July, 2012

Towards Better Estimation of Jump Diffusion Models

Richard H. Serlin, *University of Arizona*

Towards Better Estimation of Jump Diffusion Models

Richard Serlin

July 2nd, 2007

I Maximum Likelihood Estimation and The E-M Algorithm

Maximum likelihood estimation (MLE) is a highly regarded estimation technique for several reasons. It is consistent and asymptotically normal (CAN). It is asymptotically unbiased, although not necessarily in small samples. and most impressively it achieves the Cramer Rao lower bound, meaning that it has the lowest asymptotic variance of any consistent estimator.

It should be kept in mind, though, that MLE is a "large sample" technique. There is often little known about its small sample properties in the situation at hand, although often one might perform simulations to get an idea of small sample performance.

Standard errors are taken from the estimators' *asymptotic* variance covariance matrix, so they are just estimates of the true variance covariance matrix, and typically the smaller the sample, the worse of an estimate they will be. Thus, again it is recommended that MLE only be used with "large" samples. In fact, the exact form even of the *asymptotic* variance covariance matrix cannot be calculated because it would require knowing the true value of the parameter vector (let's call it θ). We instead use a consistent estimator of the asymptotic variance covariance matrix

One such estimator is the inverse of the Hessian of the likelihood function, evaluated at the MLE. However, it is usually much easier computationally to use the estimator of Berndt, Hall, Hall, and Houseman (1974). This estimator, also called the outer product of gradients, or OPG, estimator, involves only the evaluation of first derivatives.

Although MLE has many desirable properties, it has drawbacks that can be enormous in some cases. For the most part these involve numerical and computational difficulties. Jump diffusion models, for example, can have 10 or more parameters, and their likelihood

functions can be extremely complicated and poorly behaved. Deriving such a likelihood function may be difficult, and it may not be possible to express it in closed form. In that case the computer will have to approximate the value of the likelihood function many times each iteration, adding to the numerical difficulty.

Usually the biggest problem in complicated, high dimensional cases is *global* optimization. Although current algorithms and software are very competent at finding *local* optimums, often in complicated high dimensional cases, there exist many, or an infinite number of, local optimums, and it can be very difficult to be sure that the optimum you've found is the global one. Global optimization of complicated high dimensional functions is very difficult and often intractable, however, a recent important advance is simulated annealing.

Simulated Annealing

The method is so named because it is analogous to the physical process of annealing. Annealing is a process that occurs when metals cool from high temperatures. If they are allowed to cool slowly, their molecules align in a pure crystal state, a state of *minimum* energy. In other words, *slow* cooling allows the metal to attain a global *minimum*, while *fast* cooling, or quenching, of the metal will result in the molecules arranging themselves in a polycrystalline or amorphous state having higher energy, in other words not achieving the global energy minimum.

Simulated annealing is like slow cooling. the algorithm is slow and patient in searching for the global optimum, as opposed to being quick in its search, and running up to the top of the nearest hill only to converge to a local, but not global, maximum.

The specific simulated annealing algorithm, *for maximization*, is essentially as follows (if we want to minimize an objective function, we can just maximize the negative of that function):

1) Choose a starting point (let's call it θ_0) by using prior information, and perhaps by applying another optimization technique. It should be clear as the algorithm is explained further that the choice of starting point is much less important with simulated annealing than with conventional local optimization techniques.

2) To find θ_{i+1} given θ_i , try to think of an efficient search algorithm, with a random component, which will select a number of good candidate points in a neighborhood of θ_i. These are candidates to be the next θ, θ_{i+1}. We want our search algorithm to be intelligent

and efficient in that it selects points likely to be a move us smartly in the direction of the global optimum. Call those candidates $c_1 ... c_K$.

3) Evaluate the objective function, let's call it $l(\theta)$, at all of the candidate points. Then, assign a probability of being "elected" the next θ, to each of the c's, according to the following formula

$$Prob\left[c_k \text{ is chosen to be } \theta_{i+1}\right] = \exp\left(\left[l\left(c_k\right) - l\left(\theta_i\right)\right]/T\right)$$

and

$$Prob\left[\theta_i \text{ is chosen to be } \theta_{i+1}\right] = 1 - \sum_{k=1}^{K} \exp\left(\left[l\left(c_k\right) - l\left(\theta_i\right)\right]/T\right)$$

If any c has a probability greater than or equal to 1, then it is chosen with certainty. If more than one c has a probability greater than or equal to 1, then the one with the highest probability is chosen with certainty.

This probability function, a Boltzman probability function, is the type used in the physical model of the annealing process. Thus, it is another reason why this global optimization technique is called simulated annealing.

In accordance with the probabilities of the c's and of θ_i, one of them is chosen via a random draw. Because of this randomness, it is possible that $l\left(\theta_{i+1}\right)$ will be lower than $l\left(\theta_i\right)$. Thus, the algorithm has the capability to jump out of inferior local optimum. This is in contrast to local optimization algorithms which always go in the direction of greatest local increase, and thus will tend to stick at a local optimum.

The parameter T was given that letter because it is analogous to temperature in annealing. As can be seen from the probability function, when T is large, or when temperature is high, the probability of moving from the current point is higher. Also, the candidate search algorithm is typically made a function of T in that when temperature is high, the average distance of candidates from the current point, x_i, is greater than when temperature is low. Thus, when the process is hot, it tends to roam widely sampling the surface of the objective function.

Another part of the simulated annealing process is that the temperature T is gradually decreased according to some schedule that the programmer decides on. Again, analogous to

physical annealing, if the temperature is cooled quickly, the process may quickly settle in a local optimum and not leave before a programmer supplied convergence criteria is reached. Thus, if the process is "quenched", an inferior point is likely to be reached.

With "slow cooling" the process will first build up a rough view of the surface by moving often and with large steps. As the temperature falls and step length decreases it slowly focuses on the most promising area.

Thus, the slower the cooling schedule, the more likely it is that the optimum found will be global, but the longer the process will take. Therefore, with a difficult and/or high dimensional objective function, "slow enough" cooling may take an impractical amount of computing time. On the other hand, if we cool too quickly, we may converge to only a local optimum. It may be difficult even with simulated annealing to be sure that the optimum we have converged to is global.

We do, however, note that Veall (1990) contains a test for global optimization applicable in some cases. Also, certain annealing schemes, that is candidate selection algorithms and cooling schedules, have been proven to converge to the global optimum almost surely in some cases. See Judd (1998) for details. But even here the guarantee is that when the algorithm does converge, it will converge to the global optimum, no limit is given as to how much computing time this will take.

Thus, simulated annealing is a powerful advance, but far from a cure all, and its effectiveness will depend on the cleverness of the programmer in deciding on the candidate search algorithm and cooling schedule.

The method was first developed by Metropolis et al. (1953). Goffe, Ferrier, and Rogers (1994) provide a nice overview of the algorithm. They also test it against three common conventional optimization routines with four important econometric problems, and find impressive results. In addition they provide ideas for more efficient specific implementation.

Singularities in the Likelihood Function

Sometimes the global optimum does not exist, and in fact – dramatic statement – this is probably the case with the likelihood function of *any* jump diffusion model. Kiefer (1978) shows that if the likelihood function consists of a product of mixtures of normal densities, then it contains an infinite number of singularities, and the same reasoning behind why this

is true implies that an infinite number of singularities will exist with almost *any* mixture of densities. We will discuss and develop this idea in detail in the pages to come.

Honoré (1998) develops keifer's insight further, showing how singularities exist in the likelihood function of the Merton (1976) jump diffusion model. When there are multiple singularities in the likelihood function, clearly it can be shown that maximum likelihood estimation may not be reliable. We will also be show that the true parameter vector need not be at one of these singularities. And, in fact, it won't be in any of the models we would work with because the singularity points do not make economic sense. Thus, maximum likelihood estimation will not be consistent, and will yield nonsense answers, which, as Honoré points out, has been reported by a number of researchers.

Let's now examine in more detail why these singularities exist. Please consider the following:

Case 1: MLE when $x_t \sim N(\mu, \sigma^2)$

In this case:

$$L(x_1 ... x_T \mid \mu, \sigma) = f_{N(\mu,\sigma^2)}(x_1) \cdot f_{N(\mu,\sigma^2)}(x_2) \cdot \ldots \cdot f_{N(\mu,\sigma^2)}(x_T)$$

$$= \left[\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left[\frac{x_1 - \mu}{\sigma}\right]^2\right)\right]\left[\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left[\frac{x_2 - \mu}{\sigma}\right]^2\right)\right]$$

$$\ldots \left[\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left[\frac{x_T - \mu}{\sigma}\right]^2\right)\right]$$

Now, suppose $\sigma \to 0$ and $\mu \to x_1$. First, lets consider what happens intuitively. When $\sigma \to 0$ and $\mu \to x_1$, the normal density becomes an infinite spike at $\mu = x_1$. So, we end up with:

$$L(x_1 ... x_T \mid \mu = x_1, \sigma = 0) = f_{N(x_1,0)}(x_1) \cdot f_{N(x_1,0)}(x_2) \cdot \ldots \cdot f_{N(x_1,0)}(x_T) = \infty \cdot 0 \cdot \ldots \cdot 0$$

and it can be shown that the relative speeds of convergence are such that this product is 0 (essentially because an exponential function converges far faster than a power function). So, the strategy of trying to maximize the likelihood function, by creating an infinite spike at one of the sample points, will *not* create a singularity, but as we will see below, this kind of strategy *will* create a singularity when we have a mixture of normal densities.

Case 2: MLE with a Mixture of Normal Densities, for Example; $f_{x_t}(x_t) = wf_{N_1(\mu_1,\sigma_1^2)}(x_t) + (1-w)f_{N_2(\mu_2,\sigma_2^2)}(x_t)$

In this case:

$$L(x_1...x_T \mid \mu, \sigma) =$$

$$\Big[wf_{N_1(\mu_1,\sigma_1^2)}(x_1) + (1-w)f_{N_2(\mu_2,\sigma_2^2)}(x_1)\Big] \cdot \Big[wf_{N_1(\mu_1,\sigma_1^2)}(x_2) + (1-w)f_{N_2(\mu_2,\sigma_2^2)}(x_2)\Big]$$

$$\cdot \ldots \cdot \Big[wf_{N_1(\mu_1,\sigma_1^2)}(x_T) + (1-w)f_{N_2(\mu_2,\sigma_2^2)}(x_T)\Big]$$

Now, suppose we try a similar strategy to the one we used above in case 1; for the parameters in the first normal, let $\sigma_1 \to 0$ and $\mu_1 \to x_1$, but for the other normal in this mixture of normal densities, let's *not* make a spike. Let's let $\mu_2 = a$ and $\sigma_2 = b$ where a can be any finite real, and b is any finite real that's *not* 0, so that second normal won't be an infinite spike, only the first normal will. What happens? First, lets look at the first term:

$$\big[wf_{N_1(x_1,0)}(x_1) + (1-w)f_{N_2(a,b^2)}(x_1)\big]$$

The first part is ∞, and the second part is some finite number; let's call it k_1 . So we have $\infty + k_1 = \infty$.

But what happens with the second term:

$$\big[wf_{N_1(x_1,0)}(x_2) + (1-w)f_{N_2(a,b^2)}(x_2)\big]$$

The first part is 0, and the second part is some finite number; let's call it k_2 . So we have $0 + k_2 = k_2$. If we didn't have a mixture we would only have the first part, and it would be 0. Then, we would be multiplying $0 \cdot \infty$, and because of the rates of convergence we would get 0, so there would be no problem with a singularity. The mixture, however, allows us to choose parameters for the second normal that don't yield 0, so now instead of $\infty \cdot 0 = 0$, we have $\infty \cdot (0 + k_2) = \infty$. So with a mixture of normals (as well as a mixture of most kinds of densities), there exist singularities, thus there is no single global optimum, and maximum likelihood estimation is not consistent.

The number of singularities is infinite. A singularity exists at every point ($\mu_1 = x_t,\ \sigma_1 = 0,\ \mu_2 = c_1,\ \sigma_2 = c_2$), $x_t \in \{x_1...x_T\},\ c_1 \in \mathbb{R},\ c_2 \in \mathbb{R}\backslash 0$ as well as every

point $(\mu_1 = c_1,\ \sigma_1 = c_2,\ \mu_2 = x_t,\ \sigma_2 = 0),\quad x_t \in \{x_1 ... x_T\},\ c_1 \in \mathbb{R},\ c_2 \in \mathbb{R} \backslash 0$.

Examples of the Improper Use in the Literature of Standard Maximum Likelihood Estimation when the Likelihood Function is a Mixture

Honoré (1998) points out several papers that improperly use standard maximum likelihood estimation. These include Ball and Torous (1983,1985), Beckers (1981), Frost (1993), and Jorion (1988). Let's consider the example of Jorion (1988).

Jorion estimates the following jump diffusion model:

$$\frac{dP_t}{P_t} = \alpha dt + \sigma dz_t + Y_t dq_t$$

where dq_t is a Poisson process with mean number of jumps λ, and Y_t is the jump size; $\log Y_t \sim N(\theta, \delta^2)$. x_t, the continuously compounded return, can be derived to be:

$$x_t = \log(P_t/P_{t-1}) = \mu + \sigma z + \sum_{i=1}^{n_t} \log Y_i$$

where n_t is the actual number of jumps that occurred during the interval, and $\mu \equiv \alpha - \sigma^2/2$.

The log likelihood function for this process (taken directly from the appendix of the paper) is:

$$l_j = -T\lambda - \frac{T}{2}\log(2\pi) + \sum_{t=1}^{T} \log \left[\sum_{j=0}^{\infty} \frac{\lambda^j}{j!\sqrt{\sigma^2 + \delta^2 j}} \exp\left(-\frac{[x_t - \mu - \theta j]^2}{2\left[\sigma^2 + \delta^2 j\right]} \right) \right] \tag{1}$$

Jorion assumes that more than one jump can occur in a non-infintesimal period, and in fact he allows for the possibility of any number of jumps occurring. However, he must truncate the number of possible jumps in order to numerically optimize the likelihood function. He uses a formula for the upper bound of the truncation error that was derived by Ball and Torous (1985). From this Jorion deduces that an upper bound of 10 provides satisfactory accuracy for all of the parameter values encountered in his paper.

Now, I want to show that, as Honoré states, Jorion uses standard MLE, even though his likelihood function is the product of mixtures of densities, and contains singularities.

