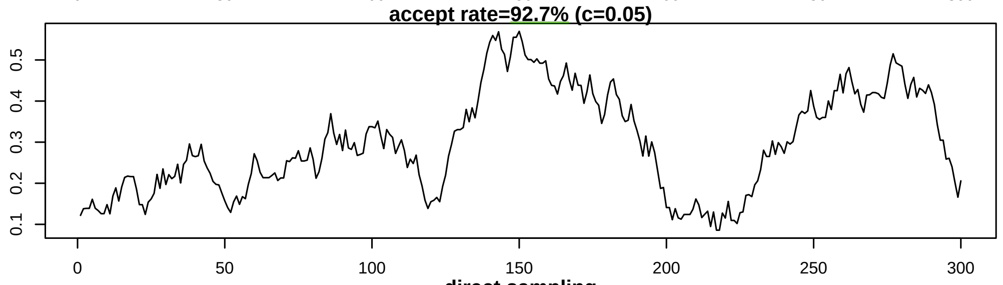
The following results are produced by separate executions of a Metropolis-Hastings algorithm using distinct values of c. In both cases, the data and prior distributions are the same: $x = 3$ incidences are observed in $n = 10$ trials, and the parameters of the prior distribution are $\alpha = 1$ and $\beta = 1$. The table below lists the posterior means and standard deviations, and a set of quantiles calculated from the algorithm output, along with the acceptance rate of candidate proposals. In each execution, the starting value is $\theta^{(0)} = 0.3$, the burn-in phase ends at iteration $b - 1 = 250$, and the final iteration number is $m - 1 = 300$. The table's bottom row lists values that are directly calculated from the posterior distribution, $\theta|x \sim \text{beta}(4, 8)$.

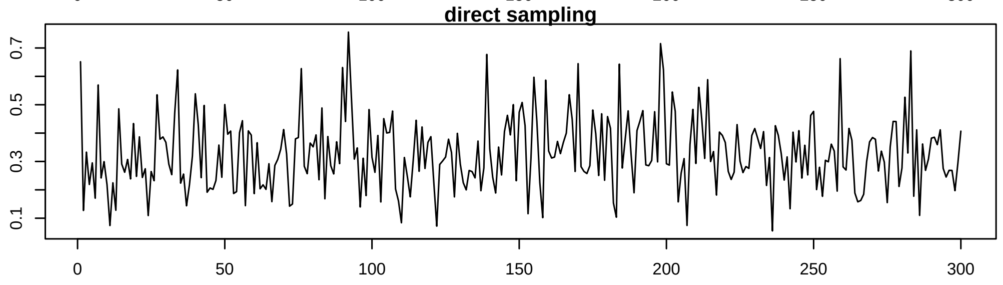
A key point observation in this table is that its results are consistent across all three methods of calculating the posterior summaries, and the results of both MCMC calculations are relatively accurate.

Method	2.5%	25%	50%	75%	97.5%	$E[\theta\|x]$	$SD[\theta\|x]$	accept %
MH, $c = 0.95$	0.1093	0.2364	0.3238	0.4205	0.6097	0.3317	0.1301	42.3%
MH, $c = 0.5$	0.1086	0.2120	0.2972	0.4072	0.5882	0.3148	0.1313	92.7%
direct	0.1093	0.2362	0.3254	0.4210	0.6074	0.3337	0.1303	n.a.

The following plots are of the simulated Markov chain values in the three-hundred iterations after the burn-in phase, $\boldsymbol{\theta}^{(b)}, \ldots, \boldsymbol{\theta}^{(m-1)}$.







The chain corresponding to the acceptance probability 42.3% is shown to densely cover the region of highest posterior density, and is reflective of a simulation that quickly visits and revisits all of the distributions important subregions. This type of pattern yields numerical results of high accuracy. The pattern observed in the chain corresponding to the acceptance probability 92.7% is of poor exploration of the posterior density, wherein the simulation tends to stay in one place and only occasionally jump to new subregions of the posterior distribution. This type of pattern produces lower numerical accuracy. □

The two specialized numerical techniques that we have discussed for Bayesian calculation are the Gibbs strategy for managing multi-parameter distributions, and the Metropolis-Hastings algorithm for managing cases where the posterior density is known only up to proportional equivalence. The following example demonstrates how these two techniques can be combined within the same algorithm, in a very straightforward way. The context of the example is inference under an autoregressive time series model. The example also highlights the ease at which forecasting future values of a time series can be incorporated into a Bayesian analysis implemented by numerical simulation.

Example: Bayesian analysis of $AR(p)$ models

Suppose $x_1, \ldots, x_n$ are measurements an $AR(p)$ time series (x_t) with non-zero mean μ. The autoregressive relationship is

$$x_t - \mu = \phi_1(x_{t-1} - \mu) + \cdots + \phi_p(x_{t-p} - \mu) + w_t,$$

wherein (w_t) is a white noise time series with variance σ_w^2. This may also be written

$$x_t = \alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t,$$

having set $\alpha = (1 - \phi_1 - \cdots - \phi_p)\mu$.

