The Information Matrix Equality

Lecture Note
Advanced Microeconometrics
Anders Munk-Nielsen
Fall 2016, version 1.1

In this note, we will prove the *Information Matrix Equality* (unconditional version).[1] The notation used herein is

$$\begin{aligned} s(w,\theta) &\equiv \nabla_\theta \log f(w;\theta), \\ H(w,\theta) &\equiv \nabla_\theta s(w,\theta). \end{aligned}$$

Here, $s(w,\theta)$ is $P \times 1$ and $H(w,\theta)$ is $P \times P$, and w is "the data", which will typically be (y, x). Throughout, θ_o will denote the true population value of θ. We will start by proving that the scores are equal to zero in expectation. Using this and some other results derived along the way, we will show that the Information Matrix (the expected value of the outer product of the scores, $\mathbb{E}[s(w,\theta)s(w,\theta)']$), is equal to the negative expected Hessian. This result is important because it greatly simplifies the task of estimating the asymptotic variance. Moreover, we can use the Information Matrix as an approximation of the (negative) Hessian. This is convenient when we are optimizing the likelihood function numerically since we can get a Hessian approximation using only the gradient (for each observation).[2]

1 A Useful Result

We start by proving a useful result, that the scores in expectation equal zero at the true parameter values. In some sense, this amounts to proving that the true parameters are *identified*, in M-estimator jargon, because it shows that the true parameters are a solution to

[1]The proof follows the exposition by Walter Sosa-Escudero in unpublished notes dated April 13, 2009, available online `http://www.econ.uiuc.edu/~wsosa/econ507/MLE.pdf` (as of 2015).

[2]You will not be required to go through the proof at the exam but you must understand the result and the

the first-order conditions. With relatively few additional assumptions, one gets from this to uniqueness, but that is not the purpose of this note.

Proposition: The score vector is equal to the zero vector in expectation when evaluated at the true parameters, i.e.

$$\mathbb{E}_w\left[s(w,\theta_o)\right] = 0_{P\times 1}.$$

Proof: For any θ,

$$\int f(w;\theta)\,\mathrm{d}w = 1.$$

Take derivatives on both sides, then

$$\nabla \int f(w;\theta)\,\mathrm{d}w = 0.$$

Note that whenever we use the gradient-operator, ∇ ("nabla"), the result is a $P \times 1$ vector. We will omit this explicit reference to dimensionality for brevity further along. If we assume sufficient smoothness that we can interchange differentiation and integration (see Newey and McFadden, 1994),

$$\int \nabla f(w;\theta)\,\mathrm{d}w = 0.$$

The score is defined as

$$\begin{aligned} s(w,\theta) &\equiv \nabla \log f(w;\theta) = \frac{\nabla f(w;\theta)}{f(w;\theta)} \\ \Leftrightarrow \nabla f(w;\theta) &= s(w,\theta) f(w;\theta). \end{aligned} \tag{1}$$

(omitting the minus for a time) Inserting this,

$$\int s(w,\theta) f(w;\theta)\,\mathrm{d}w = 0. \tag{2}$$

Note that by assumption of a correctly specified model,

$$f(w;\theta_o) = p_o(w),$$

where p_o is the true data density. Thus, the integral becomes the true expectation over the population distribution of w, when we evaluate it at θ_o. That is,

$$0 = \int s(w,\theta_o) f(w;\theta_o)\,\mathrm{d}w = \mathbb{E}\left[s(w,\theta_o)\right]. \quad \blacksquare$$

Remark: The final step is very important and at the core of maximum likelihood analysis.

Recall that $f(w;\theta)$ is a family of densities for w, indexed (or parameterized) by θ. However, our assumption about a correctly specified model implies that when we evaluate them for $\theta = \theta_o$, then it becomes equivalent to the true data density, $p_o(w)$. Therefore, the final equality occurs because for any function, $g : \mathbb{R} \to \mathbb{R}$, we *define* the expectation of $g(w)$ as

$$\int g(w) p_o(w) \, \mathrm{d}w \equiv \mathbb{E}[g(w)].$$

2 The Information Matrix Equality

Proposition: *(The Information Matrix Equality)* The negative expected value of the Hessian matrix is equal to the expected outer product of the scores, i.e.

$$-\mathbb{E}\left[H(w, \theta_o)\right] = \mathbb{E}\left[s(w, \theta_o) s(w, \theta_o)'\right].$$

Note that $s(w, \theta_o) s(w, \theta_o)'$ is the $P \times P$ outer product of the gradients.