If the likelihood function contains singularities, then so does the *log*likelihood function,

and it is easier to analyze the likelihood function, so we back it out by exponentiating the *log*likelihood function:

$$\exp(l_j) = L_j = \exp(-T\lambda) + \exp\left(-\frac{T}{2}\log(2\pi)\right) + \prod_{t=1}^{T}\left[\sum_{j=0}^{\infty}\frac{\lambda^j}{j!\sqrt{\sigma^2+\delta^2 j}}\exp\left(-\frac{[x_t-\mu-\theta j]^2}{2\left[\sigma^2+\delta^2 j\right]}\right)\right]$$

$$\Rightarrow L_j \propto \prod_{t=1}^{T}\left[\sum_{j=0}^{\infty}\frac{\lambda^j}{j!\sqrt{\sigma^2+\delta^2 j}}\exp\left(-\frac{1}{2}\frac{[x_t-(\mu+\theta j)]^2}{\left[\sqrt{\sigma^2+\delta^2 j}\right]^2}\right)\right]$$

Now,we will show that this is the product of mixtures of normals.

$$= \prod_{t=1}^{T}\left[\sum_{j=0}^{\infty}\left(\frac{\lambda^j}{j!\sqrt{\sigma^2+\delta^2 j}}\right)\exp\left(-\frac{1}{2}\left[\frac{x_t-(\mu+\theta j)}{\sqrt{\sigma^2+\delta^2 j}}\right]^2\right)\right]$$

$$= \prod_{t=1}^{T}\left[\sum_{j=0}^{\infty}\left(\frac{\lambda^j}{j!\sqrt{\sigma^2+\delta^2 j}}\right)\left(\sqrt{2\pi}\sqrt{\sigma^2+\delta^2 j}\right)\left(\frac{1}{\sqrt{2\pi}\sqrt{\sigma^2+\delta^2 j}}\right)\exp\left(-\frac{1}{2}\left[\frac{x_t-(\mu+\theta j)}{\sqrt{\sigma^2+\delta^2 j}}\right]^2\right)\right]$$

$$= \prod_{t=1}^{T}\left[\sum_{j=0}^{\infty} w_j \cdot f_{N\left(\mu+\theta j,\sigma^2+\delta^2 j\right)}(x_t)\right]$$

where $w_j = \dfrac{\sqrt{2\pi}\lambda^j}{j!}$

So, as we can see, we have a likelihood function that is proportional to one where each data point has a density that is a mixture of normal densities. Thus, we can create singularities using the same strategy that we showed earlier. Let's actually do this, so we can see, concretely and specifically, how this actually happens in an actual model from the

top teir literature. First, we truncate the summation at 10 as Jorion did.

$$L_j \propto \prod_{t=1}^{T} \left[\sum_{j=0}^{10} w_j \cdot f_{N\left(\mu+\theta j, \sigma^2+\delta^2 j\right)}\left(x_t\right) \right] =$$

$$\left[w_0 \cdot f_{N\left(\mu+\theta\cdot 0, \sigma^2+\delta^2\cdot 0\right)}\left(x_1\right) + w_1 \cdot f_{N\left(\mu+\theta\cdot 1, \sigma^2+\delta^2\cdot 1\right)}\left(x_1\right) + \ldots + w_{10} \cdot f_{N\left(\mu+\theta\cdot 10, \sigma^2+\delta^2\cdot 10\right)}\left(x_1\right) \right] \cdot$$

$$\left[w_0 \cdot f_{N\left(\mu+\theta\cdot 0, \sigma^2+\delta^2\cdot 0\right)}\left(x_2\right) + w_1 \cdot f_{N\left(\mu+\theta\cdot 1, \sigma^2+\delta^2\cdot 1\right)}\left(x_2\right) + \ldots + w_{10} \cdot f_{N\left(\mu+\theta\cdot 10, \sigma^2+\delta^2\cdot 10\right)}\left(x_2\right) \right] \cdot \ldots \cdot$$

$$\left[w_0 \cdot f_{N\left(\mu+\theta\cdot 0, \sigma^2+\delta^2\cdot 0\right)}\left(x_T\right) + w_1 \cdot f_{N\left(\mu+\theta\cdot 1, \sigma^2+\delta^2\cdot 1\right)}\left(x_T\right) + \ldots + w_{10} \cdot f_{N\left(\mu+\theta\cdot 10, \sigma^2+\delta^2\cdot 10\right)}\left(x_T\right) \right]$$

Now, following our strategy, let's let $\mu = x_1$, $\sigma = 0$, and let's let θ and δ be some regular finite numbers a and b , where $b \neq 0$. This will give us:

$$\left[w_0 \cdot f_{N(x_1,0)}\left(x_1\right) + w_1 \cdot f_{N(x_1+a,b^2)}\left(x_1\right) + \ldots + w_{10} \cdot f_{N(x_1+10a,10b^2)}\left(x_1\right) \right] \cdot$$

$$\left[w_0 \cdot f_{N(x_1,0)}\left(x_2\right) + w_1 \cdot f_{N(x_1+a,b^2)}\left(x_2\right) + \ldots + w_{10} \cdot f_{N(x_1+10a,10b^2)}\left(x_2\right) \right] \cdot \ldots \cdot$$

$$\left[w_0 \cdot f_{N(x_1,0)}\left(x_T\right) + w_1 \cdot f_{N(x_1+a,b^2)}\left(x_T\right) + \ldots + w_{10} \cdot f_{N(x_1+10a,10b^2)}\left(x_T\right) \right] \qquad (2)$$

This is a singularity, as it is equivalent to:

$$\left[w_0 \cdot \infty + w_1 k_{11} + \ldots + w_{10} k_{110} \right] \cdot \left[w_0 \cdot 0 + w_1 \cdot k_{21} + \ldots + w_{10} k_{210} \right]$$

$$\cdot \ldots \cdot \left[w_0 \cdot 0 + w_1 \cdot k_{T1} + \ldots + w_{10} k_{T10} \right]$$

where $k_{ij} \neq 0$

$$= \infty \cdot k_2 \cdot \ldots \cdot k_{10} = \infty$$

where $k_i \neq 0$

It can also be shown more directly that Jorions' likelihood contains singularities by setting $\mu = x_1$, $\theta = a$, $\delta = b$, and then taking the limit as $\sigma \to 0$.

Because almost any density (or perhaps *any* density) can become an infinite spike at a data point, with some parameter selections, any likelihood function made up of mixtures of

densities is likely to have (or perhaps surely has) a singularity problem.

Next, we will address ways to deal with this problem.

Solutions to the Singularity Problem

Honoré provides two solution techniques, that is to say he provides two methods which obtain consistent and asymptotically normal (CAN) estimates. And, these two methods are very similar to, or very related to, standard maximum likelihood estimation. One he created himself, and the other, he points out, is described in Chapter 22 of Hamilton's book, "Time Series Analysis" (1994). First, let's go over Honoré's method.

Honoré's Method for Solving the Singularity Problem

I must say that I didn't fully understand this when I first read it. It's sketchy, sometimes poorly worded with respect to clarity, and sometimes even literally false, perhaps because English is not his native language, and although often cited, this is a working paper. I bring this up to say that this paper is a good example of why I usually prefer texts. Of course, the more one learns the tricks, the math, the background, etc., the more one quickly recognizes the steps, explanation, etc. that the author is leaving out, and the less his sketchiness costs you in either; 1. taking a lot longer than it could have to understand the article, or 2. just unneccessarily understanding it in a poorer, more surface, way (or misunderstanding it).

What is really important is not how long it takes to say the words in your head, but how long it takes to understand the subject matter. Very often a shorter written work will take much longer to understand because of all of the explanation and clarity that is left out. Years ago printing was relatively a lot more expensive than today and there was more justification for sketchiness, but today printing is very cheap (and computer storage space is virtually free), while the time of highly skilled readers is very expensive, so it's a mistake to make it shorter at the expense of making it take significantly longer to understand.

And, often people make the mistake of thinking that they are saving the reader time by making it less words, but if the words left out are ones that would have made understanding the material, faster, better, easier, then they are actually costing the reader time, and doing him no favor.

On the other hand, if your audience has tremendous background, and has seen every math trick in the book before at least three times, then it would make sense to leave a

lot of steps and explanation out – to be sketchy. With such an audience it won't slow understanding, and will actually make the learning faster.

But, I would beware of the Duffie example. His text assumes the reader has a tremendous background, and has seen every trick and theorem there is at least five times. So, his explanations leave a tremendous amount out, yet the book is often used for second and third year Ph.D. students, who are mystified about how he got from math line 1 to math line 2, because they don't know the 5 in between steps he left out that utilize the 2 tricks and the 3 useful theorems that they don't know or haven't seen enough to recognize.

Finally, If you really want to make it shorter, when the audience has a relatively limited background, then perhaps sometimes you should make it more of an *understandable* overview, with less depth attempted. Although, it does depend on the specific situation, audience, purpose, and other particulars. Sometimes, it might even be appropriate to use a Duffie approach for an audience with limited background.

So, that's my commentary on writing style. It is a literature review, so I thought it might be alright to comment a bit on writing style as well as content.

Now, on to Honoré's method. The good news is that after being in this program a while, I was able to, I think, fully understand this paper when I went through it again carefully. So, this is my description of Honoré's method for dealing with the singularity problem.

Our situation is as I described earlier. The data points in our sample are generated from a density that is a mixture of densities, like a mixture of normals, so our likelihood function looks like this:

$$L(x_1...x_T \mid \mu, \sigma) = \left[w f_{N_1(\mu_1,\sigma_1^2)}(x_1) + (1-w) f_{N_2(\mu_2,\sigma_2^2)}(x_1) \right] \cdot$$

$$\ldots \cdot \left[w f_{N_1(\mu_1,\sigma_1^2)}(x_T) + (1-w) f_{N_2(\mu_2,\sigma_2^2)}(x_T) \right]$$

As we showed earlier, the likelihood function goes to infinity when:

a) One of the variances is zero, and one isn't: $\sigma_1 = 0,\ \sigma_2 \neq 0$ or $\sigma_1 \neq 0,\ \sigma_2 = 0$.

and

b) One of the means equals a data point.

What Honoré does is he constrains the parameters so that this can't happen. Specifically, he makes the restriction:

$\sigma_1^2 = m\sigma_2^2$, where m is a real non-zero finite constant (and we will discuss how m is chosen later).

Note what this restriction does: Now, if σ_1 equals 0, then σ_2 must equal 0 also, and if σ_1 is not 0, then σ_2 will not be 0 either. So, with this condition it is impossible to have one of the variances equal 0, but not the other one. Therefore, it is impossible to satisfy condition a above, and so it is impossible for the likelihood function to be evaluated at a singularity point.

Thus, you have restricted the domain of the likelihood function to include no points that will cause singularities. On the other hand, a given *single* choice of m tremendously restricts the domain, and could easily restrict it to a subset of points that doesn't include the true θ. Therefore, the m is not restricted to a *single* point. You can either make m an unknown parameter to be estimated, where m can be any non-zero finite real, or perhaps you can incorporate some prior information and make m an unknown parameter to be estimated on some subset of the reals.

A difficulty that I see with Honorés' technique, however, is the requirement that m be constrained. Otherwise, a good global optimization program will try to create a singularity by driving m to infinity, thus allowing it to drive σ_1 to zero, *while keeping* σ_2 *non-zero.* Also, the program could drive m to zero, which would remove any constraints on σ_1 and σ_2. Thus, the programmer must constrain m away from zero and infinity, but if she does not constrain it far enough away then the constraint will bind, and the program will not converge to the (unconstrained) bounded maximum. On the other hand, if the programmer constrains m too heavily, then the parameters may be constrained to a space that does not include the (unconstrained) bounded maximum.

As we can see again, global optimization, constrained or unconstrained, is a very serious problem with likelihood based frequentist techniques that may make it hard to have confidence in estimates researchers generate from these techniques. I found interesting a quote from Veall's 1990 Econometrica article (pg. 1459):

> Ideally for a particular problem it would be possible to show that any local maximum must be a global maximum as well, such as if a likelihood function can be shown to be everywhere concave in the parameters. Goldfeld and Quandt (1972, pg. 21) write, however, that such a result is available "only in the rarest cases". Therefore, perhaps the most popular approach to the problem is to try a number of sets of starting values, perhaps randomly generated, and check that in each case the maximization algorithm converges to the same solution. While the more starting values that are tried without finding a new maximum, the more likely it is that the true global maximum has been located, *it appears to be difficult to quantify the degree of certainty.* Moreover, optimization algorithms often move very slowly or even break down entirely in certain regions of the parameter space. *As a result, there may be a temptation, one suspects, to choose alternative sets of starting values fairly close to either the initial set or the solution, easing computational difficulties but making it less likely that another maximum would be found if it existed.*

The problem of global optimization is compounded further by the fact that our field, unlike the physical sciences, places little value on duplicating results, so as to increase their credibility. In fact, usually a researcher will be severely penalized for duplicating the results of another, because the odds are slim that this will lead to a publication, yet he ticks off a lot of valuable time from his tenure clock. And, from what I've seen, in peer review, things like global optimization are little checked , and usually not made publicly available (this is also documented in McCullough and Vinod 1999). So, global optimization is an important issue, and as we will see later, a key advantage of Bayesian techniques is that they usually don't require global optimization.

Now, let's get back to the specifics of Honoré's method. We substitute his constraint; $\sigma_1^2 = m\sigma_2^2$ into the likelihood function, so that we are optimizing:

$$L(x_1...x_T \mid \theta_1, \sigma_1, m\sigma_1, m)$$

with respect to θ_1, σ_1, and m, rather than in standard MLE, where we would would be attempting to optimize:

$$L(x_1...x_T \mid \theta_1, \sigma_1, \sigma_2)$$

with respect to θ_1, σ_1, and σ_2

What you get with Honoré's method is the highest point on the likelihood function, *except for the singularities.* Intuitively, it seems like this point could be CAN, and that this technique could achieve the Cramer-Rao lower bound. Honoré claims his estimator is CAN, but does not refer to any proof, although he does generally reference Kiefer (1978) (and, again Honoré 1998 is a working paper). Kiefer showed that likelihood functions from mixture densities do, in fact, still contain a bounded local maximum that is a CAN *and efficient* estimate for θ (see also Hartley 1978, pg. 738) and standard errors can be constructed in the usual ways used with regular MLE.

Kiefer (1978)

It is interesting to note that the problem of singularities in the likelihood function, when data is generated from mixture densities was pointed out in the literature as early as 1978 in Kiefer's Econometrica paper, yet this problem still seems to be unknown to many. In my own work (Santa Clara, Serlin, and Yan 2004), I was asked to do standard maximum likelihood estimation on such a model, and experienced some of the same peculiarities that Honoré reported. And, as I noted, Honoré mentions several well known papers in top journals that performed standard maximum likelihood estimation on mixture likelihood functions (note that the likelihood function will always be a mixture in jump diffusion models).

