An Efficient Algorithm to Compute Maximum Entropy Densities

Dirk Ormoneit and Halbert White

Abstract— **We describe an algorithm to efficiently compute maximum entropy densities, i.e. densities maximizing the Shannon entropy $-\int p(x)\log p(x)dx$ under a set of constraints $E[g_i(x)] = c_i,\ i = 1, \ldots, n$. Our method is based on an algorithm by Zellner and Highfield, which has been found not to converge under a variety of circumstances. To demonstrate that our method overcomes these difficulties, we conduct numerous experiments for the special case $g_i(x) = x^i, n = 4$. An extensive table of results for this case is available on the World Wide Web.**

Keywords— **Density Estimation, Maximum Entropy Principle, Shannon Entropy**

I. Introduction

Maximum entropy densities have long been of interest as convenient and flexible tools for approximating probability distributions known only to satisfy certain moment properties. Typically, the task is to find a density $p(x)$ which maximizes the Shannon entropy $W = -\int p(x)\log p(x)dx$ under a set of moment constraints $E[g_i(x)] = c_i,\ i = 0, \ldots, n$.* Such constraints may result, for example, if the density $p(x)$ is to share the moments of a given sample. An example from the financial area is the estimation of maximum entropy risk-neutral densities for derivative assets where the restrictions are implied by the observed option prices at varying strike prices and maturities (e.g. [2], [3], [4], [5]). It is easily shown that the maximum entropy problem is solved by densities of the form

$$f(x,\lambda) \equiv e^{-\sum_{i=0}^{n}\lambda_i g_i(x)}, \tag{1}$$

for some unknown values $\lambda_0, \ldots, \lambda_n$. Depending on the moment functions $g_i(x)$, the maximum entropy density $f(x,\lambda)$ can take the form of several well-known members of the exponential famililiy, which include multimodal generalizations of the normal, gamma, inverse gamma, and beta distributions, as discussed by Cobb et al. in [6].

To find a suitable parametrization for (1) we need to compute values $\lambda^* = (\lambda_0^*, \ldots, \lambda_n^*)$ such that $f(x,\lambda^*)$ fulfills the constraints

$$G_i(\lambda) \equiv \int g_i(x) f(x,\lambda)dx = c_i. \tag{2}$$

The solution to this problem can be shown to be equivalent to that of the unconstrained optimization problem

$$\max_{\lambda_1,\ldots,\lambda_4} \tilde{\lambda}_0 = \log \int e^{-\sum_{i=1}^{n}\lambda_i(g_i(x)-c_i)}dx, \tag{3}$$

as can be seen immediately from inspection of the first order conditions for (3). Note that choosing $\lambda_0 = \tilde{\lambda}_0 - \sum_{i=1}^{n}\lambda_i c_i$ automatically ensures that the resulting maximum entropy density integrates to unity.

The problem of how to compute the "correct" λ^* has to our knowledge first been considered by Agmon et al. ([7]). Zellner and Highfield suggested an iterative algorithm for the special case where $g_i(x) = x^i,\quad n = 4$ ([8]). However, as was found by Maasoumi ([9]) as well as in our own experiments, this algorithm only converges to the solution for a small set of given moments c_i and even then requires the initial values for λ to be close to the final solution. The purpose of this paper is to describe a number of modifications and refinements to the algorithm of Zellner and Highfield that deliver a procedure which proved to compute the desired λ-values reliably and efficiently in our experiments.

II. A Maximum Entropy Density Estimation Algorithm

Zellner and Highfield apply Newton's method for nonlinear equations to solve (2).† The first step in the derivation of their procedure (called the ZH procedure in the following) is thus to take a Taylor expansion of the density moments and to drop the second and higher order terms. This leads to a linear equation system which is solved by the update step $\lambda' = \lambda + G^{-1}b$, where $G_{ij} \equiv \int g_i(x)g_j(x)e^{-\sum_{i=0}^{n}\lambda_i g_i(x)}dx$ and $b_i = c_i - G_i(\lambda)$. As may be shown, the Jacobian matrix G of this Newton step is positive definite, so that a unique solution exists. Unfortunately it turns out that the actual computation of λ^* with the ZH algorithm is infeasible in most cases for various reasons. First, the update step involves several numerical integrals, which imposes a major restriction on the precision of the updated approximation. Also, G has near-singularities in large regions of the λ-space. As a result, the algorithm is relatively unstable in the sense that it only converges for specific c-values and only leads to satisfactory results if started near the solution.

