Antithetic Monte Carlo Linear Solver

Chih Jeng Kenneth Tan

School of Computer Science
The Queen's University of Belfast
Belfast BT7 1NN
Northern Ireland
United Kingdom**
cjtan@acm.org

Abstract. The problem of solving systems of linear algebraic equations by parallel Monte Carlo numerical methods is considered. A parallel Monte Carlo method with relaxation parameter and dispersion reduction using antithetic variates is presented. This is a report of a research in progress, showing the effectiveness of this algorithm. Theoretical justification of this algorithm and numerical experiments are presented. The algorithms were implemented on a cluster of workstations using MPI.
Keyword: Monte Carlo method, Linear solver, Systems of linear algebraic equations, Parallel algorithms.

1 Introduction

One of the more common numerical computation task is that of solving large systems of linear algebraic equations

$$Ax = b \tag{1}$$

where $A \in \mathbb{R}^{n \times n}$ and $x, b \in \mathbb{R}^n$. A great multitude of algorithms exist for solving Equation 1. They typically fall under one of the following classes: direct methods, iterative methods, and Monte Carlo methods. Direct methods are particularly favorable for dense A with relatively small n. When A is sparse, iterative methods are preferred when the desired precision is high and n is relatively small. When n is large and the required precision is relatively low, Monte Carlo methods have been proven to be very useful [5, 4, 16, 1].

As a rule, Monte Carlo methods are not competitive with classical numerical methods for solving systems of linear algebraic equations, if the required precision is high [13].

In Monte Carlo methods, statistical estimates for the components of the solution vector x are obtained by performing random sampling of a certain random variable whose mathematical expectation is the desired solution [14, 18]. These

** Now at **SHARC**NET, Western Science Center, The University of Western Ontario, London, Ontario, Canada N6A 5B7.

P.M.A. Sloot et al. (Eds.): ICCS 2002, LNCS 2330, pp. 383–392, 2002.

techniques are based on that proposed by von Neumann and Ulam, extended by Forsythe and Liebler [13, 6].

Classical methods such as non-pivoting Gaussian Elimination or Gauss-Jordan methods require $\mathcal{O}(n^3)$ steps for a $n \times n$ square matrix [2]. In contrast, to compute the full solution vector using Monte Carlo the total number of steps required is $\mathcal{O}(nNT)$, where N is the number of chains and T is the chain length, both quantities independent of n and bounded [1]. Also, if only a few components of x are required, they can be computed without having to compute the full solution vector. This is a clear advantage of Monte Carlo methods, compared to their direct or iterative counterpart.

In addition, even though Monte Carlo methods do not yield better solutions than direct or iterative numerical methods for solving systems of linear algebraic equations as in Equation 1, they are more efficient for large n. Also, Monte Carlo methods have been known for their embarrassingly parallel nature. Parallelizing Monte Carlo methods in a coarse grained manner is very often straightforward. This characteristic of Monte Carlo methods has been noted even in 1949, by Metropolis and Ulam [12].

This paper presents an Antithetic Monte Carlo algorithm for the solution of systems of linear algebraic equations. Antithetic variates have been used in Monte Carlo algorithms for integral problems and solution of differential equations. But there are no known previous attempts to use antithetic variates in Monte Carlo algorithm for the solution of systems of linear algebraic equations.

2 Stochastic Methods for Solving Systems of Linear Algebraic Equations

Consider a matrix $A \in \mathbb{R}^{n \times n}$ and a vector $x \in \mathbb{R}^{n \times 1}$. Further, A can be considered as a linear operator $A\left[\mathbb{R}^{n \times 1} \rightarrow \mathbb{R}^{n \times 1}\right]$, so that the linear transformation

$$Ax \in \mathbb{R}^{n \times 1} \tag{2}$$

defines a new vector in $\mathbb{R}^{n \times 1}$.

The linear transformation in Equation 2 is used in iterative Monte Carlo algorithms, and the linear transformation in Equation 2 is also known as the iteration. This algebraic transform plays a fundamental role in iterative Monte Carlo algorithms.

In the problem of solving systems of linear algebraic equations, the linear transformation in Equation 2 defines a new vector $b \in \mathbb{R}^{n \times 1}$:

$$Ax = b, \tag{3}$$

where A and b are known, and the unknown solution vector x is to be solved for. This is a problem often encountered as subproblems on in various applications such as solution of differential equations, least squares solutions, amongst others.