Nonetheless, those papers still might have gotten efficient CAN estimates as they might have restricted the domain of θ in such a way that they found the maximum except for the singularities, that is the bounded maximum, or they might have started a local optimization routine close enough to this point that the routine moved to it rather than one of the singularities.

Kiefer also provides a technique for finding the bounded maximum, which is iterative and involves the method of moment generating functions of Quandt (1972).

Hamilton's Method

Hamiltons method is actually a special case of the EM algorithm developed by Dempster, Laird, and Rubin (1977), and as such there is formal proof that it is CAN, and efficient under regularity conditions. Standard errors can be obtained using the same techniques as with standard MLE, as both methods share the same asymptotic variance covariance matrix. The advantage of EM is that the global optimization necessary can be much easier. When Hamilton describes his method as, "a special case of the EM algorithm", what

this essentially means is this *is* the E-M algorithm. Let me clarify. The E-M algorithm is like GMM in that GMM is a broad general technique. In a sense, there's no single specific GMM algorithm. There's one for every set of moment conditions that one can think of, and a clever researcher who can think of especially good moment conditions can provide much value to the field. An example is Aït-Sahalia (2004). Likewise, the EM algorithm is a broad general technique, and there's room for a lot of creativity and cleverness in coming up with a good specific implementation of it, or what Hamilton calls a "special case" of it. Let's now discuss the E-M algorithm in detail.

The EM Algorithm

This is a key intuition regarding the EM algorithm: Suppose you have a data generating process (DGP) with a parameter vector Ψ, and you have some data that came from that DGP, y. So, $Y \sim f_{Y|\Psi}(y \mid \Psi = \psi)$, or $Y \sim f(y \mid \psi)$. You want to estimate ψ via maximum likelihood estimation, but the likelihood function is very difficult to optimize, or intractable.

So, you say to yourself, if only I had this additional data that I can't observe, call it u , for unobservable, then, it would be easy (or easier) to do MLE, because then the likelihood function would be:

$$\prod_{t=1}^{T} f_{Y_t,U_t}(y_t, u_t \mid \psi) \tag{3}$$

instead of

$$\prod_{t=1}^{T} f_{Y_t,}(y_t, \mid \psi) \tag{4}$$

Now, a few things to note here:

First, the ψ in (3) is *exactly* the same as the ψ in (4) . The model doesn't change, we're just now dreaming about how wonderful it would be to observe, if only we could, additional data that was generated from *that same* model, additional data that we call u, because if only we could observe u, then our likelihood function would be (3) instead of (4) . And, likelihood function (3) is a function that's much easier to optimize than likelihood function (4).

Let's give an example to make this more concrete and clear. Suppose the model is the following Euler discretized jump diffusion:

$$R_t = \left(\mu - \frac{1}{2}\sigma_{t-1}^2 + \lambda\sigma_{t-1}^2\right)\Delta t + \sigma_{t-1}\sqrt{\Delta t}\epsilon_t + Q_t^R Ber_t(P)$$

where:

$\epsilon_t \sim N(0,1)$

$Q_t^R \sim N(\mu_{QR}, V_{QR})$

$Ber_t(P)$ is a standard Bernoulli random variable with event probability P.

And, our data is R_t , $t = 1, ..., T$.

The likelihood function, when this is all the data we have, is:

$$L(R_1...R_T) = \prod_{t=1}^{T}\left\{Pf_t^J(R_t) + (1-P)f_t^N(R_t)\right\} \tag{5}$$

where $f_t^N(R_t)$ is the density of R_t given that a jump does *not* occur, and $f_t^J(R_t)$ is the density of R_t given that a jump *does* occur.

f_t^N is a $N(\mu_{Nt}, \sigma_{Nt})$ density.

and

f_t^J is a $N(\mu_{Jt}, \sigma_{Jt})$ density.

where,

$\mu_{Nt} = \left(\mu - \frac{1}{2}\sigma_{t-1}^2 + \lambda\sigma_{t-1}^2\right)\Delta t$

$\sigma_{Nt} = \sigma_{t-1}\sqrt{\Delta t}$

$\mu_{Jt} = \left[\left(\mu - \frac{1}{2}\sigma_{t-1}^2 + \lambda\sigma_{t-1}^2\right)\Delta t + \mu_{QR}\right]$

$$\sigma_{Jt} = \sqrt{\sigma_{t-1}^2 \Delta t + V_{QR}}$$

As we can see, this likelihood is the product of mixtures of normals, so it contains an infinite number of singularities, and we will not be able to globally optimize it.

But, what if we had more data than just the observed values $R_1, ..., R_T$. What if we also knew whether a jump had actually occurred at time t, that is what if we could directly observe the unobservable value of $Ber_t(P)$ for $t = 1, ..., T$.

The model wouldn't change; the parameter vector would still be *exactly* the same. It would still be $\Psi = \left[\mu, \lambda, \mu_{QR}, V_{QR}, P\right]$. It's just that we would now have more data that occurred from the model, with which to do maximum likelihood estimation on its parameters. And, because of that extra data, the form of our likelihood function would be different. It would be much nicer.

Before we had:

$$L = f\left(R_1, ..., R_T \mid \psi\right) = \prod_{t=1}^{T} f\left(R_t \mid \psi\right) = \prod_{t=1}^{T} \left[P f_t^J(R_t) + (1-P) f_t^N(R_t)\right]$$

Now, we have:

$$L_c = f\left(\left[R_1, Ber_1(P)\right], ..., \left[R_T, Ber_T(P)\right] \mid \psi\right) = \prod_{t=1}^{T} f\left(R_t, Ber_t(P) \mid \psi\right) \tag{6}$$

$$= \prod_{t=1}^{T} f\left(R_t \mid Ber_t(P), \psi\right) f\left(Ber_t(P) \mid \psi\right) \tag{7}$$

$$= \left[\prod_{t=1}^{T_N} \left[f(R_t \mid Ber_t(P) = 0) f(Ber_t(P) = 0)\right] \prod_{t=1}^{T_J} \left[f(R_t \mid Ber_t(P) = 1) f(Ber_t(P) = 1)\right] \mid \psi\right]$$

$$= \left[\prod_{t=1}^{T_N} \left[(1-P)\, f(R_t \mid Ber_t(P) = 0)\right] \prod_{t=1}^{T_J} \left[P f(R_t \mid Ber_t(P) = 1)\right] \mid \psi\right] \tag{8}$$

As we can see, this likelihood function no longer contains mixtures of normals, so it does not have the singularity problem, and is globally optimizable. By the way, this likelihood function is called in the literature the "complete information" likelihood function, and is

thus denoted, L_c.

So, if only we could observe $Ber_t(P)$ (our u_t in this example). But, alas, that's just a dream. We just can't know for sure, by looking at our observable data R_t (our y_t in this example) whether a jump has occurred at time t. So let's get back to reality. We can't observe $Ber_t(P)$, $t = 1, ..., T$, so we can't observe likelihood function (8). And, if we can't observe likelihood function (8), we can't argmax likelihood function (8) for Ψ to find an MLE for Ψ.

But, while we can't find the Ψ that makes likelihood function (8) the biggest, because we can't observe likelihood function (8) – that is for any given Ψ we don't know for sure exactly how big (8) is – we can put informed probabilities on how big (8) is when evaluated at any given Ψ. And, from these probabilities we can find the Ψ that, *on average*, makes likelihood function (8) the biggest.

Intuitively, it seems like the Ψ that makes the mean size of the likelihood function the biggest would be a nice estimate of the true Ψ. After all, if you had a choice between estimator $\widetilde{\Psi}$ which made the mean size of the likelihood function 1,000, and an estimator $\widehat{\Psi}$ which made the mean size of the likelihood function 1,400, all other things equal, which would you take? I tried developing this intuition more formally, and explaining the EM algorithm along these lines, but although this story seems to start out nice and intuitive, it runs into some irritating complications and questions that I wasn't able to resolve before time became too short. So, perhaps the EM algorithm could be developed and explained nicely along these lines, but I wasn't able to do it before the alarm clock went off. So instead, I will explain and prove the EM algorithm in the standard way it is done in the literature.

We start with the following application of Bayes rule:

$$f(y \mid \psi) = \frac{f(y, u \mid \psi)}{f(u \mid y, \psi)}$$

Next, we take the log of both sides of the equation:

$$\Rightarrow \log f(y \mid \psi) = \log f(y, u \mid \psi) - \log f(u \mid y, \psi) \tag{9}$$

So, now we have a new way to express the standard MLE loglikelihood function. We can express it in terms of any additional un directly observable data, u, by utilizing the RHS of

(9). And, obviously, because the RHS and LHS are equivalent, any ψ that maximizes the RHS, will also maximize the LHS.

When we resort to the EM algorithm, it will be because maximizing the standard MLE loglikelihood function, the LHS, is very difficult, or intractable, so we might consider maximizing instead the RHS. The problem is we can't because we don't know what u is in the two RHS terms, but we can know the expected value of those two terms *given* some information. For example, we can know:

$$E\left(\left[\log f\left(y, u \mid \psi\right)\right] \mid \mathbb{C}\right) \tag{10}$$

Now, let's talk about an important subject in the EM algorithm, notation. It's hard to explain this algorithm with notation that is clear and not too bulky and complicated. The notation used in the literature, which I will go over, makes it easy to misunderstand, or not be clear on what's going on. So, I am going to make some innovations that I hope will make things clearer and less ambiguous. In fact, I already made one; the use of u to denote the unobserved data, as opposed to z and x used in the literature. Because "unobservable" starts with u, it should be easier to keep this variable straight, and y, which is used in the literature, is not too hard to remember as it is a variable we are used to seeing to denote observed data.

The expectation in (10) is conditioned on the set of conditions $\mathbb{C}$ (again the hardly original, but sometimes underused, pneumatic device of using the first letter of the word, $\mathbb{C}$ for conditions). In the literature, conditions and sub-conditions are written out explicitly. The conditions can get long and/or easy to confuse or misunderstand. The notation I'm using will make the equations more compact, plus then I will have the luxury of writing out the conditions elsewhere, in a nice clear expansive way. We'll see how it works.

Now, what is the expectation (10) conditioned on? look at the expectand: $\left[\log f\left(y, u \mid \psi\right)\right]$. Already we see layers of conditions: $f\left(y, u\right)$ is given, or conditioned on, ψ, and then the whole expression that it is included in is conditioned on $\mathbb{C}$. So, we have to be careful to notate and explain this in a clear, unambiguous way.

The expectand is $\log f\left(y, u \mid \psi\right)$. What is this expression? We will be trying to do a maximization with respect to ψ. So, ψ is just a choice variable in a maximization. It is not an unknown random variable, or random vector in the multidimensional parameter case; we decide what it is, and we pick it so that it makes our loglikelihood function the

biggest. So, given that this ψ is our choice vector in a maximization, let's denote it: ψ^m, again, to remember what it is, m for maximization. So, our expectand is $\log f\left(y, u \mid \psi^m\right)$. And, it may help to make things clearer to write this as:

$$\log f\left(y, u \mid \Psi^M = \psi^m\right) \tag{11}$$

So, in words, (11) says, "the log of $f\left(y, u\right)$ given that the Ψ^M vector we will be using in f will be the numbers ψ^m, say $[12, 5.1, 3.8, 157]$,or whatever."

So the expectand of (10), $\log f\left(y, u \mid \Psi^M = \psi^m\right)$, is just some function with y's, u's, and ψ's, like for example:

$$\log f\left(y, u \mid \Psi^M = \psi^m\right) = \sum_{t=1}^{T} \log \left[\frac{1}{\sqrt{2\pi} u_t \psi_1^m} \exp\left(-\frac{1}{2} \left[\frac{y_t - \psi_2^m}{u_t \psi_1^m} \right]^2 \right) \right] \tag{12}$$

Now, let's consider this example. It is important to note that the *only* things that are random here are the u's. That's it. Everything else is non-stochastic, a constant. The *observed* data, y, is observed and known with certainty, a constant. The ψ is some vector of constants that *we have chosen* to maximize the loglikelihood, or to evaluate the loglikelihood in the process of helping us to maximize it. The only thing that is random is u_t. The u's are not actually observed, so we don't know for certain what they are.

So because u's are the *only* random variables in (12), the expectation operator will act *only* on them, as expectation does nothing to constants. So, when we make the expectation conditional, we will make it conditional on things having to do with the *random* variables. And, of course, the pressing thing here for taking the expectation of (12) is tell us what is the density of u! We can't find the expectation of random variables, or functions of random variables, without knowing their probability density. So, we will make the expectation of (12) conditional on what the density of u is. That is what we will make $\mathbb{C}$. Now, the data u is generated from our model which has a parameter vector Ψ, and we have some data which might tell us something about u, our observed data y, so our density of u is: $u \sim f\left(u \mid y, \Psi\right)$. But that wont be good enough to do the EM algorithm, we are going to have to specify what the specific parameter vector Ψ is in u's density. So let's do that. Let's call that vector ψ^u. So, we are going to take the expectation of (12) given that the density of its random variables, u, is $f\left(u \mid y, \Psi = \psi^u\right)$ So we will make:

$$\mathbb{C} = \{u \sim f\left(u \mid y, \Psi = \psi^u\right)\}$$

O.k., now, let's get back to equation (9), with ψ now renamed ψ^m to emphasize that this is the value of psi that we are substituting in to try to maximize the loglikelihood:

$$\log f\left(y \mid \psi^m\right) = \log f\left(y, u \mid \psi^m\right) - \log f\left(u \mid y, \psi^m\right)$$

We next take the expectation of this equation given $\mathbb{C} = \{u \sim f\left(u \mid \Psi = \psi^u\right)\}$:

$$(9) \Rightarrow E\left[\log f\left(y \mid \psi^m\right) \mid \mathbb{C}\right] = E\left[\log f\left(y, u \mid \psi^m\right) \mid \mathbb{C}\right] - E\left[\log f\left(u \mid y, \psi^m\right) \mid \mathbb{C}\right]$$

$$\Rightarrow \log f\left(y \mid \psi^m\right) = E\left[\log f\left(y, u \mid \psi^m\right) \mid \mathbb{C}\right] - E\left[\log f\left(u \mid y, \psi^m\right) \mid \mathbb{C}\right] \tag{13}$$