In a Bayesian analysis based on a "non-informative" prior distribution, which is to describe ignorance, a suitable specification on the parameters μ and σ_w^2 is given by the improper prior density such that

$$p(\mu|\sigma_w^2) \propto 1 \quad \text{and} \quad p(\sigma_w^2) \propto 1/\sigma_w^2$$

Similarly, a suitable prior density on the autoregressive parameters $\phi_1, \ldots, \phi_p$ has

$$p(\phi_1, \ldots, \phi_p) \propto 1.$$

In certain situations, it may be desirable to impose an implicit constraint that the time series is causal. For instance, if $p = 1$, the prior distribution on ϕ_1 is uniform across -1 to 1; if $p = 2$, the prior distribution on ϕ_1 and ϕ_2 is uniform on the region defined by the inequalities $\phi_1 + \phi_2 < 1$, $\phi_1 - \phi_2 > -1$, and $-1 < \phi_2 < 1$, which we deduced in a previous example. Oftentimes, this is unnecessary, as a suitable alternative option may be to leave the parameters unconstrained and in post-analysis calculate the posterior probability that the parameters define a causal model.

To work with the data-generating distribution, it is convenient to factor the corresponding density according to

$$p(\boldsymbol{x}|\boldsymbol{\theta}) \quad = \quad p(x_1|\boldsymbol{\theta}) \times p(x_2|\boldsymbol{x}_{-2}, \boldsymbol{\theta}) \times \cdots \times p(x_n|\boldsymbol{x}_{-n}, \boldsymbol{\theta}),$$

writing $\boldsymbol{x}_{-t} = (x_1, \ldots, x_{t-1})$ to denote all data values preceding x_t in the data sequence.

The autoregressive relationship implies $x_t|\boldsymbol{x}_{-t}, \boldsymbol{\theta} \sim N(\alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p}, \sigma_w^2)$, whose density function is

$$p(x_t|\boldsymbol{x}_{-t}, \boldsymbol{\theta}) \quad = \quad (2\pi\sigma_w^2)^{-1/2} \exp\left[-\frac{1}{2\sigma_w^2}\left\{x_t - (\alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p})\right\}^2\right].$$

For the first p values of the time series, $x_1, \ldots, x_p$, observe that the autoregressive relationship writes each as a function of some subset of *unmeasured* data sequence, $x_{-p+1}, \ldots, x_{-1}, x_0$. For example, $x_1 = \alpha + \phi_1 x_0 + \cdots + \phi_p x_{-p+1} + w_t$. These unmeasured values would appear in the first p factors of $p(\boldsymbol{x}|\boldsymbol{\theta})$, as written above. One way to manage them is to include them among the parameters of the model, and estimate them along with the parameters μ, σ_w^2, and $\phi_1, \ldots, \phi_p$. Thus, the full set of parameters in our Bayesian model is

$$\boldsymbol{\theta} = (\sigma_w^2, \mu, x_{-p+1}, \ldots, x_{-1}, x_0, \phi_1, \ldots, \phi_p).$$

The additional parameters are assigned an improper prior density

$$p(x_{-p+1}, \ldots, x_{-1}, x_0) \propto 1$$

as a description of ignorance. Including the unmeasured data values, $x_{-p+1}, \ldots, x_{-1}, x_0$, among the parameters of the model is not the *only* way to manage them, but it would be appropriate in some situations, *e.g.*, when inference on such "starting values" is a specific aim of the analysis project. The upcoming coding examples featured in learning unit 5 manage these parameters differently, by working out the joint distribution of the *measured* data values only, $x_1, \ldots, x_n$ under stationarity.

Having adopted that approach to include the unmeasured data values among the parameters of the model, the data-generating density may be understood through the formula

$$p(\boldsymbol{x}|\boldsymbol{\theta}) \;=\; (2\pi\sigma_w^2)^{-n/2} \exp\left[-\frac{1}{2\sigma_w^2} U(\boldsymbol{x}, \boldsymbol{\theta})\right],$$

where

$$\begin{aligned} U(\boldsymbol{x}, \boldsymbol{\theta}) &= \sum_{t=1}^{n} \{x_t - (\alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p})\}^2 \\ &= \sum_{t=1}^{n} \{(x_t - \mu) - \phi_1(x_{t-1} - \mu) - \cdots - \phi_p(x_{t-p} - \mu)\}^2 . \end{aligned}$$

A Gibbs sampling strategy may be used to simulate from the posterior distribution. Within each iteration, new simulated values of the parameters are obtained by cycling through them, each time holding fixed the parameters that were simulated in a previous portion of the cycle. One potential sequence is to first simulate σ_w^2, then μ, then the unmeasured data values $x_{-p+1}, \ldots, x_{-1}, x_0$, and finally the autoregressive parameters $\phi_1, \ldots, \phi_p$.

The sample mean typically offers a suitable initial value for μ, and the same initial value is often suitable for the unmeasured data values $x_{-p+1}, \ldots, x_{-1}, x_0$ as well. Initial values for $\phi_1, \ldots, \phi_p$ and σ_w^2 may be determined by guesswork, or they could be specified as estimated produced by classical procedures, such as the *Yule-Walker estimates*, which are discussed below.