It is known that system of linear algebraic equation given by Equation 3, can be rewritten in the following iterative form [2, 18, 4]:

$$x = Lx + b, \tag{4}$$

where

$$(I - L) = A. \tag{5}$$

Assuming that $\|L\| < 1$, and $x^0 \equiv 0$, the von Neumann series converges and the equation

$$\lim_{k\to\infty} x^{(k)} = \lim_{k\to\infty} \sum L^m b = (I - L)^{-1} b = A^{-1} b = x \tag{6}$$

holds.

Suppose now $\{s_1, s_2, \ldots, s_n\}$ is a finite discrete Markov chains with n states. At each discrete time $t = 0, 1, \ldots, N$, a chain S of length T is generated:

$$k_0 \to k_1 \to \ldots \to k_j \to \ldots \to k_T$$

with $k_j \in \{s_1, s_2, \ldots, s_n\}$ for $j = 1, \ldots, T$.

Define the probability that the chain starts in state s_α,

$$\mathbf{P}\ [k_0 = s_\alpha] = p_\alpha \tag{7}$$

and the transition probability to state s_β from state s_α

$$\mathbf{P}\ [k_j = s_\beta | k_{j-1} = s_\alpha] = p_{\alpha\beta} \tag{8}$$

for $\alpha = 1, \ldots, n$ and $\beta = 1, \ldots, n$.

The probabilities $p_{\alpha\beta}$ thus define the transition matrix P. The distribution $(p_1, \ldots, p_n)^T$ is said to be acceptable to vector h, and similarly that the distribution $p_{\alpha\beta}$ is acceptable to L, if [14]

$$\begin{cases} p_\alpha > 0 \text{ when } h_\alpha \neq 0 \\ p_\alpha \geq 0 \text{ when } h_\alpha = 0 \end{cases} \text{ and } \begin{cases} p_{\alpha\beta} > 0 \text{ when } l_{\alpha\beta} \neq 0 \\ p_{\alpha\beta} \geq 0 \text{ when } l_{\alpha\beta} = 0 \end{cases} \tag{9}$$

Define the random variables W_j according to the recursion

$$W_j = W_{j-1} \frac{l_{k_{j-1}k_j}}{p_{k_{j-1}k_j}},\ W_0 \equiv 1 \tag{10}$$

The random variables W_j can also be considered as weights on the Markov chain.

Also, define the random variable

$$\eta_T(h) = \frac{h_{k_0}}{p_{k_0}} \sum_{j=0}^{T-1} W_j b_{k_j}. \tag{11}$$

From Equation 6, the limit of $\mathbf{M}\ [\eta_T\,(h)]$, the mathematical expectation of $\eta_T\,(h)$ is

$$\mathbf{M}\ [\eta_T\,(h)] = \left\langle h, \sum_{m=0}^{T-1} L^m b \right\rangle = \left\langle h, x^{(T)} \right\rangle \Rightarrow \lim_{T\to\infty} \mathbf{M}\ [\eta_T\,(h)] = \langle h, x \rangle \tag{12}$$

Knowing this, one can find an unbiased estimator of $\mathbf{M}\left[\eta_\infty(h)\right]$ in the form

$$\theta_N = \frac{1}{N}\sum_{m=0}^{N-1}\eta_\infty(h) \tag{13}$$

Consider functions $h \equiv h^j = (0,0,\ldots,1,\ldots,0)$, where $h_i^j = \delta_i^j$ is the Kronecker delta. Then

$$\langle h, x\rangle = \sum_{i=0}^{n-1} h_i^j x_i = x_j. \tag{14}$$

It follows that an approximation to x can be obtained by calculating the average for each component of every Markov chain

$$x_j \approx \frac{1}{N}\sum_{m=0}^{N-1}\theta_T^m\left[h^j\right]. \tag{15}$$

In summary, N independent Markov chains of length T is generated and $\eta_T(h)$ is calculated for each path. Finally, the j-th component of x is estimated as the average of every j-th component of each chain.