Proof: Omitting arguments to functions, let us take derivatives on both sides of (2), we obtain

$$\begin{aligned}
\nabla_\theta 0 &= \nabla_\theta \int sf \, \mathrm{d}w \\
&= \int \nabla_\theta (sf) \, \mathrm{d}w \\
&= \int f \nabla s + s \nabla f \, \mathrm{d}w \\
&= \int f \nabla s \, \mathrm{d}w + \int s \nabla f \, \mathrm{d}w \\
&= \int f \nabla s \, \mathrm{d}w + \int ssf \, \mathrm{d}w,
\end{aligned}$$

where we have used equation (1) to say that $\nabla f = sf$. Note that for the term "ssf", we have omitted a transpose for the score — it should be the outer product ss', which is $P \times P$, which we will be explicit about in the following. We observe, again, that when $\theta = \theta_o$, we are integrating wrt. $p_o(w)$, which is the same as taking a conditional expectation. Using the definition that $\nabla s(w, \theta) \equiv H(w, \theta)$,

$$\begin{aligned}
-\int H(w, \theta_o) f(w; \theta_o) \, \mathrm{d}w &= \int s(w, \theta_o) s(w, \theta_o)' f(w; \theta_o) \, \mathrm{d}w \\
\Leftrightarrow -\mathbb{E}\left[H(w, \theta_o)\right] &= \mathbb{E}\left[s(w, \theta_o) s(w, \theta_o)'\right]. \quad \blacksquare
\end{aligned}$$

2.1 Implications for Variance Estimation

This result is incredibly useful for maximum likelihood estimation (MLE). Normally, in order to obtain standard errors for an M-estimator, one needs both the average outer product of the scores, $\hat{B}$, and the average Hessian, $\hat{A}$, to obtain the variance estimator using the "sandwich formula", $\hat{V} = \frac{1}{N}\hat{A}^{-1}\hat{B}\hat{A}^{-1}$. But for MLEs, it holds that $A_o = B_o$, so we can estimate the variance as either $\hat{V} = \frac{1}{N}\hat{A}^{-1}$ or $\hat{V} = \frac{1}{N}\hat{B}^{-1}$.[3]

Suppose we have coded up a model in Matlab, where the function `Q(theta)` returns the *sum* of the negative log likelihood, given by a function called, say, `mle.criterion(y,x,theta)`. Then we may use the following code to obtain standard errors directly from the built-in optimizer, `fminunc`;

```
% sum of negative log likelihood contributions
Q = @(theta) sum(mle.criterion(y,x,theta));
% choose some starting values
theta0 = zeros(P,1);
% find the minimum
[thetahat,nll0,flag,out,grad,hess] = fminunc(Q,theta0);
% estimate of covariance matrix
A = hess/N;
cov = 1/N * A^-1;
% standard errors
se = sqrt(diag(cov));
```

To obtain the outer product of the scores, $\hat{B}$, we need to use the criterion function that returns the likelihood in $N \times 1$ vector format (in M-estimator jargon, we usually call this function $q(y, x, \theta)$). Then we need some function that can return the scores in an $N \times P$ matrix, where $P = \dim \theta$. Such a function should take as inputs `fhandle`, a function "handle", and `x0`, the point at which to do the derivative. Suppose the function is called `estimation.centered_grad(fhandle,x0)`. Then the code might be:

```
% create function handle
q = @(theta) mle.criterion(y,x,theta);
% compute finite-difference in N*P format
s = estimation.centered_grad(q,thetahat);
% covariance matrix
B = s'*s/N; % avg. outer product of the gradients, K*K
cov = 1/N * B^-1;
% standard errors
se = sqrt(diag(cov));
```

[3]Note that the information matrix equality is formulated for the (positive) log likelihood but the A and B matrices are defined for the negative log likelihood, hence the seemingly omitted minus in "$A_o = B_o$".