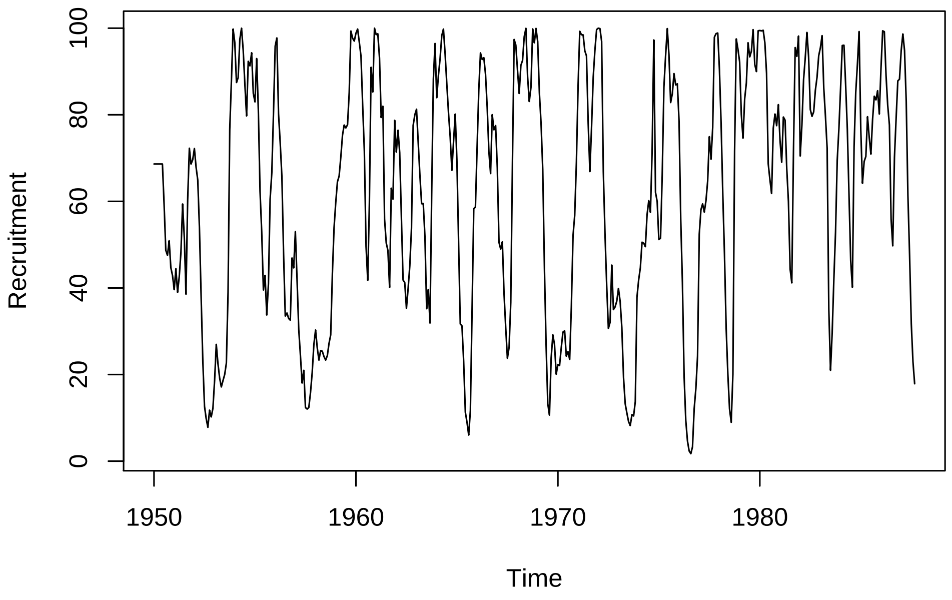
Proposal distributions for all but the white-noise variance parameter, σ_w^2, may be specified by *uniform, symmetric shifts*. For a given parameter value θ (using this symbol generically, not in reference to a moving-average parameter), a symmetric shift γ is obtained as a simulated value from a uniform distribution between $\theta - c$ and $\theta + c$, for some value c. To accommodate the lower bound of zero for the variance parameter, σ_w^2, its proposal distribution may be specified as a *uniform, symmetric shift with a reflecting lower boundary.* This is obtained as the *absolute value* of a simulated value from a uniform distribution between $\sigma_w^2 - c$ and $\sigma_w^2 + c$, for some c. Each of these operations induces to a symmetric proposal distribution, such that $q(\gamma|\theta) = q(\theta|\gamma)$. Standard errors produced by the classical estimates can be helpful to specify effective settings of c across distinct parameters. Setting c to about one such standard deviation often works nicely. After burning in the algorithm, the c values can be adjusted using preliminary standard errors calculated from intermediate MCMC results.

Forecasts may be produced by incorporating into the MCMC algorithm a step within each iteration in which future values of the time series are simulated from the data-generating distribution. For instance, a sequence of predictions, $\boldsymbol{x}^* = (x^*_{n+1}, \ldots, x^*_{n+m})$ would be simulated from the density

$$p(\boldsymbol{x}^*|\boldsymbol{x}, \boldsymbol{\theta}) \quad = \quad p(x^*_{n+1}|\boldsymbol{x}, \boldsymbol{\theta}) \times p(x^*_{n+2}|\boldsymbol{x}^*_{-(n+2)}, \boldsymbol{\theta}) \times \cdots \times p(x^*_{n+m}|\boldsymbol{x}^*_{-(n+m)}, \boldsymbol{\theta}),$$

having written $\boldsymbol{x}^*_{-(n+k)}$ for the predicted or measured values before time $t = n + k$.

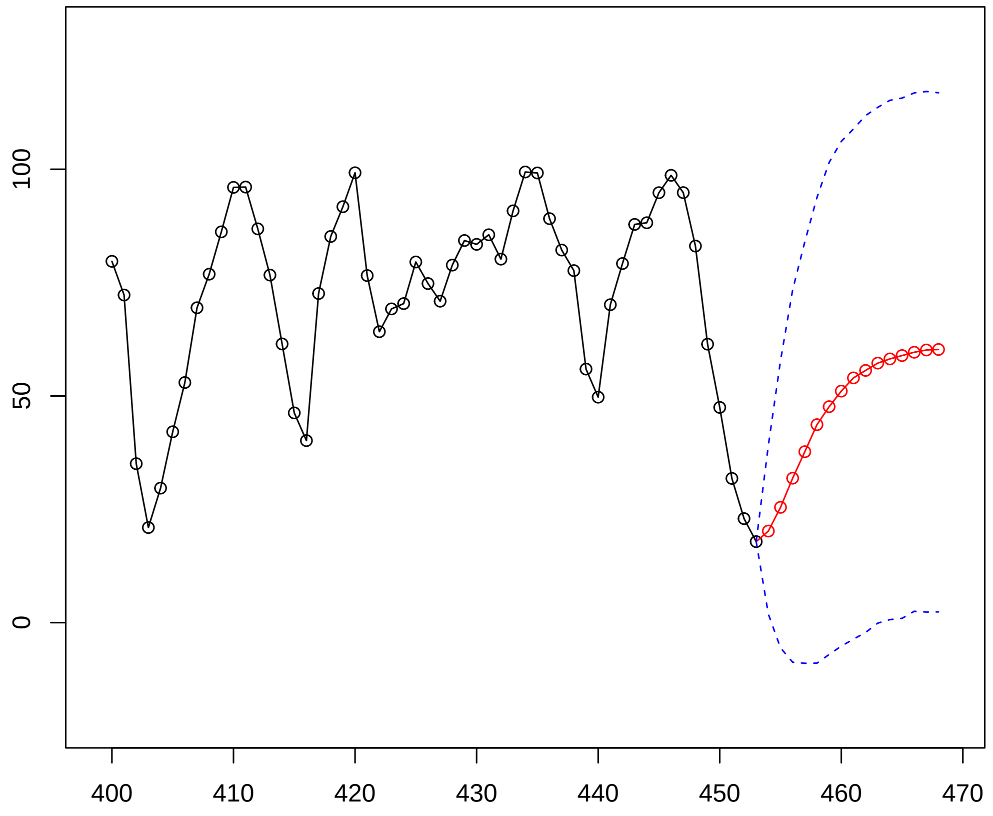
Working with the algorithm outlined above can be computationally intensive when n is large, especially when p is also large. However, it is manageable for analysis under and $AR(2)$ model of the example time series of $n = 453$ monthly fish population counts in the Pacific Ocean from 1950-1988, which we have looked at before. A plot of this time series is as follows.