D. Ormoneit is a member of the Graduiertenkolleg at the Department of Computer Science at the Technische Universität München, Germany. During the research for this paper he was visiting at the Department of Economics at the University of California, San Diego. This visit was supported by a grant from the German Academic Exchange Service ("DAAD-Doktorandenstipendium aus Mitteln des zweiten Hochschulsonderprogramms"). E-mail: ormoneit@informatik.tu-muenchen.de .

Halbert White is Professor at the Department of Economics, University of California, San Diego. White's participation was supported in part by NSF grant SBR-9511253. E-mail: hwhite@albert.ucsd.edu.

*For notational convenience, we define $g_0(x) = 1$ and $c_0 = 1$ to impose the density constraint. Note, that for specific choices of n and $g_i(x)$ such a density might not exist, as for example in the case $n = 3$, $g_i(x) = x^i$ (see [1]).

†For a detailed review of numerical methods for optimization and nonlinear equations, see [10].

To derive a robust and efficient algorithm for the computation of λ^* we follow the procedure suggested by Gill ([11]) for the minimization of numerical integrals of form $\int_a^b h(x,\lambda)dx$ with respect to λ. One makes use of the fact that if a standard quadrature method is used to compute the numerical integrals involved in the computation of λ^*, one obtains an approximation which may be written as $\int_a^b h(x,\lambda)dx \approx \sum_{k=1}^{N-1} \Delta_k h(x_k,\lambda)$. Taking account of this approximation, the constraints (2) to be satisfied by λ^* become

$$G_i^N(\lambda) \equiv \sum_{k=1}^{N-1} \Delta_k g_i(x_k) f(x_k,\lambda) = c_i. \qquad (4)$$

Let λ^{N^*} denote a λ-value satisfying (4). To compute λ^{N^*}, we proceed entirely in analogy to Zellner and Highfield, i.e., first, we Taylor-expand $G_i^N(\lambda)$ and drop the second and higher order terms, yielding

$$G_i^N(\lambda') \approx G_i^N(\lambda) + \sum_{j=0}^{n} \frac{\partial G_i^N(\lambda)}{\partial \lambda_j}(\lambda_j' - \lambda_j). \qquad (5)$$

Then we substitute (5) into (4), yielding the linear equation system

$$G^N(\lambda)\cdot(\lambda' - \lambda) = b^N(\lambda), \qquad (6)$$

where $G_{ij}^N(\lambda) \equiv -\frac{\partial G_i^N(\lambda)}{\partial \lambda_j} = \sum_{k=1}^{N-1} \Delta_k g_i(x_k) g_j(x_k) f(x_k,\lambda)$ and $b_i^N(\lambda) \equiv c_i - G_i^N(\lambda)$. Solving (6) for λ' yields the update equation

$$\lambda' = \lambda + G^N(\lambda)^{-1} b^N(\lambda). \qquad (7)$$

The advantages of this apparently simple strategy are several. First, the derivatives involved in the optimization may be computed precisely, i.e. without errors due to numerical integration. A high precision of these derivatives is typically required to guarantee convergence in most Newton-like algorithms. Second, and even more importantly, we can set up an algorithm in which λ^{N^*} is first computed for a relatively simple problem, i.e. for a small N, and then use this solution as an initial value for successive optimizations with larger N. This procedure splits up an apparently complex estimation problem into several simpler substeps and also saves computation time because a significant part of the estimation may be completed while N is still relatively small.

Further, we can modify the update step (7) in such a way that convergence is guaranteed. This is possible because the Jacobian $G^N(\lambda)$ in the Newton step (7) is positive definite.‡ We combine Newton's method for nonlinear equations with a backtracking line search, which guarantees convergence if the backtracking algorithm is chosen appropriately. λ is updated according to $\lambda^r = \lambda + \alpha_r G^N(\lambda)^{-1} b^N(\lambda)$ for a sequence of values $\alpha_r = 2^{-r+1}, r = 1,2,\ldots$, until the condition $||b^N(\lambda^r)|| \leq (1-1/4r)\cdot||b^N(\lambda)||$ is satisfied, where $||\cdot||$ is the Euclidean norm. For a more detailed discussion of Newton's algorithm in combination with backtracking, see [10].