3 Minimal Probable Error

Let I be any functional to be estimated by Monte Carlo method, θ be the estimator, and n be the number of trials. The probable error for the usual Monte Carlo method is defined as [14]:

$$\mathbf{P}\left[|I-\theta| \geq r\right] = \frac{1}{2} = \mathbf{P}\left[|I-\theta| \leq r\right]. \tag{16}$$

Equation (16) does not take into consideration any additional a priori information regarding the regularity of the solution.

If the standard deviation $(\mathbf{D}\left[\theta\right])^{\frac{1}{2}}$ is bounded, then the Central Limit Theorem holds, and

$$\mathbf{P}\left[|I-\theta| \leq x\left(\frac{\mathbf{D}\left[\theta\right]}{n}\right)^{\frac{1}{2}}\right] \approx \Phi(x). \tag{17}$$

Since $\Phi(0.6745) \approx \frac{1}{2}$, it is obvious that the probable error is

$$r \approx 0.6745\,(\mathbf{D}\left[\theta\right])^{\frac{1}{2}}. \tag{18}$$

Therefore, if the number of Markov chains N increases the error bound decreases. Also the error bound decreases if the variance of the random variable θ decreases.

This leads to the definition of almost optimal transition frequency for Monte Carlo methods. The idea is to find a transition matrix P that minimize the second moment of the estimator. This is achieved by choosing the probability proportional to the $|l_{\alpha\beta}|$ [5]. The corresponding almost optimal initial density vector, and similarly the transition density matrix $P = \{p_{\alpha\beta}\}_{\alpha,\beta=1}^n$ is then called the almost optimal density matrix.

4 Parameter Estimation

The transition matrix P is chosen with elements $p_{\alpha\beta} = \frac{|l_{\alpha\beta}|}{\sum_{\beta}|l_{\alpha\beta}|}$ for $\alpha, \beta = 1, 2, ..., n$. In practice the length of the Markov chain must be finite, and is terminated when $|W_j b_{k_j}| < \delta$, for some small value δ [14]. Since

$$|W_j b_{k_j}| = \left| \frac{l_{\alpha_0\alpha_1} \cdots l_{\alpha_{j-1}\alpha_j}}{\frac{|l_{\alpha_0\alpha_1}|}{\|L\|} \cdots \frac{|l_{\alpha_{j-1}\alpha_j}|}{\|L\|}} \right| |b_{k_j}| = \|L\|^i \|b\| < \delta\,, \tag{19}$$

it follows that

$$T = j \leq \frac{\log\left(\frac{\delta}{\|b\|}\right)}{\log \|L\|} \tag{20}$$

and

$$\mathbf{D}\left[\eta_T(h)\right] \leq \mathbf{M}\left[\eta_T^2\right] = \frac{\|b\|^2}{(1-\|L\|)^2} \leq \frac{1}{(1-\|L\|)^2}. \tag{21}$$

According to the Central Limit Theorem,

$$N \geq \left(\frac{0.6745}{\epsilon}\right)^2 \frac{1}{(1-\|L\|)^2} \tag{22}$$

is a lower bound on N.

5 Antithetic Monte Carlo Method

It has been discussed in Section 3 that, from Equation 18, the error bound may be decreased by increasing the number of samples, or by decreasing the dispersion in the random variable θ. As previously mentioned, that decreasing the error bound by decreasing the dispersion in the random variable θ is a more feasible solution. The emphasis on efficient Monte Carlo sampling dates back to the early days of digital computing [8], but are still as important today, since the problems solved are considerably large. A major part of Monte Carlo numerical methods research has, thus been dedicated to dispersion reduction. Note that dispersion reduction is also commonly known as variance reduction.

One of the techniques of dispersion reduction is by using antithetic variables. Consider the problem of estimating the integral

$$\theta = \int_a^b f(x)\, dx. \tag{23}$$

Define a random variable Y within the range (a, b), with a corresponding density $p(y)$, and a function g such that

$$\mathbf{M}\left[g(Y)\right] = \int_a^b g(y)\, p(y)\, dy = \int_a^b f(y)\, dy = \theta. \tag{24}$$

If the function g is taken to be f, and Y is defined to have a uniform density over the range $[a, b]$, therefore

$$\theta = (b-a)\ \mathbf{M}\left[f\left(Y\right)\right]. \tag{25}$$

By drawing N samples of $f\left(y_i\right)$, an estimate of θ, $\hat{\theta}$,

$$\hat{\theta} = \frac{1}{N}\left((b-a)\sum_{i=0}^{N-1} f\left(Y_i\right)\right), \tag{26}$$

may be obtained. The random variable θ is called the crude Monte Carlo estimate of θ.