The table below lists quantiles of each parameter from the posterior distribution.

Parameter	2.5%	25%	50%	75%	97.5%	$E[\theta\|x]$	$SD[\theta\|x]$
σ_w^2	79.760	86.397	90.410	94.767	103.782	90.763	6.101
μ	53.312	58.824	61.623	64.496	70.580	61.717	4.349
x_{-1}	-61.355	23.080	65.512	107.500	186.093	65.086	63.273
x_0	27.967	54.559	67.806	81.318	105.462	67.606	19.502
ϕ_1	1.268	1.319	1.345	1.372	1.423	1.345	0.040
ϕ_2	-0.534	-0.478	-0.452	-0.425	-0.372	-0.452	0.040

Quantiles of forecasted values are displayed in the following plot.



Section 4B.2: Classical estimation.

The discussion thus far in this learning unit has focused on implementing a Bayesian approach to inference in time series analysis. We now discuss two fairly general classical techniques that could be used to estimate a model's parameters: the *method of moments* (MOM) and *maximum likelihood estimation* (MLE). The context assumes an $AR(p)$, $MA(q)$, or $ARMA(p, q)$ framework in which p or q is known, and the objective is to estimate the parameters ϕ_j and θ_j. As indicated in the last example, these techniques can sometimes be useful to specify initial values and tuning parameters of an MCMC algorithm.

To cover the method of moments, suppose (x_t) is a causal $AR(p)$ time series, for which

$$x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t,$$

where (w_t) is Gaussian white noise with variance σ_w^2. Now multiply each side of this formula by x_{t-h} and take expectations; the result is

$$E[x_t x_{t-h}] = \phi_1 E[x_{t-1}x_{t-h}] + \cdots + \phi_p E[x_{t-p}x_{t-h}] + E[w_t x_{t-h}],$$

which can be rewritten

$$\gamma(h) = \phi_1\gamma(h-1) + \cdots + \phi_p\gamma(h-p) + E(w_t x_{t-h}).$$

Taking this further, by causality,

$$x_t = \sum_{j=0}^{\infty} \psi_j w_{t-j},$$

in which $\psi_1 = 1$. It follows that

$$E(w_t x_{t-h}) = \sum_{j=0}^{\infty} \psi_j E(w_t w_{t-j}) = \begin{cases} \psi_0 E(w_t^2) = \sigma_w^2 & \text{if } h = 0 \\ 0 & \text{if } h \geq 1. \end{cases}$$

Such derivation gives rise to a set of mathematical objects called the *Yule-Walker equations*, given by

$$\begin{aligned} \rho(h) &= \phi_1\rho(h-1) + \cdots + \phi_p\rho(h-p) \text{ for } h = 1, \ldots, p \\ \sigma_w^2 &= \gamma(0)\{1 - \phi_1\rho(1) - \cdots - \phi_p\rho(p)\} \end{aligned}$$

Starting with the Yule-Walker equations, substitute $\hat{\gamma}(0)$ for $\gamma(0)$ and $\hat{\rho}(h)$ for $\rho(h)$; the *Yule-Walker estimates*, $\hat{\phi}_1, \ldots, \hat{\phi}_p$ and $\hat{\sigma}_w^2$, are the values of $\phi_1, \ldots, \phi_p$ and σ_w^2 that solve the resulting equations. These are method-of-moments estimates for the autoregressive model.

Additional ideas that are related to the method of moments are covered in the following examples.

Example: Yule-Walker equations in matrix-vector notation

In matrix-vector notation, write

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{bmatrix}, \quad \boldsymbol{\Gamma} = \begin{bmatrix} \gamma(0) & \gamma(-1) & \cdots & \gamma(1-p) \\ \gamma(1) & \gamma(0) & \cdots & \gamma(2-p) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma(p-1) & \gamma(p-2) & \cdots & \gamma(0) \end{bmatrix}, \text{ and } \boldsymbol{\gamma} = \begin{bmatrix} \gamma(1) \\ \gamma(2) \\ \vdots \\ \gamma(p) \end{bmatrix}.$$

The Yule-Walker equations are equivalent to

$$\boldsymbol{\Gamma}\boldsymbol{\phi} = \boldsymbol{\gamma} \quad \text{and} \quad \sigma_w^2 = \gamma(0) - \boldsymbol{\phi}^T\boldsymbol{\gamma}$$