Now, it is very important to be clear on the difference between ψ^m and ψ^u. ψ^u is the vector of numbers we use for Ψ *only* for the purpose of calculating the probabilities of u. It is used *only* for u's density, while ψ^m is the different vector of numbers we use the purpose of evaluating the loglikelihood function in our process of searching to find that loglikelihood function's maximum. Let's look at this using the example equation (12) :

$$\log f\left(y, u \mid \Psi^M = \psi^m\right) = \sum_{t=1}^{T} \log\left[\frac{1}{\sqrt{2\pi} u_t \psi_1^m} \exp\left(-\frac{1}{2}\left[\frac{y_t - \psi_2^m}{u_t \psi_1^m}\right]^2\right)\right]$$

Now, we will take the expectation given $\mathbb{C}$, of both sides of the equation, to get the form of (13) :

$$\log f\left(y, u \mid \Psi^M = \psi^m\right) = E\left(\sum_{t=1}^{T} \log\left[\frac{1}{\sqrt{2\pi} u_t \psi^m} \exp\left(-\frac{1}{2}\left[\frac{y_t - \psi^m}{u_t \psi^m}\right]^2\right)\right] \mid \mathbb{C}\right)$$

And, in this example, let's suppose we are assuming that the density of u has $\psi^u = [8, 80]$. And, let's say that we want to evaluate the loglikelihood function at $[7, 70]$, so $\psi^m = [7, 70]$. Then we will have:

$$\log f\left(y, u \mid \Psi^M = \psi^m = [7, 70]\right) = E\left(\sum_{t=1}^{T} \log\left[\frac{1}{\sqrt{2\pi} u_t \cdot 70} \exp\left(-\frac{1}{2}\left[\frac{y_t - 7}{u_t \cdot 70}\right]^2\right)\right] \mid \mathbb{C}\right)$$

$$= \sum_{t=1}^{T} E\left(\log\left[\frac{1}{\sqrt{2\pi} u_t \cdot 70} \exp\left(-\frac{1}{2}\left[\frac{y_t - 7}{u_t \cdot 70}\right]^2\right)\right] \mid \mathbb{C}\right)$$

$$= \sum_{t=1}^{T} \left\{\int_{u(y)} \log\left[\frac{1}{\sqrt{2\pi} u_t \cdot 70} \exp\left(-\frac{1}{2}\left[\frac{y_t - 7}{u_t \cdot 70}\right]^2\right)\right] f\left(u \mid y, \Psi = \psi^u = [8, 80]\right) du\right\}$$

Let's now go back to equation (13) :

$$\log f\left(y \mid \psi^m\right) = E\left[\log f\left(y, u \mid \psi^m\right) \mid \mathbb{C}\right] - E\left[\log f\left(u \mid y, \psi^m\right) \mid \mathbb{C}\right]$$

This gives us another way to express the loglikelihood function, $\log f\left(y \mid \psi^m\right)$, a way that the EM algorithm makes use of. But first, still more notation:

$$(13) \equiv \log f\left(y \mid \psi^m\right) = E\left[\log f\left(y, u \mid \psi^m\right) \mid \psi^u\right] - E\left[\log f\left(u \mid y, \psi^m\right) \mid \psi^u\right] \tag{14}$$

$$\equiv Q\left(\psi^m, \psi^u\right) - H\left(\psi^m, \psi^u\right)$$

$$\equiv Q\left(\psi^m, \psi_i^u\right) - H\left(\psi^m, \psi_i^u\right) \tag{15}$$

$$\equiv Q\left(\psi, \psi_i\right) - H\left(\psi, \psi_i\right) \quad \text{and} \quad \log f\left(y \mid \psi^m\right) \equiv \log f\left(y \mid \psi\right) \tag{16}$$

Thus:

$$\log f\left(y \mid \psi\right) = Q\left(\psi, \psi_i\right) - H\left(\psi, \psi_i\right) \tag{17}$$

In line (15) we add the subscripts, because the EM algorithm is iterative. Line (17) uses the notation that is most common in the literature. This is the notation we will now use.

And now, the moment you've been waiting for, we're done with the notation and we can actually give the algorithm. Again, we use the EM algorithm when we want to do maximum likelihood estimation, but the loglikelihood function, as expressed in the standard way, the LHS of (17), is difficult or intractable to optimize, so instead we utilize the RHS of (17) .

This is what we will do: First, we will make our initial guess of what the true value of ψ is. We will call this initial guess ψ_0. Next, we will find the ψ that maximizes *just* $Q\left(\psi; \psi_0\right)$. Notice that $Q\left(\psi; \psi_0\right)$ is the expected value of the *full information* loglikelihood function *given* that the unobservable data u has a probability density $u \sim f\left(u \mid Y = y, \Psi = \psi_0\right)$.

We will call the ψ that maximizes $Q\left(\psi; \psi_0\right)$, ψ_1.

Now, let's compare ψ_1 to ψ_0: $Q\left(\psi_1; \psi_0\right) \geq Q\left(\psi_0; \psi_0\right)$ because ψ_1 was chosen as the argmax of $Q\left(\psi; \psi_0\right)$. So, if we can show that $H\left(\psi_1; \psi_0\right) \leq H\left(\psi_0; \psi_0\right)$, then we will have shown

that ψ_1 makes the standard MLE loglikelihood function bigger than ψ_0 does (or at least as big). So, now let's show that it is, in fact, true that $H(\psi_1;\psi_0) \leq H(\psi_0;\psi_0)$. And, sorry, you'll notice some new notation: $E_{\psi_0}[\cdot]$. This is used in the literature, and it does have a nice compactness. $E_{\psi_0}[\cdot] \equiv E[\cdot \mid u \sim f(u \mid y, \psi_0)]$.

$$H(\psi_1;\psi_0) - H(\psi_0;\psi_0) \equiv E_{\psi_0}[\log f(u \mid y, \psi_1)] - E_{\psi_0}[\log f(u \mid y, \psi_0)]$$

$$= E_{\psi_0}\left[\log\left(\frac{f(u \mid \psi_1)}{f(u \mid \psi_0)}\right) \mid y\right] \leq \log E_{\psi_0}\left[\frac{f(u \mid \psi_1)}{f(u \mid \psi_0)} \mid y\right] \tag{18}$$

$$= \log \int_{u(y)} \frac{f(u \mid \psi_1, y)}{f(u \mid \psi_0, y)} f(u \mid \psi_0, y)\, du = 0 \tag{19}$$

Line (18) is due to Jensens inequality. Line (19) first just utilizes the definition of expectation, then after cancellation of the $f(u \mid \psi_0, y)$ terms you have the integral of a density over its entire support, which is 1.

Thus, by choosing ψ_1 so that it maximizes the *expected value* of the full information loglikelihood function *given* that the unobservable data u has a probability density $u \sim f(u \mid y \text{ and } \Psi = \psi_0)$, that is by choosing ψ_1 so that it maximizes $Q(\psi;\psi_0)$, we are guaranteed that $\log L(\psi_1) \geq \log L(\psi_0)$. Thus, by iterating this process to produce $\psi_0, \psi_1, \psi_2, ...$, in the limit we will converge to the standard MLE, and that is the EM algorithm. It is a way to find the MLE, but a way that involves maximizing $Q(\psi;\psi_i)$ (iteratively), rather than maximizing $\log L(\psi)$, and if we choose our u well, $Q(\psi;\psi_i)$ can be far easier to maximize than $\log L(\psi)$.

Now, let's do an example of the EM algorithm, the one we started earlier, the jump diffusion model on page 16. In that model our y is $R = R_1, ..., R_T$. And, we decided to make our u, $Ber(P) = Ber_1(P), ..., Ber_T(P)$, where $Ber_t(P)$ is whether or not a jump occurred at time t, something that we cannot directly observe with certainty, just by looking at our observable data, R. We decided to make $Ber(P)$ our u because we had a hunch it would make the resulting complete information loglikelihood function, and that functions conditional expectation, much easier to optimize than the intractable standard loglikelihood function for this model.

The $Q(\psi;\psi_0)$ for our example model can be derived as follows: starting with the defi-

nition of $Q(\psi;\psi_0)$ from equation (7):

$$Q(\psi;\psi_0) = E[\log f(y,u \mid \psi) \mid \psi_i] = E_{\psi_i}[\log f(y,u \mid \psi)]$$

$$= E_{\psi_i}[\log f(R, Ber(P)) \mid \psi]$$

$$= E_{\psi_i}\left[\log \prod_{t=1}^{T} f(R_t, Ber_t(P) \mid \psi)\right]$$

$$= E_{\psi_i}\left[\log \prod_{t=1}^{T} f(R_t \mid Ber_t(P), \psi) f(Ber_t(P) \mid \psi)\right]$$

$$= E_{\psi_i}\left[\log \left\{ \prod_{t=1}^{1-T_J} f(R_t \mid Ber_t(P) = 0, \psi) f(Ber_t(P) = 0) \right.\right.$$

$$\left.\left. \cdot \prod_{t=1}^{T_J} f(R_t \mid Ber_t(P) = 1, \psi) f(Ber_t(P) = 1) \mid \psi \right\}\right] \tag{20}$$

$$= E_{\psi_i}\left[\log \left\{ \prod_{t=1}^{1-T_J} (1-P) f_t^N(R_t) \prod_{t=1}^{T_J} P f_t^J(R_t) \mid \psi \right\}\right]$$

$$\Rightarrow Q(\psi;\psi_0) = E_{\psi_i}\left[\left\{ \sum_{t=1}^{1-T_J} \log\left[(1-P) f_t^N(R_t)\right] + \sum_{t=1}^{T_J} \log\left[P f_t^J(R_t)\right] \mid \psi \right\}\right] \tag{21}$$

Now, let's consider equation (21): what is random in the expectand? The P is an element of the vector Ψ, and the Ψ in that expression is given $\Psi = \psi$. So, P is given. It's a constant. All of the parameters in $f_t^N(\cdot)$ and $f_t^J(\cdot)$ are in ψ, so they're given and therefore constants, and the data R_t is observed and non-stochastic. So, the only thing

that's random in the expectand is T_J. Therefore:

$$E_{\psi_i}\left[\left\{\sum_{t=1}^{T-T_J}\log\left[(1-P)\,f_t^N\left(R_t\right)\right]+\sum_{t=1}^{T_J}\log\left[Pf_t^J\left(R_t\right)\right]\mid\psi\right\}\right] \tag{22}$$

$$=\left\{\sum_{t=1}^{T}\log\left[(1-P)\,f_t^N\left(R_t\right)\right]+\sum_{t=1}^{0}\log\left[Pf_t^J\left(R_t\right)\right]\mid\psi\right\}\Pr\text{ob}\left(T_J=0\right)$$

$$+\left\{\sum_{t=1}^{T-1}\log\left[(1-P)\,f_t^N\left(R_t\right)\right]+\sum_{t=1}^{1}\log\left[Pf_t^J\left(R_t\right)\right]\mid\psi\right\}\Pr\text{ob}\left(T_J=1\right)$$

$$\vdots$$

$$+\left\{\sum_{t=1}^{0}\log\left[(1-P)\,f_t^N\left(R_t\right)\right]+\sum_{t=1}^{T}\log\left[Pf_t^J\left(R_t\right)\right]\mid\psi\right\}\Pr\text{ob}\left(T_J=T\right) \tag{23}$$

Thus:

$$Q\left(\psi;\psi_0\right)=\sum_{k=1}^{T}\left[\left\{\sum_{t=1}^{T-k}\log\left[(1-P)\,f_t^N\left(R_t\right)\right]+\sum_{t=1}^{k}\log\left[Pf_t^J\left(R_t\right)\right]\mid\psi\right\}\Pr\text{ob}\left(T_J=k\right)\right] \tag{24}$$

Next, we need to derive Prob$(T_J = k)$. This is the probability that there were k 1's, out of T independent Bernoullis, or Bernoulli trials, so it looks like it's a binomial, but it's not because in a binomial, all of the independent Bernoulli trials have *the same* probability of success. But, that is not true in our case. First, remember the expectation in (22) is given $Ber(P) \sim f\left(Ber(P) \mid R, \psi_i\right)$. In our example model, the jumps are assumed to be independent of each other, and it is assumed that the probability of a jump having occurred depends only on the current return. So, this is equivalent to saying: $Ber_t(P) \sim f\left(Ber_t(P) \mid R_t, \psi_i\right)$, $t = 1, ..., T$. Thus, the first Bernoulli is distributed $Ber_1(P) \sim f\left(Ber_1(P) \mid R_1, \psi_i\right)$, the second is distributed $Ber_2(P) \sim f\left(Ber_2(P) \mid R_2, \psi_i\right)$, and so on. So, all of the Bernoullis have a different probability of a success (a jump). Therefore, the probability of k successes is unfortunately not a nice simple binomial(T, k). From probability theory, the probability of k successes is the sum of the probabilities of all the ways that k successes can occur. So, for example, the probability of 4 successes, or jumps, is: the probability observations $t = (3, 892, 1007, 3708)$ are jumps, plus the probability observations $t = (15, 734, 1598, 3724)$ are jumps, and so on, until you include every possible way you can have 4 jumps. And,

the formula for the number of possible ways you can have k jumps is:

$$\frac{T!}{k!\,(T-k)!}$$

where, in our example, $T = 3739$. So, for example, the number of ways approximately half of those days, 1869, can be jump days is:

$$\frac{3739!}{1869!\,(3739-1869)!} = 4.641\,2 \times 10^{1123}$$

Forget it, not if you have all the computers in the world! So, have we hit a dead end? Maybe not. A first thought is to just average all of the jump probabilities for $Ber_1(P)$ through $Ber_T(P)$, and then just use that average in a binomial. Usually, in a complicated expression, we cannot replace random variables with their averages because there is some Jensen's inequality problem, but actually here, upon first examination, it actually looks like there may not be a problem. $Q(\psi;\psi_0)$ is a linear function of $Ber_1(P),..,Ber_T(P)$. Post script: I later found out that something similar was was done in Böhning and Nityasuddhi (2003), so that encourages me that this will still allow monotonic convergence to the bounded maximum.

Another possible solution for what to do about $\text{Prob}(T_J = k)$ is a simulation idea. Instead of trying to actually calculate this probability analytically, let's go back to equation (21) :

$$Q(\psi;\psi_0) = E_{\psi_i}\left[\left\{\sum_{t=1}^{1-T_J}\log\left[(1-P)\,f_t^N(R_t)\right] + \sum_{t=1}^{T_J}\log\left[P f_t^J(R_t)\right] \mid \psi\right\}\right]$$

Rather than optimizing the expected value of the expectand conditional on $Ber_t(P) \sim f(Ber_t(P) \mid R_t, \psi_i)$, we could instead take a draw from the density of the expectand conditional on $Ber_t(P) \sim f(Ber_t(P) \mid R_t, \psi_i)$[1], and optimize that. Again, the only thing random in this expectand is T_J, so we would just have to take a draw from the density of T_J. Now, $T_J = [Ber_1(P) \mid R_t, \psi_i] + ... + [Ber_T(P) \mid R_t, \psi_i]$. So, to take a draw from T_J, we just take a draw from each of the Bernoullis and then add those draws together. To do this we will need to derive $f(Ber_t(P) \mid R_t, \psi_i)$.

