Similarly if

$$\frac{1}{T-j}\sum_{t=j+1}^{T}(Y_t-\mu)\left(Y_{t-j}-\mu\right)\xrightarrow{p}\gamma_j$$

then $\{Y_t\}$ is *ergodic for second moments*. If $\{y_t\}$ is covariance-stationary and

$$\sum_{j=0}^{\infty}\left|\gamma_j\right|<\infty$$

then $\{Y_t\}$ is ergodic for the mean. Often, covariance-stationarity and ergodicity come hand in hand. For an example of a stationary but non ergodic process see Hamilton (1994, page 47).

V. Maximum likelihood

Now we have sufficient structure to say something about estimation. If we are willing to assume that our data is generated by a stationary ergodic process then we can hope to recover information from the past that will help us to predict the future.

V.1. **The likelihood function.**

Suppose we have a sample of size T on a random variable Y_t and let

$$f_{Y_1,Y_2,\ldots Y_T}(y_1,y_2,\ldots y_T;\theta)$$

denote the joint density of $Y_1,Y_2\ldots,Y_T$. The parameter vector θ captures elements of the model that we would like to estimate. If we treat the joint density as a function of θ for a given sample of data on Y; the result is called the *sample likelihood function*. For example, if we have a sample of independent variables, their joint density is the product of the individual densities and we have

$$f_{Y_1,Y_2,\ldots Y_T}(y_1,y_2,\ldots y_T;\theta)=\prod_{t=1}^{T}f_{Y_t}(y_t;\theta).$$

Taking logs leads to the log likelihood function which we denote

$$(\theta;y_1,y_2,\ldots y_T)=\sum_{t=1}^{T}\log f_{Y_t}(y_t;\theta).$$

If the Y_t are normal then this can be written

$$(\theta;y_1,y_2,\ldots y_T)=\sum_{t=1}^{T}\log\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y_t-\mu)}{2\sigma^2}\right)$$

or

$$(\theta;y_1,y_2,\ldots y_T)=k-\frac{T}{2}\log\left(\sigma^2\right)-\sum_{t=1}^{T}\frac{(y_t-\mu)^2}{2\sigma^2}$$

where the constant $k=-\frac{T}{2}\log(2\pi)$.

V.2. **Numerical optimization.** For the simple example above the maximum likelihood estimator has an algebraic expression.

$$\hat{\mu} = \frac{\sum_{t=1}^{T} y_t}{T},$$

and

$$\hat{\sigma}^2 = \frac{\sum_{t=1}^{T} (y_t - \hat{\mu})^2}{T}.$$

More generally it will be possible to write an expression for the log likelihood function but it may not be possible to find an analytic expression for the solution. In this case (the usual case) we resort to numerical optimization. The state of the art optimizer - written by Chris Sims - is csminwel. This is Matlab code that takes the form;

$$\begin{aligned}[fhat, xhat, ghat, Hhat, itct, fcount, retcodehat] =\\ csminwel(fcn, x0, H0, grad, crit, nit, varargin)\end{aligned}$$

where the arguments are defined as follows:

(1) fcn: string naming the objective function to be minimized
(2) $x0$: initial value of the parameter vector
(3) $H0$: initial value for the inverse Hessian. Must be positive definite.
(4) $grad$: Either a string naming a function that calculates the gradient, or the null matrix. If it's null, the program calculates a numerical gradient.
(5) $crit$: Convergence criterion. Iteration will cease when it proves impossible to improve the function value by more than crit.
(6) nit: Maximum number of iterations.
(7) $varargin$: A list of optional length of additional parameters that get handed off to fcn each time it is called. Note that if the program ends abnormally, it is possible to retrieve the current x, f, and H from the files $g1.mat$ and $H.mat$ that are written at each iteration and at each hessian update, respectively. (When the routine hits certain kinds of difficulty, it write $g2.mat$ and $g3.mat$ as well. If all were written at about the same time, any of them may be a decent starting point. One can also start from the one with best function value.)

V.3. **Exercise for next week.** In the class directory you will find a copy of csminwel. There is also a function called $artificial(mu, sig2, T)$ which generates sequences of random variables with mean mu and variance $sig2$ and stores them in a vector called dat. There are also two rump functions called $Estimateit$ and $makelike$. Your job is to write code for these two m-files.

V.3.1. *Makelike.* This function takes as arguments:

(1) a parameter vector $thet = [mu, sig2]$ and
(2) a data vector y.

The function must return a value for the negative of the log likelihood assuming that y is $N(\mu, \sigma^2)$. (Negative because csminwel is a minimizing function, not a maximizer).

V.3.2. *Estimateit.* This function must

1. Load the data vector
2. Plot the data vector (its always a good idea to eyeball the data).
3. Set the initial values and default options for *csminwel*. These include $H0$, *grad*, *crit*, *nit* and the initial parameter vector. Note - we are calling this *thet* but in csminwel it is $x0$.
4. *Varargin* consists of any arguments passed to the function to be minimized. In our case we need to pass the data vector y.
5. Call *csminwel* and set *thet* (the output of the program) to the variable *xhat* set by *csminwel*.

Try running *artificial* lots of times and see how close your estimates come to the true values. You might also want to see what happens if you use initial values that are a long way from the true ones.

References

Doob, J. L. (1953): *Stochastic Processes*. John Wiley and Sons.
Hamilton, J. D. (1994): *Time Series Analysis*. Princeton University Press, Princeton.
Royden, H. L. (1988): *Real Analysis*. Prentice Hall.