In terms of the ACF, the equations are

$$\boldsymbol{R}\boldsymbol{\phi} = \boldsymbol{\rho} \quad \text{and} \quad \sigma_w^2 = \gamma(0)\{1 - \boldsymbol{\phi}^T\boldsymbol{\rho}\},$$

where

In matrix-vector notation, write

$$\boldsymbol{R} = \begin{bmatrix} \rho(0) & \rho(-1) & \cdots & \rho(1-p) \\ \rho(1) & \rho(0) & \cdots & \rho(2-p) \\ \vdots & \vdots & \ddots & \vdots \\ \rho(p-1) & \rho(p-2) & \cdots & \rho(0) \end{bmatrix}, \text{ and } \boldsymbol{\rho} = \begin{bmatrix} \rho(1) \\ \rho(2) \\ \vdots \\ \rho(p) \end{bmatrix}.$$

Define the matrix $\hat{\boldsymbol{\Gamma}}$ from $\boldsymbol{\Gamma}$ and vector $\boldsymbol{\gamma}$ from $\hat{\boldsymbol{\gamma}}$ by substituting $\hat{\gamma}(h)$ for $\gamma(h)$. Define the vector $\hat{\boldsymbol{\phi}}$ from $\boldsymbol{\phi}$ by substituting $\hat{\phi}_j$ for ϕ_j. The Yule-Walker estimates may be calculated from the inverse matrix of $\hat{\boldsymbol{\Gamma}}$; the estimates are

$$\hat{\boldsymbol{\phi}} = \hat{\boldsymbol{\Gamma}}^{-1}\hat{\boldsymbol{\gamma}} \quad \text{and} \quad \hat{\sigma}_w^2 = \hat{\gamma}(0) - \hat{\boldsymbol{\gamma}}^T\hat{\boldsymbol{\Gamma}}^{-1}\hat{\boldsymbol{\gamma}}.$$

Such manipulations are relevant for expressing the uncertainty of the Yule-Walker estimates. When n is large, the vector of Yule-Walker estimates, $\hat{\boldsymbol{\phi}}$, is approximately normal with mean and covariance matrix given by

$$E[\hat{\boldsymbol{\phi}}] = \boldsymbol{\phi} \quad \text{and} \quad Cov[\hat{\boldsymbol{\phi}}] = \sigma_w^2\hat{\boldsymbol{\Gamma}}^{-1}/n$$

In particular, an approximate $(1-\alpha)100\%$ confidence interval for an individual $\hat{\phi}_j$ is given by the formula

$$\hat{\phi}_j \pm z_{\alpha/2}\sqrt{\hat{\sigma}_w^2[\hat{\boldsymbol{\Gamma}}^{-1}]_{jj}/n}$$

in which $z_{\alpha/2}$ is the $(1-\alpha/2)$ quantile of the standard normal distribution, and $[\hat{\boldsymbol{\Gamma}}^{-1}]_{jj}$ is the j'th diagonal entry of $\hat{\boldsymbol{\Gamma}}^{-1}$. The quantity

$$SE(\hat{\phi}_j) = \sqrt{\hat{\sigma}_w^2[\hat{\boldsymbol{\Gamma}}^{-1}]_{jj}/n}$$

is the *standard error* of $\hat{\phi}_j$. $\square$

Recall the monthly fish population count data from the previous example (and those before), and suppose we adopt an $AR(2)$ model for analysis. Relevant sample autocovariance values are

$$\hat{\gamma}(0) = 780.99, \hat{\gamma}(-1) = \hat{\gamma}(1) = 719.92, \text{ and } \hat{\gamma}(2) = 611.45,$$

which yield the matrix-vector objects

$$\hat{\boldsymbol{\Gamma}} = \begin{bmatrix} \hat{\gamma}(0) & \hat{\gamma}(-1) \\ \hat{\gamma}(-1) & \hat{\gamma}(0) \end{bmatrix} = \begin{bmatrix} 780.99 & 719.92 \\ 719.92 & 780.99 \end{bmatrix}, \text{ and } \boldsymbol{\gamma} = \begin{bmatrix} \hat{\gamma}(1) \\ \hat{\gamma}(2) \end{bmatrix} = \begin{bmatrix} 719.92 \\ 611.45 \end{bmatrix}.$$

The Yule-Walker estimates are $\hat{\boldsymbol{\phi}} = \hat{\boldsymbol{\Gamma}}^{-1}\hat{\boldsymbol{\gamma}}$ and $\hat{\sigma}_w^2 = \hat{\gamma}(0) - \hat{\boldsymbol{\gamma}}^T\hat{\boldsymbol{\Gamma}}^{-1}\hat{\boldsymbol{\gamma}}$. Noting that

$$\hat{\boldsymbol{\Gamma}} = \begin{bmatrix} 780.99 & 719.92 \\ 719.92 & 780.99 \end{bmatrix}^{-1} = \begin{bmatrix} 0.0085 & -0.0079 \\ -0.0079 & 0.0085 \end{bmatrix},$$

the matrix calculations are

$$\begin{bmatrix} \hat{\phi}_1 \\ \hat{\phi}_2 \end{bmatrix} = \begin{bmatrix} 780.99 & 719.92 \\ 719.92 & 780.99 \end{bmatrix}^{-1} \begin{bmatrix} 719.92 \\ 611.45 \end{bmatrix} = \begin{bmatrix} 1.3316 \\ -0.4445 \end{bmatrix}$$

and

$$\hat{\sigma}_w^2 = \hat{\gamma}(0) - \begin{bmatrix} 719.92 & 611.45 \end{bmatrix} \begin{bmatrix} 780.99 & 719.92 \\ 719.92 & 780.99 \end{bmatrix}^{-1} \begin{bmatrix} 719.92 \\ 611.45 \end{bmatrix} = 94.17.$$