[1]By taking a draw from a density, I mean actually generating from some random number computer program, a realization, a number, from, that density. Given the limitations of this project, from this point on I will have to assume that the reader has a basic working knowlege of Bayesian and Monte Carlo methods. For those who are interested, a good introduction can be found in Kennedy (2003, Ch. 13), Lee (2004), and Koop (2003).

By Bayes rule:

$$f\left(Ber_t(P)=1 \mid R_t, \psi_i\right)=\frac{f\left(R_t, Ber_t(P)=1 \mid \psi_i\right)}{f\left(R_t \mid \psi_i\right)}$$

$$=\left[\frac{P f_t^J\left(R_t\right)}{(1-P) f_t^N\left(R_t\right)+P f_t^J\left(R_t\right)} \mid \psi_i\right] \tag{25}$$

And, note that (25) is conditioned on $\Psi=\psi_i$, so if $\psi_i=\left[\mu_i, \lambda_i, \mu_{QR_i}, V_{QR_i}, P_i\right]=$ [20,14,7,2,.01], then the P in (25) is .01. Post script: Later I found out this is a method called the stochastic EM algorithm, or SEM algorithm. Its convergence is proven under reasonable conditions, and we will discuss this technique later on page 30. Note that with this technique, the simulated $Q\left(\psi ; \psi_0\right)$ that we are maximizing, allows for relatively easy analytical derivation of its gradient and even its Hessian to improve the speed and accuracy of the maximization, or "M" step. This is not uncommon in EM type algorithms. Lange (1999) notes, "Because the M step usually involves maximizing a much simpler function than the likelihood of the observed data, we can often solve the M step analytically" (pg. 115)

Before we discuss extensions, or variations, of the EM algorithm, let's first discuss a technique or framework that can be used in combination with a member of the EM algorithm family to sometimes make estimation more tractable – limited information maximum likelihood (LIML).

FIML and LIML

Let's give an example to make this more concrete and clear. Let's consider my own research (Santa Clara, Serlin, and Yan 2004). We have a jump diffusion model whose Euler discretization is:

$$R_t=\left(\mu-\frac{1}{2} \sigma_{t-1}^2+\lambda \sigma_{t-1}^2\right) \Delta t+\sigma_{t-1} \sqrt{\Delta t} \epsilon_t+Q_t^R Ber_t(P) \tag{26}$$

$$\Delta \sigma_t=\left(\theta+\kappa_1 \sigma_{t-1}+\kappa_2 \frac{1}{\sigma_{t-1}}+\kappa_3 \sigma_{t-1} \ln \sigma_{t-1}\right) \Delta t+c \sigma_{t-1}^\gamma \sqrt{\Delta t} \eta_t+Q_t^\sigma Ber_t(P) \tag{27}$$

where:

ϵ_t and η_t are standard normals with correlation ρ.

Q_t^R and Q_t^σ are independent of each other and all other random variables in the model

and:

$$Q_t^R \sim N(\mu_{QR}, V_{QR}) \quad , \quad Q_t^\sigma \sim Exp(\mu_{Q\sigma})$$

$Ber_t(P)$ is a Bernoulli random variable.

Now, with full information maximum likelihood estimation (FIML), the likelihood function that is optimized is

$$\prod_{t=1}^{T} f\left[R_t, \Delta\sigma_t \mid \Psi\right] \tag{28}$$

where Ψ is the 13 x 1 vector of all of the parameters in both of the models equations.

With limited information maximum likelihood estimation (LIML), or two step maximum likelihood estimation, the two model equations are considered separately. First, maximum likelihood estimation is done on the first equation by optimizing:

$$\prod_{t=1}^{T} f\left[R_t \mid \Psi_1\right] \tag{29}$$

where Ψ_1 is the 5 x 1 vector of all of the parameters in the first model equation. The resulting $\widehat{\Psi}_1$ will have as one of its elements a $\widehat{P}$.

That $\widehat{P}$ is then substituted in for P in the second model equation, and we do maximum likelihood estimation on this, that is to say we optimize:

$$\prod_{t=1}^{T} f\left[\Delta\sigma_t \mid \Psi_2 \text{ and } P = \widehat{P}\right]$$

for Ψ_2, where Ψ_2 is the vector of all of the parameters in the second model equation except P. This will give us an estimate for every parameter except ρ. Because ρ depends on the interaction between the two model equations (26) and (27), both must be used to jointly estimate ρ, so we must optimize the more complicated likelihood function (28), but we can use the LIML principle and insert the estimates $\widehat{\Psi}_1$ and $\widehat{\Psi}_2$ into (28) first, then the dimensionality of the optimization drops from 13 to 1.

As we suspect, because LIML underutilizes information, it is not as efficient as FIML.

For example, we used only the data R to estimate P, when the data $\Delta\sigma$ also tells us about P, and in general LIML does not take into account covariation between the equations, which often contains information about the parameters. Because LIML is not as efficient, it does not have the same asymptotic variance covariance matrix. An estimate of the LIML asymptotic variance covariance matrix is given by Murphy and Topel (1985).

It seems as though one could iterate a LIML process to, at least analytically, get better results, although like I said, until I get experience or knowledge otherwise I am leery about numerical problems; all other things equal, the more number crunching, the more round-off error. But, for example, in the model we've been working with, it does seem like we could reinsert the estimates of all of the parameters except P, and then re-estimate P using the joint likelihood function. Then, we could insert this new estimate of P, and re-estimate the other parameters. It does seem like an algorithm along these lines would be more efficient, at least analytically, than LIML, and might in the limit converge to FIML. Post script: I did actually later read something implying that this iterating of LIML does, in fact, converge to the full information maximum likelihood estimate, but feeling rushed, I didn't write it up, and now I can't remember exactly where I saw it. It might have been in ECM algorithm literature, which is related. I'll try to dig it up later.

I discuss LIML at this point because it appears it can be used in combination with EM type algorithms to cut down to size particularly nasty maximum likelihood optimizations. When finding the parameter vector that maximizes each of the individual model equations, EM can be used instead of traditional techniques for maximizing the likelihood function directly.

It is true that the EM algorithm is used when the maximization necessary to do standard maximum likelihood estimation is very difficult or intractable, but even with EM, maximization, the M step in EM, may still be very difficult or intractable. So, we may have to take further measures to chop things down to size, like utilizing the LIML frame work so that we can do EM on each of the equations separately rather than jointly, thus decreasing the dimensionality and complexity of our maximizations.

So, the EM algorithm can make tractable the finding of MLE's, or bounded MLE's, when they are intractable, but not always. The E step could be a problem, the M step could be a problem, or both. To solve these problems many variants or extensions of the EM algorithm have been developed in the literature. We will now discuss the most important ones. The first we have already mentioned, the stochastic EM algorithm. As

we have seen, this is a way of dealing with a difficult or intractable E step.

The Stochastic EM Algorithm

As I've said, when I was struggling with the E-step in the example on page 26, one idea I had was perhaps just using a draw from $Q(\psi, \psi_i)$ instead of its expectation . Shortly afterwards, I found out that this is a published variant of the EM algorithm called the Stochastic EM algorithm, and, in fact, it was developed for finite mixture models like ours (the example on 26 is the first equation from our model; Santa Clara, Serlin, and Yan 2004). Again, we first calculate $Q(\psi, \psi_i)$, then, instead of taking, and maximizing, its expectation, we instead take, and maximize, a draw from it. The algorithm was developed by Celeux, and Diebolt (1985, 1986). The stochastic EM algorithm is also related to a technique called the data augmentation algorithm, by Tanner and Wong (1987). This technique generalizes the stochastic EM algorithm, and is also suitable for Bayesian methods, so we will discuss it after our discussion on Bayesian methods.

The Monte Carle EM (MCEM) Algorithm

Again, when I was struggling with the E-step in the example on page 26, one of my first ideas was perhaps just using a draw from $Q(\psi, \psi_i)$ instead of its expectation, and that turned out to be the stochastic EM algorithm, but really the idea was not developed, or generalized, far enough. The MCEM algorithm does more. The basic idea is this: at the E-step of the EM algorithm you want to find the expectation of $Q(\psi, \psi_i)$. But, as we saw in our example, sometimes this task is intractable, so one solution is to use the stochastic EM algorithm, which says instead of maximizing the expectation of $Q(\psi, \psi_i)$, maximize a single draw from the density of $Q(\psi, \psi_i)$.

One way to look at this is, a single draw from the density of a random variable *is* an unbiased and consistent estimate of the expected value of that density. So, perhaps a single draw can be used as a proxy for that expected value, and it has been proven that doing this does, in fact, still allows the algorithm to converge to the MLE. But converging to the MLE, and converging to the MLE fast, are two different matters, and it may be possible to use a better proxy for the expected value of $Q(\psi, \psi_i)$ in order to speed convergence.

A single draw from a density is the sample mean – if the sample has a size of 1 – and the sample mean is unbiased and consistent, but clearly the sample mean from a larger sample will yield a better estimate of the mean, one whose variance is σ^2/n rather than σ^2. And, that is the idea behind the MCEM algorithm. In the stochastic EM algorithm

we use a single draw form $Q(\psi, \psi_i)$ as the estimate of the expected value of $Q(\psi, \psi_i)$. In the MCEM algorithm we use the sample average of a sample of n draws as our estimate of the expected value of $Q(\psi, \psi_i)$. Thus, each iteration of the MCEM algorithm contains less noise than each iteration of the stochastic EM algorithm, and therefore the MCEM on average converges in less iterations. However, each iteration takes longer, because n draws have to be taken instead of 1. Nonetheless, it sems as though this algorithm would usually take less computer time to converge, but I haven't seen anything on that in my reading.

MCEM was introduced by Wei and Tanner (1990). It has been used to tackle some challenging estimations. For example, Guo and Thompson (1992) used MCEM to tackle a complex genetic model, and Chan and Ledolter (1995) successfully use MCEM on a time series model with count data where the E step cannot even be done by standard numerical integration methods because of extremely high dimensionality.

The MCEM does not have the monotonicity property of the EM, that is with the EM, as we have proven, each iteration produces a parameter vector, ψ_i, which makes the likelihood function larger than its predecessor, ψ_{i-1}. With MCEM because of the randomness introduced, by bad luck, $L(\psi_i)$ may be less than $L\left(\psi_{i-1}\right)$. However, Chan and Ledolter (1995) show that under reasonable conditions this is very unlikely.

Wei and Tanner (1990) recommend starting with small values of n in the initial stages of the algorithm and then increasing it as the algorithm moves closer to convergence. For deciding on convergence, it appears that they just recommend plotting, visual inspection, and judgement.

The stochastic EM algorithm is a special case of the Monte carlo EM algorithm, the case where $n = 1$. Diebolt and Ip in chapter 15 of the book "Markov Chain Monte Carlo in Practice" (1996) (the map of France book), make some interesting points regarding the SEM. On page 262 they state, "Alternatively imputing pseudo-complete data (the S-step) and performing maximization (the M-step) generates a Markov chain $\{\theta^{(m)}\}$ which converges to a stationary distribution $\pi(\cdot)$ under mild conditions (Ip 1994). The stationary distribution is approximately centered at the MLE of θ." They later explain that the Markov chain output of the SEM algorithm, $\{\theta^{(m)}\}$, after an appropriate burn-in settles in a "plausable region" from which some inferences can be made about the likelihood function in that area. A point estimate of θ can be obtained by choosing the θ in the plausable region that makes the standard loglikelihood function the largest.

On page 260 they state, "Stochastic EM generally converges reasonably quickly to its stationary regime. Such behavior can be partially explained by the duality principle (Diebolt and Robert 1994). The deterministic M-step and the stochastic S-step generate a Markov chain which converges to its stationary distribution faster than the usual long chains of conditional stochastic draws used in many gibbs sampling schemes". So, here we see an example of a benefit a frequentist method can have over a Bayesian one. In some cases the frequentist method may be faster, easier, or more tractable. I note this because when I look at the benefits of Bayesian techniques, the entire posterior is generated etc., I ask myself when would I *not* want to use them, and one answer is that sometimes at least some bayesian methods are more difficult than frequentist alternatives, or intractable.

To estimate standard errors Diebolt and Ip recommend a conditional parametric bootstrap method. We will discuss these interesting methods in a separate section on page 37.

The Data Augmentation Algorithm

Let's start with a special case and predecessor of the data augmentation algorithm, the poor mans data augmentation algorithm 1 (PMDA1). With this algorithm, we first estimate ψ via the MCEM algorithm. Let's call that estimate $\widehat{\psi}$. Next, we use our $\widehat{\psi}$ to take n draws of the unknown data vector u from $f\left(u \mid y, \widehat{\psi}\right)$; let's call those draws $u^{(1)}, ..., u^{(n)}$. Then, we get a quick discount "poor mans" entire posterior via the formula:

$$f\left(\Psi \mid y\right) = \frac{1}{n} \sum_{j=1}^{n} f\left(\Psi \mid y, u^{(n)}\right)$$

The data augmentation algorithm is an iterative generalization of the poor mans data augmentation algorithm. And, some taxonomy, the data augmentation algorithm is a special case of the gibbs sampler, which is a special case of markov chain monte carlo methods.

The data augmentation algorithm may be desirable when the researcher wants the advantages of having an entire posterior, either in a workable analytic form, or an empirical one, but alternative methods like the Gibbs sampler are very difficult or intractable. And interestingly, Gelfand and Smith (1990) found that the data augmentation outperformed the Gibbs sampler with mixture models. Diebolt and Robert (1994) had similar findings concluding, "Here data augmentation is clearly preferable to gibbs sampling. This shows that the additional variability of this method is a drawback for small sample sizes. When

the sample size increases, the two estimators are essentially the same." (pg. 373).