‡This is easily verified by noting that

$$\begin{aligned}\gamma' G^N(\lambda)\gamma &= \sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_i\gamma_j \sum_{k=1}^{N-1} g_i(x_k)g_j(x_k)e^{-\sum_{i=0}^{n}\lambda_i g_i(x_k)} \\ &= \sum_{k=1}^{N-1}\left(\sum_{i=1}^{n}\gamma_i g_i(x_k)\right)^2 e^{-\sum_{i=0}^{n}\lambda_i g_i(x_k)} \geq 0.\end{aligned}$$

Positivity is ensured as soon as $\sum_{i=1}^{n}\gamma_i g_i(x_k) \neq 0$ for some k.

III. Implementation and Experiments

We implemented and extensively tested the above algorithm both on an HP 9000 as well as a SUN Sparc 20 C++ programming environment. The task is to find appropriate λ-values for the special case $g_i(x) = x^i, n = 4$, which was a primary focus of the Zellner and Highfield paper. To simplify the numerical integration, we first transform a given set of non-central moments $\nu_1 \equiv E[X], \ldots, \nu_4 \equiv E[X^4]$ into the standardized central moments $\mu_1 = 0, \mu_2 = 1, \mu_3 = \frac{\nu_3 - 3\nu_2\nu_1 + 2\nu_1^3}{(\nu_2-\nu_1^2)^{3/2}}, \mu_4 = \frac{\nu_4 - 4\nu_3\nu_1 + 6\nu_2\nu_1^2 - 3\nu_1^4}{(\nu_2-\nu_1^2)^2}$. As initial guesses for $\lambda_0,\ldots,\lambda_4$ we used the corresponding values of the standard normal distribution. A first source of numerical difficulties is the evaluation of the exponential function in $G_i^N(\lambda)$. An efficient way to compute $G_i^N(\lambda)$ is described in the Appendix. As a quadrature method, we employed the extended midpoint integration rule (for a discussion, see Chapter 4.1 in [12]) after mapping the function domain $]-\infty,\infty[$ to the open interval $]-1,1[$. As a transformation function we used $z(x,\beta) = \frac{\beta x}{1+|\beta x|}$, such that $x_k = \frac{z_k}{\beta(1-|z_k|)}, z_k = -1 + \frac{2k+1}{N}$, and $\Delta_k = \frac{2}{N}\frac{(1+|\beta x_k|)^2}{\beta}$ in equation (4). In our experiments, we spent considerable effort on determining a suitable combination of the transformation function $z(x,\beta)$ and the sequence of N-values for which problem (4) is satisfactorily solved. The initial value for N should be sufficiently large that $G^N(\lambda)$ is guaranteed to possess positive eigenvalues, and that a (finite) solution to (4) exists. Simultaneously, N should not be too big, so as to keep the initial estimation problem simple. A related tradeoff arises for the choice of the transformation function $z(x,\beta)$ and the rate at which we allow N to grow. These should be chosen such that the integration boundaries do not diverge to infinity too fast, because this will lead to numerical instabilities near the boundary condition $\lambda_4 > 0$. For very fat-tailed distributions, on the other hand, large integration boundaries are a requirement to yield a valid approximation – a considerable computational effort may be required if the boundaries are growing at too small a rate. After numerous experiments we found $\beta = 3$ and the sequence $N = 128, 129, \ldots, 512$ to be a suitable choice.§ If one deviates from these particular values the algorithm does not deliver satisfactory results.

To guarantee that our algorithm converges for a minimum set of μ-values, as well as to yield a better understanding of the mapping of interest, we computed λ for

§The value $N = 512$ corresponds to integration boundaries of ± 170.3 (times the unit standard deviation). If the kurtosis is large, the solution might still change if the algorithm is continued beyond this value.

each μ in the range $\mu_3 \in [0,3], \mu_4 \in [\mu_3^2 + 1.1, 10]$ (for $\mu_1 = 0$ and $\mu_2 = 1$; increment 0.1).¶

We have summarized the results in a table, which may be downloaded from the World Wide Web.‖ Note that the reported λ-values depend sensitively on the quadrature method used. In order to compute integrals over $f(x, \lambda)$ using values from this table, one should take care to implement the quadrature method precisely as above. Incorrect values can easily result otherwise. In addition to the μ- and λ-values we provide time stamps for the completion of each computation on a SUN Sparc 20. As may be seen from the table on the Web, the computation time for particular μ-values varies from a few seconds to about four minutes.