It follows that the dispersion of the estimate $\hat{\theta}$, $\mathbf{D}\left[\hat{\theta}\right]$ is

$$\begin{aligned}\mathbf{D}\left[\hat{\theta}\right] &= \frac{1}{N^2}(b-a)^2 \sum_{i=0}^{N-1} \mathbf{D}\left[f\left(Y_i\right)\right] \\ &= \frac{(b-a)^2}{N}\mathbf{D}\left[f\left(Y\right)\right] \\ &= \frac{(b-a)}{N}\int_a^b \left(f\left(x\right)-\theta\right)^2 dx.\end{aligned} \tag{27}$$

If $\hat{\theta}$ is an estimator of θ, it is possible to find another estimator, $\hat{\theta}'$, having the same unknown expectation as θ and a strong negative correlation with θ [7]. The resulting estimate,

$$\tilde{\theta} = \frac{1}{2}\left(\hat{\theta} + \hat{\theta}'\right), \tag{28}$$

will be an unbiased estimator of θ. The dispersion of $\tilde{\theta}$ is

$$\mathbf{D}\left[\tilde{\theta}\right] = \frac{1}{4}\mathbf{D}\left[\hat{\theta}\right] + \frac{1}{4}\mathbf{D}\left[\hat{\theta}'\right] + \frac{1}{2}\mathbf{C}\left[\hat{\theta}, \hat{\theta}'\right], \tag{29}$$

where $\mathbf{C}\left[\hat{\theta}, \hat{\theta}'\right]$ is the covariance between $\hat{\theta}$ and $\hat{\theta}'$. Since $\hat{\theta}'$ is negatively correlated with $\hat{\theta}$, then the value of $\mathbf{C}\left[\hat{\theta}, \hat{\theta}'\right]$ will be negative, leading to an overall smaller $\mathbf{D}\left[\tilde{\theta}\right]$ by choosing $\hat{\theta}'$ suitably.

The random variable $\hat{\theta}'$ is the antithetic variate, and this technique of dispersion reduction is classified as antithetic dispersion reduction. In general, any set of estimators which mutually compensate each other's dispersions are termed antithetic variates [7].

In Monte Carlo methods for solving linear systems of algebraic equations, antithetic dispersion reduction may be used by generating a Markov chain which is negatively correlated with another.

As usual, the random variables W_j are defined according to the recursion in Equation 10 which is given here again:

$$W_j = W_{j-1}\frac{l_{k_{j-1}k_j}}{p_{k_{j-1}k_j}},\ W_0 \equiv 1.$$

In addition, define another set of random variables, W'_j as

$$W'_j = W'_{j-1} \frac{l_{k'_{j-1}k'_j}}{p_{k'_{j-1}k'_j}},\ W'_0 \equiv 1. \tag{30}$$

Denote the estimate of the solution, x, computed by using the first Markov chain as $\tilde{x}$, and the estimate computed by using the antithetic Markov chain as $\hat{x}$.

Also, define the random variable

$$\eta'_T(h) = \frac{h_{k'_0}}{p_{k'_0}} \sum_{j=0}^{T-1} W'_j b_{k'_j}, \tag{31}$$

in line with to Equation 11.

The limit of mathematical expectation of $\eta'_T(h)$, $\mathbf{M}\left[\eta'_T(h)\right]$ is the same as that of the mathematical expectation of $\eta_T(h)$:

$$\mathbf{M}\left[\eta'_T(h)\right] = \left\langle h, \sum_{m=0}^{T-1} L^m b \right\rangle = \left\langle h, \hat{x}^{(T)} \right\rangle \Rightarrow \lim_{T\to\infty} \mathbf{M}\left[\eta'_T(h)\right] = \langle h, \hat{x} \rangle \tag{32}$$

Then it follows that the unbiased estimator required will be will be one in the form

$$\theta'_N = \frac{1}{N} \sum_{m=0}^{N-1} \eta'_\infty(h). \tag{33}$$

With the functions $h \equiv j^i = (0, 0, \ldots, 1, \ldots, 0)$, where $h_i^j = \delta_i^j$ is the Kronecka delta, then the dot product of vectors h and x is

$$\langle h, \hat{x} \rangle = \sum_{i=0}^{n} h_i^j \hat{x}_i = \hat{x}_j, \tag{34}$$

as in Equation 14.