The Expectation Conditional Maximization (ECM) Algorithm

This is a very interesting and potentially useful extension of the EM algorithm. The standard EM algorithm is looked to when the maximization in maximum likelihood estimation is very difficult or intractable. EM by introducing latent data allows for an easier maximization, but in some cases it still may not be cut down to size. The maximization still may be very difficult or intractable. In such a case, we can make the maximization much easier still, by essentially breaking the one big joint maximization for all of the elements of the parameter vector ψ, into a number of lower dimensional conditional maximizations. For example, we may conditionally maximize $Q\left(\psi, \psi_{i-1}\right)$ for ψ_1 given $\psi_{2,i-1}, \psi_{3,i-1}, ...\psi_{k,i-1}$. Then, we would conditionally maximize $Q\left(\psi, \psi_{i-1}\right)$ for ψ_2 given $\psi_{1,i}, \psi_{3,i-1}, ...\psi_{k,i-1}$, and so on. Conditional maximization could also be with respect to functions of parameters.

Although this algorithm will take more iterations to converge than standard EM, each iteration may take less computing time, so the ECM does not necessarily take more computing time than the EM. This is discussed in Meng (1994). The ECM algorithm was developed by Meng and Rubin (1993). An extension of the ECM algorithm, the expected conditional maximization either (ECME) algorithm (Liu and Rubin 1994), can have, in some cases, substantially faster convergence than the ECM or the EM algorithms, measured in either number of iterations or computing time.

Initial Value Choices and Multiple Runs in EM Type Algorithms

The starting point can be very important to the success of EM type algorithms. Discussion and strategies can be found in Lange (1995), Meng and Van Dyk (1997), and Nityasuddhi and Böhning (2003). The use of multiple runs can increase the speed and quality of convergence. See Lange (1995), Meng and Van Dyk (1997).

Accelerated EM methods

There have been many methods proposed to accelerate convergence of the EM algorithm, as the main criticism is that the algorithm is stable but often slow or unworkably slow. A number of them are gradient based. A nice review of these potentially very useful techniques is in chapter 4 of McLachlan and Krishnan (1997).

Standard Errors

Because the EM algorithm converges to the MLE, or the bounded MLE, the same estimated standard errors can be used, for example the negative inverse of an estimate of the Hessian, or the BHHH estimator. However, in maximum likelihood estimation, often a Newton or quasi-Newton method is used; such a method requires estimating the inverse Hessian anyway, so it is no extra work to just use the inverse Hessian from the last iteration to estimate the standard errors. We do not, however, have that luxury with the standard EM algorithm. Plus, the EM algorithm is usually chosen because the observed logliklihood is complicated and/or difficult, so estimating its Hessian may be no walk in the park. Thus, one of the criticisms of the EM algorithm is that it is more difficult to estimate standard errors than it is with MLE.

Methods to ease the computation have been proposed in the literature. Meng and Rubin (1991) devised a variant of the EM algorithm called the supplemented EM (SEM) algorithm that produces the standard error matrix along with the parameter vector estimate. Meiljson (1989) uses a different approach, but also has an EM type algorithm that speeds the calculation of standard errors. Louis (1982) derives a technique, in some ways analogous to that of BHHH, which utilizes the EM algorithm's latent data framework. This technique can ease the calculation of standard errors greatly, and is widely cited.

As noted earlier, bootstrap methods have been recommended for estimating standard errors with EM type algorithms. This will be discussed in detail in section III on page 37.

It is important to emphasize that the EM algorithm converges to the MLE, thus it has the same variance covariance matrix. With maximum likelihood estimation, what is used to estimate the variance-covariance matrix is an approximation of the *asymptotic* varaince-covariance matrix. This asymptotic variance-covariance matrix depends on the central limit theorem, so if the loglikelihood function is far from normal, the asymptotic variance-covariance matrix may not approximate the true finite sample varaince-covariance matrix very well. This caveat is no more or less valid whether we are using an EM type algorithm or standard MLE. What we can do, in some cases, to improve our standard error estimates is transform some of our parameters to improve the normality of our loglikelihood function. Meng and Rubin (1991) discuss this and give examples.

Effectiveness and Appropriateness of the EM Algorithm for Estimation of Mixture Models such as Jump Diffusions

Concerns were raised by some of the faculty about whether the EM algorithm was appropriate or adequate for the estimation of jump diffusions, or whether this was a "good technique". As you can see, I have done extensive reading in this area and have found no clearly disqualifying problems. In fact, I have seen the EM algorithm and its extensions referred to repeatedly as the method of choice when dealing with likelihood functions that are the product of mixture densities. In McLachlan and Krishnan's text, "The EM Algorithm and Extensions" (1997), they state, "It is generally acknowledged that the EM algorithm and its extensions are the most suitable for handling the kind of incomplete data problems discussed in this book." (page 228), and earlier they indicate that they consider mixture models an incomplete data problem:

> The situations where the EM algorithm can be applied include not only evidently incomplete-data situations, where there are missing data, truncated distributions, censored or grouped observations, but also a variety of situations where the incompleteness of the data is not all that natural or evident. These include statistical models such as random effects, mixtures, convolutions, log linear models, and latent class and latent variable structures. Hitherto intractable ML estimation problems for these situations have been solved or complicated ML estimation procedures have been simplified using the EM algorithm. The EM algorithm has thus found applications in almost all statistical contexts and in almost all fields where statistical techniques have been applied – medical imaging, dairy science, correcting census undercount, and AIDS epidemiology, to mention a few.

(page 1). And, mixture models are covered extensively throughout the book. Interestingly, they mention convolutions as well as mixtures. Our models *joint* likelihood actually contains both. It is the product of terms that are mixtures of a normal and a convolution of a normal and an exponential.

Computational Statistics and Data Analysis had a special issue in 2003 devoted entirely to mixture models. In the opening article, Böhning and Siedel write, "The EM algorithm is one of the most frequently used algorithms for determining maximum likelihood estimates in mixture models because of its numerical stability and monotonicity property." (pg. 352).

Delyon, Lavielle, and Moulines (1999) state, "Problems where the EM algorithm has proven to be useful include, among many others, maximum likelihood estimation of the

parameters of mixture densities [see, e.g., Titterington, Smith and Makov (1985)]..."

In Jamshidian and Jennrich's 1997 Journal of the Royal Statistical Society review of accelerated EM type algorithms, they tested the major variants of the EM algorithm aimed at speeding convergence. The tests were in three areas, two of which involved mixture densities: multivariate normal mixtures and Poisson mixtures.

Berkely economist Paul Ruud, in his 1991 Journal of Econometrics survey of EM type algorithms writes, "Statistical models with mixtures are a natural setting for the EM algorithm...".

Diebolt and Ip (1996) state, "The widespread use of EM since Dempster et al. (1977) is phenomenal."

And, I could go on. In general, the top statistics journals are full of theoretical research on the EM algorithm and extensions of it, and applications of these techniques to a wide range of important problems. Some of the worlds top statisticians have worked extensively with this family of algorithms, including Donald Rubin, the chairman of the department of statistics at Harvard, Andrew Gelman of the Berkely statistics department, and Xiao Li Meng of the Chicago statistics department.

There are, however, potentially, two significant problems with EM type algorithms pointed out in the literature. Convergence can be slow, or extremely slow, and there can be more work involved in obtaining standard errors than with standard maximum likelihood estimation. However, as we will discuss, variations of the EM algorithm have been developed to deal with these problems. In part, due to these variations, as I have noted, EM type algorithms are still widely used in statistics and the physical sciences, and are one of the most popular, or the most popular, types of methods for estimating mixture models.

So, it appears there is ample evidence to show that EM type algorithms are appropriate, or even preferred, techniques for estimating mixture models such as jump diffusions, but I do still wonder why neither I, nor any of the faculty I have spoken with, have seen them used to estimate jump diffusions in the finance literature to date, although Honoré (1998) suggests their use, so I will have to investigate this further. Any feedback is most welcome.

II Mixture Models

It is important to note that the estimation of mixture models – and jump diffusions clearly are mixture models – is a huge topic in the statistics literature of the last 30 years. This is largely because in many fields mixture models are a natural choice, as they arise when there are two or more distinct groups, with distinct associated distributions, where group membership is unobserved. With jump diffusions, it is jump observations and non-jump observations. In medicine there may be two or more types of subjects, and it cannot be, or is not, directly observed which type a given subject is. As Böhning and Siedel (2003) put it, "The importance of mixture distributions, their enormous developments and frequent applications over recent years is due to the fact that mixture models offer natural models for unobserved population heterogeneity." (pg. 350). As I have noted, the most common way of estimating these models is with a member of the EM family, which for some reason I haven't seen used with jump diffusions. I have also not seen some of the important results and techniques from the statistics literature mentioned in the finance literature that involves jump diffusion estimation. Along this line, when Philippe Jorion came in recently, I talked about my work and mentioned Honoré (1998) and Kiefer (1978). He was not familiar with either and did not know about the singularity problem, even today in 2004, and even though he, as pointed out by Honoré, improperly estimated a jump diffusion by standard MLE in 1988, completely unaware of the singularity problem, and this paper was published in RFS, so the referee and editors may not have caught it either (I might not mention a story like this in a paper going out to the public. I would be careful about offending or embarrassing someone). so, my question to the faculty is, are contributions in statistics and mathematics sometimes slow, or very slow, in making their way into finance? Another example is Alex Paseka used to say that a lot of the top finance publications with "new" or "original" techniques, actually use techniques that have been implemented in math and physics for decades.

An interesting miscellaneous note on mixture densities: they can be multimodal or unimodal. Hamilton (1994 pg. 687) provides examples of each. It seems, however, considering Hamilton's examples, that usually in financial cases they will be multimodal.

III Bootstrap Methods

The name bootstrap was given because these methods can seem, at least at first glance, like an outlandish trick, that wouldn't actually work, a trick analogous to the one told of by the fictional character Baron Munchausen. A spinner of tall tales, he boasted of how once

upon falling to the bottom of a lake, he got out grabbing his bootstraps and pulling himself up until he was out (according to Alex Paseka, the Russian version has him pulling himself up by his mustache, so in Russia do they call them mustache methods?). The analogy to these methods is both that they seem fanciful and that they are resourceful. They make use of the sample to draw more samples. But before we think about that too much, let's get to some specifics, so we can see exactly what is meant and why these techniques do actually make sense and have been formally proven in a rich literature extending back at least to Efron's seminal article of 1979, and in fact, a similar procedure, the jacknife goes back to Quenouille (1949).

Let's start with the basic bootstrap of Efron (1979). Suppose we have a model with parameter $\mathbf{\Psi}$, and this model generated an i.i.d. random sample of observed data $\mathbf{y}$. Suppose also that we have an estimator of Ψ_i, $\Psi_i(\mathbf{y})$. If it is difficult to analytically calculate the finite sampling distribution of $\Psi_i(\mathbf{y})$, or its asymptotic distribution, then we might try approximating it via this bootstrap method. Also, we may be concerned that we cannot rely on asymptotic arguments, that given our sample size, the finite sample distribution of $\Psi_i(\mathbf{y})$ (that is to say the true and actual distribution of $\Psi_i(\mathbf{y})$) might be very different from its asymptotic distribution. Again, we may use bootstrap methods. There are many cases where bootstrap methods have been shown to provide better small sample approximations than asymptotic methods.

The basic bootstrap goes like this: First we draw B bootstrap samples. What is a bootstrap sample? We start with our regular observed sample, $\mathbf{y} = [y_1, ...y_T]$, and we sample from this set, from $[y_1, ...y_T]$, *with replacement*, T times. So, we would have the computer draw T times from a discrete uniform[1,T]. The numbers drawn would correspond to the y's in that bootstrap sample, so if we drew: 274,15,87,..,31 then our bootstrap sample would be $y_{274}, y_{15}, y_{87}, ...y_{31}$.

Then, for each of the B bootstrap samples, let's call them $\mathbf{y}^1 ... \mathbf{y}^B$, we would calculate the estimator $\Psi_i(\mathbf{y})$. Suppose B = 50,000, then we would have $\Psi_i(\mathbf{y}^1), \Psi_i(\mathbf{y}^2), ..., \Psi_i(\mathbf{y}^{50,000})$. So, we would have an empirical density for the estimator $\Psi_i(\mathbf{y})$. From this we could find out anything we wanted to know about $\Psi_i(\mathbf{y})$. For example, we could estimate the standard error of $\Psi_i(\mathbf{y})$ as:

$$\widehat{s.e.}\left[\Psi\left(\mathbf{y}\right)\right] = \sqrt{\frac{1}{50,000}\sum_{b=1}^{50,000}\left[\Psi_i(\mathbf{y}^b) - \overline{\Psi_i(\mathbf{y})}\right]^2} \tag{30}$$

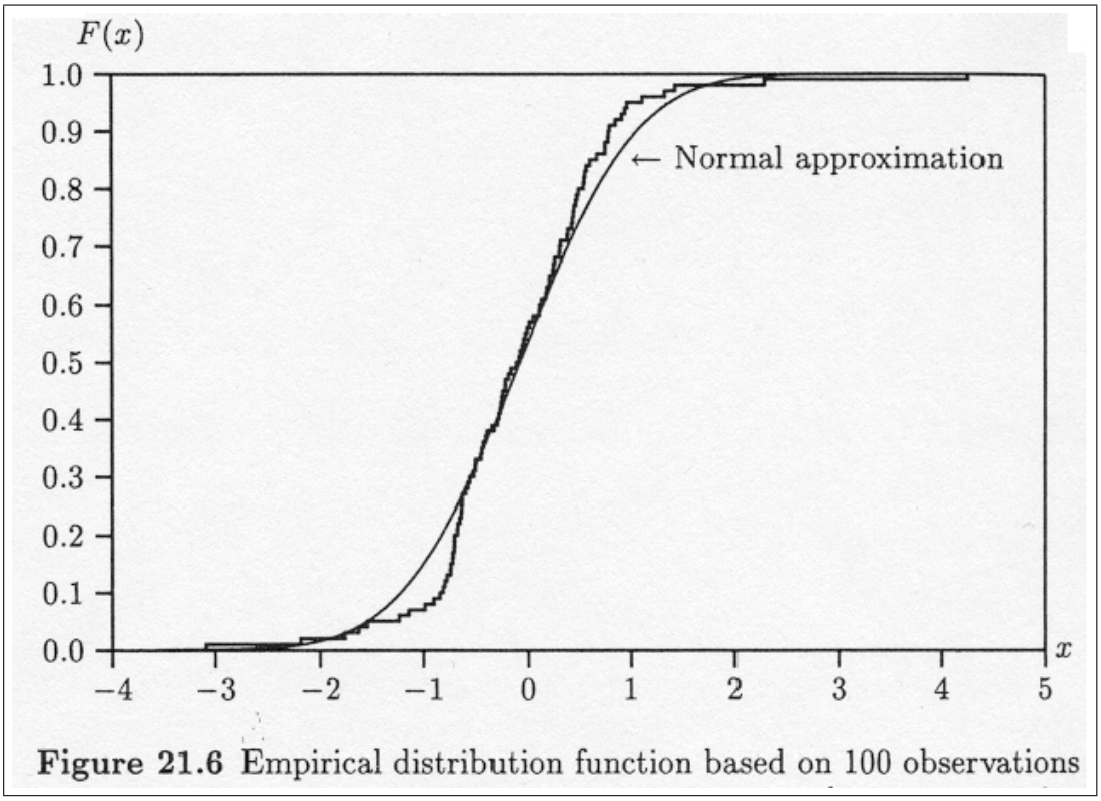
And now its time to step back and get some intuition. What's going on here and why does this make sense? We generated B = 50,000 samples, from our regular observed sample. The idea behind doing that is this: our observed sample is i.i.d, so it comes from observing T *actual* generations that came from the model. We observed T actual draws that really came from our model[2]. Suppose $T = 100,000$; suppose we had a huge sample. What that means is that we had 100,000 draws from the DGP. That, in most cases, gives us a really accurate empirical density! If we graph those 100,000 draws in a histogram, they'll probably fit the shape of the true density extremely closely. Suppose $T = 1$ *million*, or 1 *billion*, eventually the fit will be, for all practical purposes, perfect. So, our regular old i.i.d. random sample is actually an empirical density, and the bigger our sample is, the closer an approximation our empirical density will be to the true density.