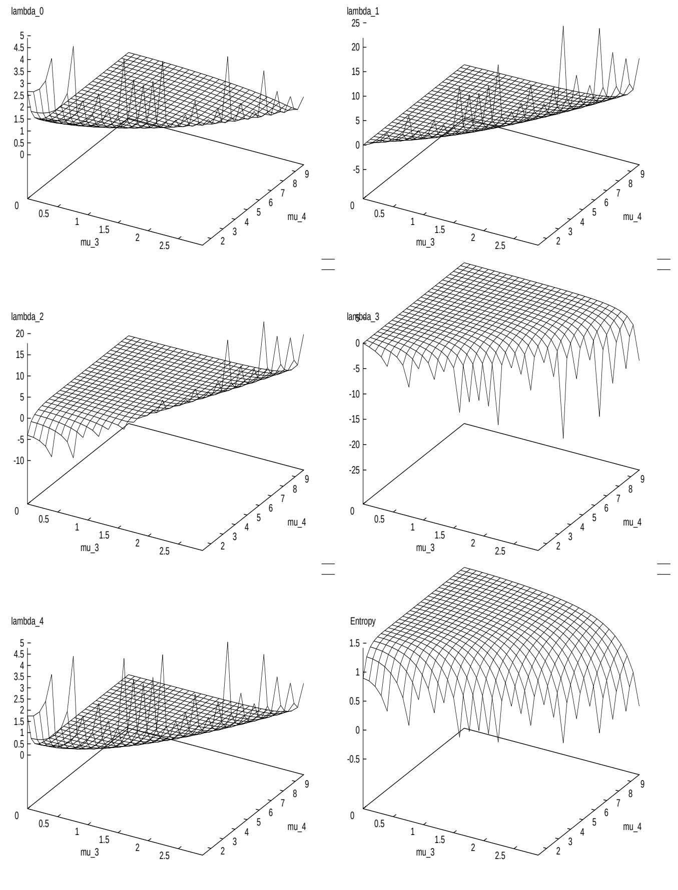

Fig. 1. *Upper left to lower left:* $\lambda_0, \ldots, \lambda_4$ in the range $\mu_3 \in [0,3], \mu_4 \in [\mu_3^2 + 1.1, 10]$ ($\mu_1 = 0$, $\mu_2 = 1$). *Lower right:* entropy.

The dependence of the λ-values as well as the entropy of f on μ_3 and μ_4 is graphically illustrated in Figure 1. Note that the λ-surfaces are very flat in almost the entire region and diverge quickly near the boundary $\mu_4 > \mu_3^2 + 1$. This is consistent with our observation that during the optimization $G^N(\lambda)$ is near-singular in a large portion of the λ-space and its elements diverge quickly when approaching the limiting conditions.

IV. Conclusion

We propose an algorithm to compute maximum entropy densities based on modifying the procedure of Zellner and Highfield. The central idea is to take explicit account of the numerical integration in this procedure, which enables us to split up the apparently complex estimation problem into a sequence of simpler tasks. We used this algorithm to compute a table and graphical representations of the necessary λ-values for a leading case. The algorithm proved to be reliable as well as computationally efficient for these tasks.

V. Acknowledgements

The authors greatly profited from discussions with Philip E. Gill, Essie Maasoumi, Gustavo Deco, and researchers at the Department of Economics at UCSD.

APPENDIX: A useful trick to compute $G_i^N(\lambda)$

One central issue in our algorithm is the computation of

$$G_i^N(\lambda) \equiv \sum_{k=1}^{N-1} \Delta_k x_k^i e^{-\sum_{i=0}^{n} \lambda_i x_k^i}. \tag{8}$$

This is a non-trivial task because the exponential functions provided by most programming environments only deliver meaningful results for a relatively small domain, say from -100 to +100. The exponent in (8) typically takes on values far outside this range. The idea is to write $G_i^N(\lambda) = e^{\zeta - q} \cdot \sum_{k=1}^{N-1} \Delta_k x_k^i e^{-\sum_{i=0}^{n} \lambda_i x_k^i - \zeta + q}$, where $\zeta = \max_k \left[-\sum_{i=0}^{n} \lambda_i x_k^i \right]$ and q is some constant. Obviously, this ensures that $\max_k \left[-\sum_{i=0}^{n} \lambda_i x_k^i - \zeta + q \right] = q$ and thus prevents the exponent from becoming too large. It is advisable to choose q as some large number to exploit the full range of numerical precision of the exponential function. ζ could in principle be determined prior to the actual function evaluation (which would involve computing the zeros of a third order polynomial), but it is easily updated dynamically after computing the exponent for each k. Of course, a dynamic change in ζ has to be compensated for by an adequate normalization of the previously aggregated values. To clarify this, we display a slightly modified version of the C++ function we used to compute $G_i^N(\lambda)$:

```
void compute_moments(long int N,DVec& lambda,DVec& mu,
        DVec& g,double& cf)  {  //=== 'DVec' are vectors
    int k, j;
    const double q = 10.0;
    double sum, x, x_j, w, z;
    double zeta, h;

    zeta = 0.0;
    g.set(0.0);
```