It the follows that the approximation to $\hat{x}$, using the antithetic Markov chain, can be obtained by calculating the average for each component of every Markov chain

$$\hat{x}_j \approx \frac{1}{N} \sum_{m=0}^{N-1} \theta'^m_T \left[h^j\right]. \tag{35}$$

The approximation to the solution vector x can be calculated by taking the average of the estimate $\tilde{x}$, computed with the first Markov chain, and $\hat{x}$, computed with the second, antithetic, Markov chain:

$$x = \frac{1}{2}(\tilde{x} + \hat{x}). \tag{36}$$

It does not take much effort to see that the Antithetic Monte Carlo method can be used on its own, or it can be used in combination with some, or all, of the previously proposed methods, namely, the almost optimal Monte Carlo method proposed by Megson, Alexandrov and Dimov in [9], the Monte Carlo method with chain reduction and optimization [3], and the Relaxed Monte Carlo method [15].

6 Numerical Experiments

A parallel version of the Antithetic Monte Carlo algorithm was developed using Message Passing Interface (MPI) [11, 10]. Version 1.2.0 of the MPICH implementation of the Message Passing Interface was used. As the programs were written in C, the C interface of MPI was the natural choice interface.

Table 1 show the results for experiments with the Monte Carlo method and Table 2 show the results for experiments with the Antithetic Monte Carlo method. The matrices used in these experiments were dense (general) randomly populated matrices, with a specified norm. The stochastic error, ϵ, and the deterministic error parameters were both set to 0.01. The `PLFG` parallel pseudo-random number generator [17] was used as the source of randomness for the experiments conducted.

Data set	**Norm**	**Solution time (sec.)**	**RMS error**	f No. chains
100-A1	0.5	0.139	4.76872e-02	454900
100-A2	0.6	0.122	4.77279e-02	454900
100-A3	0.7	0.124	4.78072e-02	454900
100-A4	0.8	0.127	4.77361e-02	454900
100-A5	0.5	0.137	3.17641e-02	454900
100-A6	0.6	0.124	3.17909e-02	454900
100-A7	0.7	0.124	3.17811e-02	454900
100-A8	0.8	0.119	3.17819e-02	454900
100-B1	0.5	0.123	3.87367e-02	454900
100-B2	0.6	0.126	3.87241e-02	454900
100-B3	0.7	0.134	3.88647e-02	454900
100-B4	0.8	0.125	3.88836e-02	454900
100-B5	0.5	0.121	2.57130e-02	454900
100-B6	0.6	0.119	2.57748e-02	454900
100-B7	0.7	0.120	2.57847e-02	454900
100-B8	0.8	0.126	2.57323e-02	454900

Table 1. Monte Carlo method with `PLFG`, using 10 processors, on a DEC Alpha XP1000 cluster.

7 Acknowledgment

I would like to thank M. Isabel Casas Villalba from Norkom Technologies, Ireland for the fruitful discussions and the MACI project at the University of Calgary, Canada, for their support, providing part of the computational resources used.

References

[1] Alexandrov, V. N. Efficient Parallel Monte Carlo Methods for Matrix Computations. *Mathematics and Computers in Simulation 47* (1998).

Data set	Norm	Solution time (sec.)	RMS error	f No. chains
100-A1	0.5	0.123	4.77496e-02	454900
100-A2	0.6	0.124	4.75949e-02	454900
100-A3	0.7	0.122	4.77031e-02	454900
100-A4	0.8	0.119	4.77248e-02	454900
100-A5	0.5	0.125	3.19065e-02	454900
100-A6	0.6	0.118	3.17608e-02	454900
100-A7	0.7	0.120	3.17462e-02	454900
100-A8	0.8	0.127	3.17776e-02	454900
100-B1	0.5	0.116	3.88795e-02	454900
100-B2	0.6	0.124	3.88260e-02	454900
100-B3	0.7	0.121	3.87219e-02	454900
100-B4	0.8	0.114	3.88401e-02	454900
100-B5	0.5	0.114	2.57260e-02	454900
100-B6	0.6	0.097	2.58110e-02	454900
100-B7	0.7	0.130	2.57297e-02	454900
100-B8	0.8	0.123	2.57341e-02	454900

Table 2. Antithetic Monte Carlo method with `PLFG`, using 10 processors, on a DEC Alpha XP1000 cluster.