The standard errors $SE(\hat{\phi}_j) = \sqrt{\hat{\sigma}_w^2[\hat{\boldsymbol{\Gamma}}^{-1}]_{jj}/n}$ are

$$\begin{aligned} SE(\hat{\phi}_1) &= \sqrt{\hat{\sigma}_w^2[\hat{\boldsymbol{\Gamma}}^{-1}]_{11}/n} = \sqrt{(94.17)(0.0085)/453} = 0.0421 \\ SE(\hat{\phi}_2) &= \sqrt{\hat{\sigma}_w^2[\hat{\boldsymbol{\Gamma}}^{-1}]_{22}/n} = \sqrt{(94.17)(0.0085)/453} = 0.0421. \end{aligned}$$

□

The next example illustrates a shortcoming of the method of moments when it is applied to moving-average models.

Example: Moment-matching in $MA(1)$

Suppose (x_t) is an $MA(1)$ time series such that

$$x_t = w_t + \theta_1 w_{t-1}$$

where (w_t) is Gaussian white noise with $\sigma_w^2 = Var[w_t]$.

Our previous work with this model provides formulas for the first two autocovariance values

$$\gamma(0) = \sigma_w^2(1+\theta_1^2) \quad \text{and} \quad \gamma(1) = \sigma_w^2\theta_1,$$

from which the lag-one autocorrelation is

$$\rho(1) = \frac{\gamma(1)}{\gamma(0)} = \frac{\theta_1}{1+\theta_1^2}.$$

To estimate θ_1 by matching moments, one would substitute the sample autocorrelation $\hat{\rho}(1)$ for $\rho(1)$ and solve for θ_1. This would yield two solutions,

$$\hat{\theta}_1 = \frac{-1 \pm \sqrt{1-4\hat{\rho}(1)^2}}{2\hat{\rho}(1)}.$$

To identify the model, one would select the solution that yields invertibility. Nevertheless, it is possible that no *real-valued* solution exists, for it is possible that $\hat{\rho}(1) \geq 1/2$, even when the data are generated from an $MA(1)$ model with $\rho(1) < 1/2$, in which case the quantity under the square-root, $1 - 4\hat{\rho}(1)^2$, is negative. □

While estimation by moment matching underlies the formulation of Yule-Walker estimates, and typically works well for $AR(p)$ models, it is regarded as a generally unsuitable approach to estimation with $MA(q)$ or $ARMA(p, q)$ models. The last example hints at this deficiency. Maximum likelihood estimation is sometimes regarded as a more capable alternative to moment matching.

The starting point to maximum likelihood estimation is the *likelihood function*, which is a re-expression of the data-generating probability mass or density function written as a function of the parameters,

$$L(\boldsymbol{\theta}; \boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{\theta}).$$

The maximum likelihood estimate $\hat{\boldsymbol{\theta}}$ is the value of $\boldsymbol{\theta}$ that maximizes the likelihood function. Maximum likelihood estimation captures the idea that setting the model parameters to $\hat{\boldsymbol{\theta}}$ makes the data, $\boldsymbol{x}$, most likely to have been observed.

Example: Maximum likelihood estimation in $AR(1)$ with non-zero mean

Suppose (x_t) is a causal $AR(1)$ time series such that

$$x_t - \mu \quad = \quad \phi_1(x_{t-1} - \mu) + w_t,$$

where μ and ϕ_1 are constant parameters and (w_t) is Gaussian white noise with $\sigma_w^2 = Var[w_t]$. In this formulation, $\mu = E[x_t]$, which is to be estimated, along with ϕ_1 and σ_w^2. That is the full set of parameters is $\boldsymbol{\theta} = (\mu, \phi_1, \sigma_w^2)$.

Suppose the data $\boldsymbol{x} = (x_1, \ldots, x_n)$ are observed. To find the likelihood function, we factor the data-generating density as in the previous example:

$$L(\boldsymbol{\theta}; \boldsymbol{x}) = p(x_1|\boldsymbol{\theta}) \times p(x_2|x_1, \boldsymbol{\theta}) \times \cdots \times p(x_n|x_{n-1}, \boldsymbol{\theta}).$$

For $t > 1$, the autoregressive relationship implies $x_t|x_{t-1} \sim N\left(\mu + \phi_1(x_{t-1} - \mu), \sigma_w^2\right)$. However, for $t = 1$, $x_t = x_1$ would be written as a function of the unmeasured data value, x_0. While the unmeasured data could be incorporated as an additional parameter to be estimated, an alternative approach is to work with its marginal distribution. For this, recall the formula

$$Var[x_t] = \gamma(0) = \sigma_w^2/(1 - \phi_1\rho(1)) = \sigma_w^2/(1 - \phi_1^2),$$

which is deduced from the difference equations for this model. Because (x_t) is a Gaussian time series, we thus know that $x_1 \sim N\left(\mu, \sigma_w^2/(1 - \phi_1^2)\right)$.