Let's look at an example. Davidson and MacKinnon (1993) took 100 draws from a standard normal, so it's like they have an observed i.i.d. random sample from this data generating process, from this model. This sample of size 100, as we have said, actually comprises an empirical density, and once you have a density, you have a distribution. Specifically, the empirical distribution is this:

$$F_X^e(x) = \frac{\text{Number of observations whose value is less than } x}{\text{Total number of observations}}$$

The graph of this is a step function, and is displayed below, along with the true distribution. Note that even with just 100 observations, it turned out to be a fairly close fit, *so if we generated a random sample that was drawn from this empirical distribution it would not be that much different from generating a random sample drawn from the actual true distribution.* And, that is when our sample size is 100. As the sample size get's bigger and bigger, the difference between drawing from the true density and from our empirical one, will tend to get smaller and smaller, and in the limit there will be no difference at all.

[2] In frequentist statistics, the first step is to select a model that we have reason to believe is reasonably close to the true, and probably very messy, complicated, real data generating process. Then, for mathematical tests and analysis, we assume that the data actually came from this model. But, in the end, when we get the results from our mathematical tests and analyses, we balance them with how, and how much, we think our model did, in fact, differ from the reality that actually generated the data. Then, taking all of this into account, we make our conclusions and decisions.



From Davidson and MacKinnon (1993, pg. 764)

Thus, our plain old i.i.d. random sample of data is actually an approximation of the true density, and we can draw from this approximation of the true density just like we draw from our posterior in Bayesian inference, and to achieve the same goals. In fact, the analogy is really very close; the Bayesian posterior is the probability density – *given all that we currently know, which includes our observed sample of data.* If we have bad luck, the observed sample of data may be very unrepresentative of the true underlying data generating process, then our posterior may not be well concentrated around the true value of the parameter. But, that's statistics; there's always a chance we're off. We can have bad luck and a very unlikely, unrepresentative, sample can end up occurring. But, if our sample is "large", it is unlikely that we'll be too far off, and this is reflected in the narrowing variance of our posterior. The key is we don't know ex-ante if we had good or bad luck, so the posterior *is* our <u>best</u> guess, given *all* of the information that we *d*o have. It encompasses *all* of that information and with *complete* precision – given the assumptions of the model, the choice of prior, no errors in data gathering, no algebra errors, etc. The EDF is analogous. It encompasses *all* of the information available in the data, whether that's a lot or a little, with complete precision.

It's interesting to think of it this way. This is my own analysis, and I'm pressed for time, so I can't fully think this out, but looking at it this way makes Bootstrapping seem very efficient, or even completely efficient (although certain assumptions may be necessary. Here for example, remember, we are making the i.i.d. assumption, which for certain things may be necessary.). It does seem as though by generating standard errors of parameter estimates, confidence intervals of parameter estimates, etc., from bootstrap samples we are fully utilizing *all* of the information that the sample contains, nothing less and nothing more. It's like utilizing a Bayesian empirical posterior, with a non-informative prior. It would be interesting to see if mathematical equivalence could be proven. One note, however, for bootstrapping to really be "completely" efficient, to fully utilize *all* of the information that the sample contains, at least it would be necessary that the number of bootstrap samples, B, be infinite. So, "complete" efficiency could never be fully achieved, but we can look at how many bootstrap samples we need to achieve a "good" level of precision, or one that is better than the alternative techniques. This has been done in the literature; there are simulation studies determining reasonable levels of B in various cases. A very good discussion of this and of bootstrap methods in general can be found in Davison and Hinkley (1997).

Now getting back to the implementation, what we do in a basic bootstrap is this. We have some estimate of a model parameter, $\Psi_i(\mathbf{y})$. As the notation emphasizes, an estimate of a parameter is just some function of the sample data, and the sample data is i.i.d.. So, the estimat*or* $\Psi_i(\mathbf{Y})$ is just some function of the random variable Y. So, to generate draws of $\Psi_i(\mathbf{Y})$, we just generate size T sets of draws of Y from our empirical density for Y, that is from our observed sample. We do this B times, and then we have an empirical density for $\Psi_i(\mathbf{Y})$.

As we know from Bayesian/Monte Carlo theory, once we have the empirical density we have the empirical everything. So, for example, as stated earlier, the standard error can be estimated by equation (30).

Some terminology: the empirical distribution is also called the bootstrap distribution.

Another note: To really be efficient, prior information must be fully and accurately utilized also. Bayesians introduce at least some prior information in a formal mathematically structured and *invariably simplified* way. Frequentists consider prior information informally – *and* by their choice of parametric model. Bootstrapping works with parametric models, but does it fully utilize the information in the parametric assumptions? Time

does not permit analyzing this now, but it is something I plan on thinking about in the future. Post script: I later found out that there are non-parametric bootstrap methods, the one I described above is an example, and parametric bootstrap methods, which incorporate the information in the parametric assumptions at least to some extent, and perhaps fully. When assuming a parametric model it does seem like it will at least usually, if not always, be more efficient and appropriate to use a parametric bootstrap. McLachen and Krishnan (1997) give a parametric bootstrap method for obtaining standard errors with the EM algorithm. It is as follows:

Our empirical density will now reflect the parametric structure that we have assumed the data generating process has, by the model that we have chosen to represent, or approximate, the process of interest:

- Step 1: Estimate $\mathbf{\Psi}$ via the EM type algorithm that you decide to use to estimate the model. Let's call the resulting estimate $\widehat{\boldsymbol{\psi}}$.
- Step 2: We now make our empirical density $f\left(\mathbf{Y} \mid \widehat{\boldsymbol{\psi}}\right)$, or in other notation $f_{\widehat{\boldsymbol{\psi}}}\left(\mathbf{Y}\right)$. The actual density is $f\left(\mathbf{Y} \mid \boldsymbol{\psi}\right)$ where $\boldsymbol{\psi}$ is unknown. The empirical density is $f\left(\mathbf{Y} \mid \widehat{\boldsymbol{\psi}}\right)$, where $\widehat{\boldsymbol{\psi}}$ is the estimate we obtain from implementing our EM type algorithm.
- Step 3: Next, we take B bootstrap samples of size T from the empirical density $f\left(\mathbf{Y} \mid \widehat{\boldsymbol{\psi}}\right)$. Let's call those bootstrap samples $\mathbf{Y}^1, ..., \mathbf{Y}^B$.
- Step 4: For each of the bootstrap samples we re-estimate via our EM type algorithm $\mathbf{\Psi}$. So, that we obtain B bootstrap estimates of $\widehat{\boldsymbol{\psi}}$, $\widehat{\boldsymbol{\psi}}^1, ..., \widehat{\boldsymbol{\psi}}^B$. That is our empirical density for $\widehat{\boldsymbol{\psi}}$. From that we can estimate the standard errors of $\widehat{\boldsymbol{\psi}}$ as:

$$\widehat{cov}\left(\widehat{\boldsymbol{\psi}}\right) = \frac{1}{B-1}\sum_{b=1}^{B}\left(\widehat{\boldsymbol{\psi}}^b - \overline{\widehat{\boldsymbol{\psi}}}\right)\left(\widehat{\boldsymbol{\psi}}^b - \overline{\widehat{\boldsymbol{\psi}}}\right)'$$

According to McLachlan and Krishnan, "It has been shown that 50 to 100 bootstrap replications are generally sufficient for standard error estimation (Efron and Tibshirani 1993)." As we can see, bootstrap methods can be highly computationally intensive, and for the most part have only become feasible in recent decades. In some cases, they will still exceed the practical limits of current PCs. For such instances, it has been point out the natural fit of these methods to parallel processing, and that in many cases it will be worth the effort learn and do such programming in order to obtain the increased accuracy that bootstrap methods can provide.

It should be noted that time series characteristics must be taken into account in bootstrapping. If the model says that observations are correlated, then bootstrap samples cannot be drawn i.i.d. They must be drawn in a way that reflects the covariation implied by the model, and for this Markov chain Monte Carlo methods can be used. A good discussion of these issues can be found in Davison and Hinkley (1997) chapter 8.

Now, I present some evidence that bootstrap methods are highly regarded and considered to have great potential.

According to Kennedy (2003), "This procedure which estimates sampling distributions by using only the original data (and so 'pulls itself up by its own bootstraps') has proved to be remarkably successful. In effect it substitutes computing power, the price of which has dramatically decreased, for theorem proving, whose price has held constant or even increased as we have adopted more complicated procedures (pg. 69).

And, Chernick (1999) states, "Since the publication of *The Jackknife, the Bootstrap, and Other Resampling Plans* by Bradly Efron (1982), research activity on the bootstrap has grown exponentially. There have been many theoretical developments on the asymptotic consistency of bootstrap estimates and some counterexamples." (pg. 1). Chernick, in fact, was at Stanford with Efron in 1977 when he was working on his seminal paper, he comments, "At the time, the bootstrap seemed so simple and straightforward. many of us did not see it as part of a revolution in statistical methods that many of its advocates now see it as. Also, key results on the asymptotic properties of the bootstrap appeared to be difficult. The key papers by Bickel and Freedman (1981) and Singh (1981) had not yet appeared and Edgeworth and cornish-Fisher expansions were not yet recognized as useful tools." (pg. 3)

Bootstrap Methods have been prominently featured in prestigious statistics journals and compendiums. See Chernick (1999, pg. 4).

Julian Simon and Peter Bruce, two of the pioneers in bootstrap methods, created a software package called Resampling Stats, that can greatly facilitate the implementation of Bootstrap and other resampling methods. Information can be found at: www.resample.com. This website is also an excellent resource for resampling information in general, with links to books, research, and even teaching materials.

There are many variations of bootstrap, and other resampling methods, so it would take

a substantial amount of study to become really expert, but if the financial field will reward this expertise (in excess of how much they penalize one for taking the time to attain it), then I would be interested in further study. These techniques are very interesting, and look like they have the potential to, at least in some cases, improve substantially on current practice.

IV Bayesian Methods

Bayesian techniques have the powerful advantage of providing an entire joint posterior density for the parameter vector, usually in an easy to use emperical form, which makes tests of, and confidence intervals for, even complicated non-linear functions of the parameters easy. This is in stark contrast to frequentist methods. They also have the advantage of providing the *option* of incorporating in a mathematically formal, although invariably substantially simplifiied way, prior information. The value of this advantage is less clear. Although, essentially, since it is an option, it shouldn't decrease the value of the techniques. My personal opinion is that it can sometimes be a nice useful way to help analyze and make sense of a complicated set of prior and sample data, but it should rarely, if ever, completely replace informal analysis of prior data, it should just aid and supplement it. One specific way that it can be nice is in doing a kind of sensitivity, or "what if", analysis, with increasing levels of priors with regard to characteristics of interest.

Also, speaking more precisely, a proper prior may not always be optional. Sometimes results cannot be obtained without utilizing a proper prior. But, even there I don't know if frequentists can criticise. The priors required may be exactly equivelent to identification and other restrictions that they would require in the same cases (sorry, this is just an intuition. I have no time to even think about trying to prove this!). One example is the estimation of mixture models. A proper prior is always neccessary when using a Gibbs sampler with mixture models (see Diebolt and Robert 1994 and Eraker, Johannes, and Polson 2003).

Wei and Tanner (1990) give a very nice assesment of the advantage of posterior generating methods over standard frequentist ones, "The EM and MCEM algorithms provide a minimal amount of information to the data analyst. These algorithms yield the location of the normal approximation to the posterior distribution <but so does standard MLE, so this is not an argument for standard MLE in preference to EM type algorithms>...At the upper end are the data augmentation algorithm and the Gibbs sampler [Regarding the Gibbs sampler, see Geman and Geman (1984), Li (1988), and Gelfand and Smith (1990).]

The latter two algorithms are iterative and can be shown to converge to the true posterior distribution under mild regularity conditions." (pg. 703). That's a nice observation that EM type algorithms (and MLE) essentially give an assymptotically normal approximation of the posterior, as they use for standard errors, an assymptotic approximation based on the central limit theorem.

Bayesian techniques, as opposed to frequentist likelihood based methods do not typically require global optimization, per se. However, we don't escape the problems of complicated multi-modal likelihoods. We typically need to draw an emperical posterior via Markov chain Monte Carlo algorithms, and if it's complicated and multimodal the algorithm may never leave the vicinity of a local maximum, and we may not realize this. Visually, it may look as though the burn-in has been achieved once the algorithm settles around this local mode. This problem is quite analagous to the frequintist problem of identifying a global maximum among local ones, and perhaps just as serious. As far as difficulty, this can be a very challenging problem, but it may be solved easier than global optimization, easier in part because, at least in the limit, Markov chain Monte Carlo algorithms will usually roam over the entire posterior, no matter where we start, and converge to the stationary distribution, although to get close to this convergence, which we never completely reach, may take an inordiante amount of time, and it can be hard to determine when to stop, especially if we want to do it right. Most of the popular optimization routines, by contrast, will never converge to the global maximum if they are started too close to a local one.