[2] Bertsekas, D. P., and Tsitsiklis, J. N. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[3] Casas Villalba, M. I., and Tan, C. J. K. Efficient Monte Carlo Linear Solver with Chain Reduction and Optimization Using `PLFG`. In *High-Performance Computing and Networking, Proceedings of the9th. International Conference on High Performance Computing and Networking Europe* (2001), B. Hertzberger, A. G. Hoekstra, and R. Williams, Eds., vol. 2110 of *Lecture Notes in Computer Science*, Springer-Verlag.

[4] Dimov, I. Monte Carlo Algorithms for Linear Problems. In *Lecture Notes of the 9th. International Summer School on Probability Theory and Mathematical Statistics* (1998), N. M. Yanev, Ed., SCT Publishing, pp. 51 – 71.

[5] Dimov, I. T. Minimization of the Probable Error for some Monte Carlo Methods. In *Mathematical Modelling and Scientific Computations* (1991), I. T. Dimov, A. S. Andreev, S. M. Markov, and S. Ullrich, Eds., Publication House of the Bulgarian Academy of Science, pp. 159 – 170.

[6] Forsythe, S. E., and Liebler, R. A. Matrix Inversion by a Monte Carlo Method. *Mathematical Tables and Other Aids to Computation 4* (1950), 127 – 129.

[7] Hammersley, J. M., and Handscomb, D. C. *Monte Carlo Methods*. Methuen's Monographs on Applied Probability and Statistics. Methuen and Company, 1964.

[8] Kahn, H., and Marshall, A. W. Methods of Reducing Sample Size in Monte Carlo Computations. *Journal of Operations ResearchSociety of America 1* (1953), 263 – 278.

[9] Megson, G., Alexandrov, V., and Dimov, I. Systolic Matrix Inversion Using a Monte Carlo Method. *Journal of Parallel Algorithms and Applications 3*, 3 – 4 (1994), 311 – 330.

[10] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard*, 1.1 ed., June 1995.

[11] Message Passing Interface Forum. *MPI-2: Extensions to the Message-Passing Interface*, 2.0 ed., 1997.

[12] Metropolis, N., and Ulam, S. The Monte Carlo Method. *Journal of the American Statistical Association*, 44 (1949), 335 – 341.

[13] Rubinstein, R. Y. *Simulation and the Monte Carlo Method*. John Wiley and Sons, 1981.

[14] Sobol', I. M. *Monte Carlo Numerical Methods*. Moscow, Nauka, 1973. (In Russian.).

[15] Tan, C. J. K., and Alexandrov, V. Relaxed Monte Carlo Linear Solver. In *Computational Science (Part II)* (2001), V. N. Alexandrov, J. J. Dongarra, B. A. Juliano, R. S. Renner, and C. J. K. Tan, Eds., vol. 2074 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 1289 – 1298.

[16] Tan, C. J. K., and Blais, J. A. R. `PLFG`: A Highly Scalable Parallel Pseudo-random Number Generator for Monte Carlo Simulations. In *High Performance Computing and Networking, Proceedings of the 8th. International Conference on High Performance Computing and Networking Europe* (2000), M. Bubak, H. Afsarmanesh, R. Williams, and B. Hertzberger, Eds., vol. 1823 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 127 – 135.

[17] Tan, C. J. K., Casas Villalba, M. I., and Alexandrov, V. N. Accuracy of Monte Carlo Method for Solution of Linear Algebraic Equations Using `PLFG` and `rand()`. In *High Performance Computing Systems and Application* (2000), N. Dimopoulos and K. F. Li, Eds., Kluwer International Series in Engineering and Computer Science, Kluwer Academic Publishers. (To appear.).

[18] Westlake, J. R. *A Handbook of Numerical Matrix Inversion and Solution of Linear Equations*. John Wiley and Sons, 1968.