Putting these pieces together, the likelihood function is

$$\begin{aligned} L(\boldsymbol{\theta};\boldsymbol{x}) &= \sqrt{\frac{1-\phi_1^2}{2\pi\sigma_w^2}}\exp\left\{-\frac{1-\phi_1^2}{2\sigma_w^2}(x_1-\mu)^2\right\} \\ &\qquad \times\prod_{k=1}^{n-1}\frac{1}{\sqrt{2\pi\sigma_w^2}}\exp\left[-\frac{1}{2\sigma_w^2}\{x_{k+1}-\mu-\phi_1(x_k-\mu)\}^2\right] \\ &= (1-\phi_1^2)^{1/2}(2\pi\sigma_w^2)^{-n/2}\exp\left\{-\frac{1}{2\sigma_w^2}S(\mu,\phi_1)\right\}, \end{aligned}$$

where

$$S(\mu,\phi_1) = (1-\phi_1^2)(x_1-\mu)^2+\sum_{k=1}^{n-1}\{(x_{k+1}-\mu)-\phi_1(x_k-\mu)\}^2.$$

Setting to zero the derivative of $\log L(\boldsymbol{\theta};\boldsymbol{x})$ with respect to σ_w^2, it is readily deduced that the MLEs, $\hat{\mu}$, $\hat{\phi}_1$, and $\hat{\sigma}_w^2$, satisfy the relationship,

$$\hat{\sigma}_w^2 = \frac{1}{n}S(\hat{\mu},\hat{\phi}_1).$$

Subsequently, substituting this into $\log L(\boldsymbol{\theta};\boldsymbol{x})$ provides that the MLEs, $\hat{\mu}$ and $\hat{\phi}_1$, may be deduced by maximizing

$$l(\mu,\phi_1;\boldsymbol{x}) = \frac{1}{2}\log(1-\phi_1^2)-\frac{n}{2}\log S(\mu,\phi_1)-\frac{n}{2}\{1+\log(2\pi/n)\}$$

An equivalent criterion is to minimize the function

$$l^*(\mu,\phi_1;\boldsymbol{x}) = \log\left\{\frac{1}{n}S(\mu,\phi_1)\right\}-\frac{1}{n}\log(1-\phi_1^2).$$

There are no direct general solutions to this minimization problem. The MLEs, $\hat{\mu}$ and $\hat{\phi}_1$, must instead be found using numerical optimization algorithms. □

Let us sum up this portion of the learning unit with a handful of additional comments about these classical estimation methods:

In typical analysis situations in which the methods are suitable, both moment matching and maximum likelihood estimation produce estimates whose sampling distributions under hypothetical repeated sampling are approximately normal and centered at or near the value of the corresponding parameters. That is, they are approximately unbiased. Just as confidence-interval formulas associated with the Yule-Walker estimators are available, so too are confidence-interval formulas available that would be reported alongside the maximum likelihood estimates.

The last example describes the setup for maximum likelihood estimation under an $AR(1)$ model. Maximum likelihood methods may be implemented analogously to that example for general $ARMA(p,q)$ models. For estimation in $AR(p)$ models, a modification of the maximum likelihood approach has been proposed that yields solutions that are close to the Yule-Walker estimates. The idea is to condition on the initial value, x_1, which eliminates the marginal density $p(x_1|\boldsymbol{\theta})$ from the likelihood formulation. The modified likelihood function, to be maximized, is

$$L(\boldsymbol{\theta};\boldsymbol{x}) \quad = \quad (2\pi\sigma_w^2)^{-(n-1)/2}\exp\left\{-\frac{1}{2\sigma_w^2}\tilde{S}(\mu,\phi_1)\right\},$$

where

$$\tilde{S}(\mu,\phi_1) \quad = \quad \sum_{k=1}^{n-1}\{(x_{k+1}-\mu)-\phi_1(x_k-\mu)\}^2.$$

This is also connected to least squares methods for estimating autoregressive parameters, which focus on minimizing $\tilde{S}(\mu,\phi_1)$.

As seen in the examples, outside of autoregressive models, moment-matching computations can be nonsensical, even if they are not difficult to carry out. Maximum likelihood computations can be complicated, and rely on numerical optimization techniques. In some situations, obtaining maximum likelihood values is challenging, and requires specialized optimization approaches. Bayesian methods can be computationally challenging as well, but may be preferred in some situations for computational reasons.

Section 4B.3: Bayesian analysis of ARMA models.

The next example demonstrates a Bayesian approach to time-series analysis under an ARMA model, and along the way brings in additional ideas that we have discussed.

Example: Bayesian analysis of ARMA models

Suppose (x_t) is an $ARMA(p,q)$ autoregressive moving-average time series with non-zero mean μ, defined by the relationship

$$x_t-\mu=\phi_1(x_{t-1}-\mu)+\cdots+\phi_p(x_{t-p}-\mu)+w_t+\theta_1 w_{t-1}+\cdots+\theta_q w_{t-q}$$

where (w_t) is a Gaussian white noise time series with variance σ_w^2. This may instead be written

$$x_t=\alpha+\phi_1 x_{t-1}+\cdots+\phi_p x_{t-p}+w_t+\theta_1 w_{t-1}+\cdots+\theta_q w_{t-q},$$

where $\alpha=(1-\phi_1-\cdots-\phi_p)\mu$.