It should be noted that although Bayesian Methods have some very attractive features, they do not make frequentist methods obselete. In some cases frequentist methods are more tractable and practical, but it's not just that. Another reason is that frequentist techniques can be combined with Bayesian techniques to produce a better result, that is a result with better ex-ante statistical properties. For example, Gelman and Rubin (1992) emphasize the value of finding the MLE, or maximum a posteriori (MAP), before using iterative simulations, because of the difficulties in assessing convergence of iterative simulation methods, especially when the regions of high density are not well known before hand (we will discuss Gelman and rubin at length in the next section).

As another example of how frequentist methods may sometimes be preferable, consider the MCEM algorithm. This algorithm involves taking draws from $Q(\psi, \psi_i)$ as opposed to taking draws from the likelihood function, as is common in Bayesian methods. Sometimes it may be very difficult or intractable to obtain adequate draws from the likelihood function, but it may be tractable and much easier to obtain adequate draws from $Q(\psi, \psi_i)$.

Examples of the use of Bayesian methods in estimating jump-diffusions are Eraker, Johannes, and Polson (2003) and Eraker (2004). Both use the Gibbs sampler in combination with Metropolis-Hastings algorithms. Therefore, they had to use proper priors, although they claimed that they were relatively flat, and offered simulation based evidence that they made little difference. Still, perhaps this does offer an opportunity to publish a paper which estimates jump-diffusions with the data augmentation algorithm. This would have the advantage of still generating an emperical posterior, but without having to use a proper prior. And, I mentioned earlier, two studies in top journals have shown the data augmentation model to be superior to the Gibbs in estimating mixture models, at least in certain cases. (Gelfand and Smith1990 and Diebolt and Robert1994).

V Generalized Method of Moments (GMM)

With this technique, moment conditions are established, that are analogous to those that are assumed to hold in the model. Normally, it will not be possible to satisfy all moment conditions completely, so the errors, the amounts off from achieving the moment conditions, are weighted in importance in a variance based way. Parameter estimates are chosen so as to minimize the weighted sum of squared errors. So, a global optimization is involved, but it should be an easier one than the one required for maximum likelihood estimation, and – a very important point – with GMM if a local optimum is mistaken for the global one, the estimator is *still* consistent. It's just less efficient. With MLE, if a local optimum is mistaken for the global one, the estimator is no longer consistent, and it may be way off.

Because the likelihood function encompasses the information in *every* moment, its efficiency can't be beat – *if it's done right* – but as discussed, what an author is claiming is the global maximum, or the bounded global maximum, may not be. For global optimization, in many meaningful situations, it can take a lot of time to learn to do it right, and/or to implement doing it right, and in the meantime there could be an attitude of "Why is this taking so long. I could have done this much quicker", or "This is not a very important part of the paper or of getting a publication. Why are you spending so much time on it". Although, perhaps the field isn't that much like this. That's something I'm going to have to find out. For now I just throw it out there, in hopes of getting some informative feedback.

Anyway, the bottom line is MLE is more efficient but harder to implement, and maybe not tractable, and if a researcher presents MLE results, there can and probably should be some skepticism about whether they are consistent. There's much less of that kind of worry

with GMM. GMM can also be used as a starting point and/or ball park check for MLE, as well as for EM type and Bayesian methods.

An important recent discovery by Aït-Sahalia (2004) is new moment conditions that he uses on a Merton (1976) model, which he shows achieve an asymptotic variance of the same order as MLE, although, of course, with a higher constant. If this kind of efficiency can be shown to hold in jump diffusions in general, this may be very important. GMM may then be a technique with very similar efficiency to MLE, but much more tractable and trustworthy.

VI Unlike the Physical Sciences, Does our Field Penalize Accuracy and Quality?

This is a question I have as a researcher starting out; perhaps the more experienced faculty can give me their opinions. A researcher can, for example, do simulated annealing in a very careful thoughtful way, or he can use a canned quasi Newton algorithm, and just keep trying different starting points until the algorithm gives him a point that looks economically sensible and/or interesting, especially since it appears that technical work is rarely ever looked at by referees and editors (see Veall 1990, McCullough and Vinod 1999, and Vinod 2001).

Or, a researcher using iterative simulation can use the elaborate multistage procedures outlined in Gelman and Rubin (1992), or he can just pick a sensible looking starting point and run the algorithm for 50,000 draws and cut it off where, on a plot, it looks relatively stable. If the results look economically sensible and/or interesting he keeps them and quickly moves on. If not, he just keeps trying different starting points until he gets a visual convergence that's economically reasonable looking and interesting; never mind its robustness; never mind that it's likely just clustered around one local mode of the posterior and hasn't touched most of the posterior. Perhaps he thinks it doesn't matter because this kind of thing won't be looked at anyway (please let me stress, I'm just asking. I'm just asking those with experience whether this happens, and if so how much).

If doing technical work reliably, precisely, and accurately, a la Gelman and Rubin (1992), means that a researcher will produce 50% less papers, or worse in his early years, when he will have to spend a lot of time to learn how to do things a la Gelman and Rubin (1992), yet he will receive no bonus for having done them technically precisely and accurately, then he takes a big risk of not getting tenure to provide this quality. So, I ask the faculty how

true they think this is, because if this is essentially the case, wouldn't the field be more useful if it produced less, but more careful and more accurate papers, and, along the same lines, if it produced less new papers, but more reproductions of existing papers to test, at least qualitatively, their accuracy, so we could get more accurate and credible research?

And, a specific question I have is, can you give any examples of researchers who have followed Gelman and Rubin's 1992 procedure, or one as elaborate as that (although once one has very extensive knowledge and experience in this area, such a procedure might not take that much longer than guessing a starting point, getting x-thousand draws, and looking at plots to guess the burn-in).

VII Numerical Issues

Briefly, with the kind of work we do, and given how computers store reals, there is constant round off. If the algorithm is stable, the round off will not build to unreasonable levels. However, an algorithm can be perfect analytically, with no coding errors, and still produce results that are substantially off, or even nonsense, because it is numerically unstable, because round off error grows too fast, or even exponentially. This is an issue that is taken very seriously in some of the sciences, although I am not sure how much in finance. Discussion can be found in Vinod (1999). The field that specializes in this is called numerical analysis. I have done a good deal of reading and learned some important caveats, rules, and techniques. Texts I recommend are Cheney and Kincaid (2003), Press, et. al. (1996), and Higham (2002). I know of one finance professor who has a Ph.D. in numerical analysis from Berkely, as well as a Ph.D. in finance from UCLA, Nai-Fu Chen, currently at The University of California Irvine. It would be great if we could have him out here.

VIII Conclusion

Estimating jump diffusions is a thorny problem, but perhaps solvable with reasonable effort, with the advances of the last 15 years in theory and computer power. I am going to try to see with my paper with Shu Yan and Pedro Santa Clara. Tentatively, for a frequentist estimation, the first technique I might recommend is the MCEM algorithm, possibly in a LIML framework, but possibly in a FIML. With this algorithm FIML may not be that much harder than LIML. For standard errors I might recommend a bootstrap method. For a Bayesian estimation, I might recommend a Gibbs sampler, or data augmentation algorithm.

References

Ait-Sahalia, Y. (forthcomming). Disentangling diffusions from jumps. *Journal of Financial Economics*.

Ball, C. and W. Torous (1983). A simplified jump process for common stock returns. *Journal of Financial and Quantatative Analysis 18*(1), 53–65.

Ball, C. and W. Torous (1985). On jumps in common stock prices and their impact on call option pricing. *The Journal of Finance XL*(1), 155–173.

Beckers, S. (1981). A note on estimating the parameters of of the diffusion-jump model of stock returns. *Journal of Financial and Quantatative Analysis 16*(1), 127–140.

Berndt, E., B. Hall, R. Hall, and J. Hausman (1974). Estimation and inference in nonlinear statistical models. *Anals of Economic and Social Measurement 3/4*, 653–665.

Bickel, P. and D. Freedman (1981). Some asymptotic theory for the bootstrap. *Annals of Statistics 9*, 1196–1217.

Bohning, D. and D. Nityasuddhi (2003). Assymptotic properties of the EM algorithm estimate for normal mixture models with component specific variances. *Computational Statistics and Data Analysis 41*, 591–601.

Celeux, G. and J. Diebolt (1985). The SEM algorithm: A probabolistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly 2*, 73–82.

Celeux, G. and J. Diebolt (1986). The SEM and EM algorithms for mixtures: Numerical and statistical aspects. *Procedings of the 7th Franco-Belgian Meeting of Statistics. Bruxelles: Publication Des Facultes Universitaries St. Louis*.

Chan, K. and J. Ledolter (1995). Monte carlo estimation for time series models involving counts. *Journal of the American Statistical Association 90*, 242–252.

Cheney, W. and D. Kincaid (2003). *Numerical Mathematics and Computing* (Fifth ed.). Brooks/Cole–Thompson.

Clara, P. S., R. Serlin, and S. Yan (2004). Risk and return in the stock market: Lessons from option data. *Working Paper*.

Davidson, R. and J. MacKinnon (1993). *Estimation and Inference in Econometrics*. Oxford University Press.

Davison, A. and D. Hinkley (1997). *Bootstrap Methods and their Application*. Cambridge University Press.

Delyon, B., M. Lavielle, and E. Moulines (1999). Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics 27*(1), 94–128.

Dempster, A., N. Laird, and D. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B 39*, 1–38.

Diebolt, J. and H. Ip (1996). *Markov Chain Monte Carlo in Practice: Chapter 15*. chapman and hall/CRC.

Diebolt, J. and C. Robert (1994). Estimation of finite mixture distributions through bayesian sampling. *Journal of the Royal Statistical Society B 56*(2), 363–375.

Efron, B. (1979). Bootstrapping methods : Another look at the jacknife. *Annals of Statistics 7*, 1–26.

Eraker, B. (2004). Do stock prices and volitility jump. *Journal of Finance LIX*(3), 1367–1403.

Eraker, B., M. Johannes, and N. Polson (2003). The impact of jumps in volitility and returns. *Journal of Finance LVIII*(3), 1269–1300.

Frost, D. (1993). The dual jump diffusion model for security prices. *Ph.D Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology*.

Gelfand, A. and A. Smith (1990). Sampling based approaches to calculating marginal densities. *Journal of the American Statistical Association 85*, 398–409.

Gelman, A. and D. Rubin (1992). Inference from iterative simulation using multiple sequences. *Statistical Science 7*, 457–472.

Goffe, W., G. Ferrier, and J. Rogers (1994). Global optimization of statistical functions with simulated annealing. *Journal of Econometrics 60*, 65–99.

Goldfeld, S. and R. Quandt (1972). *Nonlinear Methods in Econometrics*. Amsterdam: North-Holland.

Guo, S. and E. Thompson (1992). A monte carlo method for combined segregation and linkage analysis. *American Journal of Human Genetics 51*, 1111–1126.

Hamilton, J. (1994). *Time Series Analysis*. Princeton University Press.

Hartley, M. J. (1978). Comment. *Journal of the American Statistical Association 73*, 738–741.

Higham, N. (2002). *Accuracy and Stability of Numerical Algorithms* (Second ed.). SIAM.

Honore, P. (1998). Pitfalls in estimating jump diffusion models. *Unpublished Working Paper. Aarhus School of Business*.

Ip, E. (1994). A stochastic EM estimator in the presence of missing data – theory and applications. *Technical Report, Department of Statistics, Stanford University*.

Jamshidian, M. and R. Jennrich (1997). Acceleration of the EM algorithm by using quasi-newton methods. *Journal of the Royal Statistical Society B 59*(3), 569–587.

Jorion, P. (1988). On jump processes in the foriegn exchange and stock markets. *Review of Financial Studies 1*(4), 427–445.

Judd, K. (1998). *Numerical Methods in Economics*. Cambridge: MIT Press.

Keifer, N. (1978). Discrete parameter variation: Efficient estimation of a switching regression model. *Econometrica 46*, 427–434.

Koop, G. (2003). *Bayesian Econometrics*. John Wiley and Sons, Inc.

Lange, K. (1999). *Numerical Analysis for Statisticians*. Springer-Verlag New York, Inc.

Lee, P. (2004). *Bayesian Statistics: An Introduction*. Arnold, Edward.

Liu, C. and D. Rubin (1994). The ECME algorithm: A simple extension of EM and ECM with faster monotone convergence. *Biometrika 81*, 633–648.

Louis, T. (1982). Finding the observed information matrix when using the EM algorithm. *Journal of the Royal Statistical Society B 44*, 226–233.

McCullough, B. and H. Vinod (1999). The numerical reliability of econometric software. *Journal of Economic Literature XXXVII*, 633–665.

McLachlan, G. and T. Krishnan (1997). *The EM Algorithm and Extensions*. John Wiley and Sons.

Meng, X. (1994). On the rate of convergence of the ECM algorithm. *Annals of Statistics 22*, 326–339.

Merton, R. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics 3*, 125–144.

Metropolis, N., A. Rosenbluf, M. Teller, and E. Teller (1953). Equations of state calculations by fast computing machines. *Journal of Chemical Physics 21*, 1087–1092.

Mieljson, I. (1989). A fast improvement to the EM algorithm on its own terms. *Journal of the Royal Statistical Society B 51*(1), 127–138.

Murphy, K. and R. Topel (1985). Estimation and inference in two step econometric models. *Journal of Business and Economic Statistics 3*, 370–379.

Press, W., S. Teukolsky, W. Vetterling, and B. Flannery (1996). *Numerical Recipes in Fortran 77* (Second ed.). Cambridge University Press.

Quenouille, M. (1949). Approximate tests of correlation in time series. *Journal of the Royal Statistical Society B 11*, 18–84.

Ruud, P. (1991). Extensions of estimation methods using the EM algorithm. *Journal of Econometrics 49*, 305–341.

Singh, K. (1981). On the assymptotic accuracy of efrons bootstrap. *Annals of Statistics 9*, 1187–1195.

Tanner, M. and A. Wong (1987). The calculation of posterior distributions by data augmentation (with discussion). *Journal of the American Statistical Association 82*, 528–550.

Titterington, D., A. Smith, and U. Makov (1985). *Statistical Analysis of Finite Mixture Distributions*. New York: Wiley.

Veall, M. R. (1990). Testing for a global maximum in an econometric context. *Econometrica 58*(6), 1459–1465.

Vinod, H. (1999). The numerical reliability of econometric software. *Journal of Economic Literature XXXVII*, 633–665.

Vinod, H. (2001). Care and feeding of reproducable econometrics. *Journal of Econometrics 100*, 87–88.

Wei, G. and M. Tanner (1990). A monte carlo implementation of the EM algorithm and the poor mans data agumentation algorithms. *Journal of the American Statistical Association 85*, 699–704.