¶For $\mu_4 > 10$ we encountered some cases where the algorithm did not converge in its current implementation. The condition $\mu_4 > \mu_3^2 + 1$ is a prerequisite for the well-posedness of the maximum entropy density estimation problem. It follows from the conditions for the existence of a solution to the reduced moment problem, which is extensively discussed in [13]. We require that the matrix $M \equiv \begin{pmatrix} 1 & 0 & \mu_2 \\ 0 & \mu_2 & \mu_3 \\ \mu_2 & \mu_3 & \mu_4 \end{pmatrix}$ be positive definite. Setting the zeros of the characteristic polynomial of M greater than zero delivers the stated condition.

‖http://wwwbrauer.informatik.tu-muenchen.de/~ormoneit/lambda_table.txt

```
    for(k=0;k<=N-1;k++) {
        //=== compute z_k and x_k
        z = lb + (double) (2*k+1) / (double) N;
        x = z/(BETA*(1.0-ABS(z)));

        //=== sum = - (lambda_1 x + lambda_2 x^2 + ...)
        //    (ignore lambda_0 for the moment)
        x_j = x;
        sum = 0.0;
        for(j=1;j<=4;j++)  {
            sum += lambda[j] * x_j;
            x_j *= x;
        };
        sum *= -1.0;

        //=== this is the mentioned normalization
        if(k==0) zeta=sum;
        else if(sum>zeta)  {
            h = (double) exp(zeta-sum);
            g.mul(h);
            zeta = sum;
        };

        //=== multiply with Delta_k
        w = 2.0/(double) N * SQUARE(1.0+ABS(BETA*x))/BETA
             * exp(sum-zeta+q);

        //=== multiply with x_k^i
        x_j = 1.0;
        for(j=0;j<=8;j++)  {
            g[j] += x_j * w;
            x_j *= x;
        };
    };
    cf = -lambda[0] + zeta - q;
};
```

After the execution of this function, $G_i^N(\lambda)$ equals `g[i]*exp(cf)`. It is advisable, however, to check the size of `cf` first, and to treat the special case where `cf` is very large separately.

References

[1] T.M. Cover and J.A. Thomas, *Elements of information theory*, Wiley series in telecommunications, 1991.

[2] H. White, "Option pricing in modern finance theory and the relevance of artificial neural networks," in *Advances in Neural Information Processing Systems 8*, 1996, Tutorial Presentation.

[3] M.S. Makivić, "Valuation of derivative securities using the maximum entropy method," in *Computational Intelligence for Financial Engineering*, 1996.

[4] P.W. Buchen and M. Kelly, "The maximum entropy distribution of an asset inferred from option prices," *Journal of Financial and Quantitative Analysis*, vol. 31, pp. 143–159, 1996.

[5] M. Stutzer, "A simple nonparametric approach to derivative security valuation," *Journal of Finance*, 1996 (forthcoming).

[6] L. Cobb, P. Koppstein, and N.H. Chen, "Estimation and moment recursion relations for multimodal distributions of the exponential family," *Journal of the American Statistical Assosiation*, vol. 78, pp. 124–130, 1983.

[7] N. Agmon, Y. Alhassid, and R.D. Levine, *An Algorithm for Determining the Lagrange Parameters in the Maximum Entropy Formalism*, in *The Maximum Entropy Formalism*, R.D. Levine and M. Tribus (Editors), MIT Press, pp 207-209, 1981.

[8] A. Zellner and R.A. Highfield, "Calculation of maximum entropy distributions and approximation of marginal posterior distributions," *Journal of Econometrics*, vol. 37, pp. 195–209, 1988.

[9] E. Maasoumi, "A compendium to information theory in economics and econometrics," *Econometric Reviews*, vol. 12, pp. 137–182, 1993.

[10] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Classics in Applied Mathematics. SIAM, 1996.

[11] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, London and New York, 1981.

[12] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes*, Cambridge University Press, 1989.

[13] J.A. Shohat and J.D. Tamarkin, *The problem of moments*, American Mathematical Society, 1943.