Bayesian methodologists have noted that in the ARMA framework the issue of parameter redundancy complicates the selection of a prior that would describe ignorance. As it turns out, when a uniform prior is specified directly on the autoregressive and moving-average parameters, $\phi_1,\ldots,\phi_p$ and $\theta_1,\ldots,\theta_q$, respectively, the posterior distribution would assign high probability to redundant parameter settings. Such settings can also introduce complications in numerical simulation algorithms that could prevent them from executing as intended. A solution to this issue is to transform the ARMA-model parameters to the first

$p+q$ coefficients in the corresponding $AR(\infty)$ representation, $\psi_1, \ldots, \psi_p$ and $\psi_{p+1}, \ldots, \psi_{p+q}$, and then specify a uniform prior on those parameters; the ψ_j and ψ_{p+k} parameters are later transformed back to the ϕ_j and θ_k when evaluating the data-generating density in the computational algorithm. This approach is effective because, as we know, any redundant parameters cancel in the formulation of the $AR(\infty)$ representation.

Recall from learning unit 3 that we worked out a general strategy for determining the π_j and π_{q+k} from the ϕ_j and θ_k. These values are to solve the recursive equations given by

$$\begin{aligned}\pi_1 &= \phi_1 + \theta_1 \\ \pi_j &= \phi_j + \theta_j - \theta_1 \pi_{j-1} - \cdots - \theta_{j-1}\pi_1\end{aligned}$$

for $j = 2, \ldots, q$, and

$$\pi_{q+k} = \phi_{q+k} - \theta_1 \pi_{q+k-1} - \cdots - \theta_q \pi_k$$

for $k = 1, 2, \ldots$. The same equations may be solved in the opposite direction to provide a general strategy for determining the ϕ_j and θ_k from the π_j and π_{q+k} for $j = 1, \ldots, q$ and $k = 1, \ldots, p$.

We are now in a position to formulate a Bayesian analysis in a parallel manner as we did for $AR(p)$ models in a previous example. As a non-informative prior for μ and σ_w^2, let us specify

$$p(\mu|\sigma_w^2) \propto 1 \quad \text{and} \quad p(\sigma_w^2) \propto 1/\sigma_w^2$$

Similarly, as a prior density on the first $q+p$ parameters of the $AR(\infty)$ representation, $\pi_1, \ldots, \pi_q$ and $\pi_{q+1}, \ldots, \psi_{q+p}$, set

$$p(\pi_1, \ldots, \pi_q, \pi_{q+1}, \ldots, \pi_{q+p}) \propto 1.$$

As before, it is also necessary to consider the *unmeasured* data values, $x_{-p+1}, \ldots, x_{-1}, x_0$, which is to be managed by treating them as additional parameters, on which the improper prior density

$$p(x_{-p+1}, \ldots, x_{-1}, x_0) \propto 1.$$

is specified.

The data data-generating density is factored according to

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = p(x_1|\boldsymbol{\theta}) \times p(x_2|\boldsymbol{x}_{-2}, \boldsymbol{\theta}) \times \cdots \times p(x_n|\boldsymbol{x}_{-n}, \boldsymbol{\theta}).$$

Each factor is determined by the ARMA relationship, for which

$$x_t|\boldsymbol{x}_{-t}, \boldsymbol{\theta} \sim N\left(\alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p}, c_t(\boldsymbol{\theta})\sigma_w^2\right),$$

where

$$c_t(\boldsymbol{\theta}) = \begin{cases} 1 & \text{if } t = 1 \\ 1 + \theta_1^2 + \cdots \theta_{t-1}^2 & \text{if } 1 < t \leq q \\ 1 + \theta_1^2 + \cdots \theta_q^2 & \text{if } t > q \end{cases}$$

The associated conditional density function is

$$p(x_t|\boldsymbol{x}_{-t},\boldsymbol{\theta}) \quad = \quad \{2\pi c_t(\boldsymbol{\theta})\sigma_w^2\}^{-1/2}\exp\left[-\frac{1}{2c_t(\boldsymbol{\theta})\sigma_w^2}\{x_t-(\alpha+\phi_1 x_{t-1}+\cdots+\phi_p x_{t-p})\}^2\right],$$

Multiplying the factors yields the data-generating density formula given by

$$p(\boldsymbol{x}|\boldsymbol{\theta}) \quad = \quad (2\pi\sigma_w^2)^{-n/2}\left\{\prod_{t=1}^{n} c_t(\boldsymbol{\theta})\right\}^{-1/2}\exp\left[-\frac{1}{2\sigma_w^2}U(\boldsymbol{x},\boldsymbol{\theta})\right],$$

where

$$\begin{aligned} U(\boldsymbol{x},\boldsymbol{\theta}) &= \sum_{t=1}^{n}\frac{1}{c_t(\boldsymbol{\theta})}\{x_t-(\alpha+\phi_1 x_{t-1}+\cdots+\phi_p x_{t-p})\}^2 \\ &= \sum_{t=1}^{n}\frac{1}{c_t(\boldsymbol{\theta})}\{(x_t-\mu)+\phi_1(x_{t-1}-\mu)+\cdots+\phi_p(x_{t-p}-\mu)\}^2 \end{aligned}$$

Simulating from the posterior distribution may proceed using a parallel Gibbs sampling strategy as in the $AR(p)$ sample: cycle through the parameters at each iteration, simulating a new set of values while holding fixed the values that were simulated before. One potential sequence for the ARMA case is to first simulate σ_w^2, then μ, then the unmeasured data values $x_{-p+1},\ldots,x_{-1},x_0$, then the autoregressive parameters $\phi_1,\ldots,\phi_p$, and finally the moving-average parameters $\theta_1,\ldots,\theta_p$. $\square$