JOURNAL OF COMPUTATIONAL PHYSICS 61, 280–285 (1985)

An Efficient Newton's Method for the Numerical Solution of Fluid Integral Equations

GILLES ZERAH

Commissariat à l'énergie atomique, Centre d'études de Limeil-Valenton, Boîte Postale N° 27, 94190 Villeneuve-Saint-Georges, France

Received June 27, 1984; revised January 15, 1985

We propose a stable, straightforward algorithm for the numerical solution of integral equations for fluid pair distribution functions. The integral equation is not solved by Picard's standard iterative procedure but by Newton's method of solution of non-linear equations. The large matrix appearing in Newton's method is inverted by a conjugate gradient procedure used as a rapidly converging iterative method. © 1985 Academic Press, Inc.

I. INTRODUCTION

Methods for numerically solving fluid integral equations fall in two broad categories [1]:

(1) Picard's or fixed point methods which frequenctly require large numbers of iterations; and

(2) Newton's method [2].

The first method is not stable at high density, and one must resort to problem-dependent tricks to achieve convergence (see, e.g., [3]); the second method is more efficient but is plagued by the need of inverting a large (typically 1000×1000) matrix. Gillan [4] has recently proposed a combination of the two methods, thereby reducing the size of the matrix to be inverted. His procedure is efficient, but not straightforward to program.

Here we adapt Newton's method in its primitive form using an efficient way of inverting the matrix, taking full advantage of its very structure. In particular, we do not need to display the elements of the matrix effectively and hence do not require much storage. The resulting algorithm is easy to program.

II. STATEMENT OF THE PROBLEM AND FORMAL SOLUTION

Integral equations of the equilibrium theory of fluids have been used since the early sixties [1] to calculate the pair distribution function $g(r)$. These integral

0021-9991/85 $3.00

equations result from the combination of two relations. The first one is the Ornstein–Zernike (OZ) relation between the total and direct correlation function $h(r)=g(r)+1$ and $c(r)$. The second one is an approximate closure between those functions, examples of which are given by the Percus–Yevick (PY) or hypernetted chain (HNC) closure. Some recent generalisations may be found, e.g., in [5].

The problem can be stated in mathematical terms in the following manner:

Given: The inverse temperature $\beta=1/k_b T$, the number density ρ, and the pair potential $V(r)$

Find: t and c functions of $\mathbf{r}$, common roots of the two non-linear operators F_1 and F_2 which, in the case of the HNC closure, are explicitly given by

$$\begin{aligned} F_1(t, c) &= \exp(-\beta V(r)+t(r))-c(r)-t(r)-1 \\ F_2(t, c) &= \rho(\hat{t}(k)+\hat{c}(k))\,\hat{c}(k)-\hat{t}(k) \end{aligned} \tag{1}$$

where $\hat{}$ denotes the 3-dimensional Fourier transform

$$\hat{h}(k)=\int \exp(2i\pi\mathbf{k}\cdot\mathbf{r})\,h(\mathbf{r})\,d^3\mathbf{r}=\frac{2}{k}\int_0^{+\infty} r\sin(2\pi kr)\,h(r)\,dr.$$

Newton's method for solving this set of non-linear equations is the following:

construct a sequence of approximations: c_n, t_n

recursively defined by: $\delta t=t_{n+1}-t_n$, $\delta c=c_{n+1}-c_n$

where δt and δc are solutions of the set of inhomogeneous linear equations:

$$\begin{aligned} &\exp(-\beta V+t_n)\,\delta t-\delta c-\delta t=-F_1(c_n, t_n) \\ &\widehat{\delta t}-\rho(\hat{t}_n+\hat{c}_n)\,\widehat{\delta c}-\rho(\widehat{\delta t}+\widehat{\delta c})\,\hat{c}_n=F_2(c_n, t_n). \end{aligned} \tag{2}$$

The left-hand side of (2) results from differentiation of F_1 and F_2 with respect to t and c.

At this point, one could attempt to solve system (2) by Picard's iterative procedure: starting with a first estimate of δt (e.g., $\delta t=0$), compute δc from the first line, and then δt from the second, and so on. As this procedure does not converge well at high densities, one usually mixes input and output estimates to speed up convergence.

But the crucial fact is that the system (2) is *linear*, a property which allow us to use algorithms specially designed for these problems. The procedure used here, the conjugate gradient method (CG) [6] is a 2-term recurrence which solves iteratively a linear system in at most N iterations, where N is the dimension of the discrete representation of the functions c and t. We first describe the procedure and its application to the problem at hand.

III. Conjugate Gradient Method [6]

The aim is to solve the linear system: $AX = B$, where A is a linear operator, and X and B are functions.

To this end one constructs a sequence of mutually orthogonal remainders R_N, which eventually tend to zero, in the following way: Starting with $X_0 = X_{-1}$ initially given, construct recursively, R_N, X_{N+1} by

$$R_N = AX_N - B$$

$$X_{N+1} = X_{N-1} + W_{N+1}(\alpha_N \tilde{A} R_N + X_N - X_{N-1})$$

where the real quantites α_N, W_{N+1} are defined by

$$\alpha_N = (R_N, R_N)/(\tilde{A}R_N, \tilde{A}R_N)$$

$$W_{N+1} = [1. - \alpha_N (R_N, R_N)/\alpha_{N-1} W_N (R_{N-1}, R_{N-1})]^{-1}$$

with $W_1 = 1$. Here, (X, Y), denotes an inner product, and $\tilde{A}$ the adjoint of A is such that $(\tilde{A}X, Y) = (X, AY)$. It should be mentionned that here we use the method in its non-symmetric form, as an iterative procedure. If the problem is ill-conditioned the method should not give reliable results. In practice, this does not appear to be the case, although the number of iterations increases with density. A preconditioning of the operator should thus be desirable [6].

For the application to the particular case, one just needs to specify the operators A and $\tilde{A}$, which is the aim of the next paragraph.

IV. Application

In order to solve (2), we eliminate δc between the two lines [noting $h_n = \exp(-\beta V + t_n) - 1$], and obtain the following equation for δt:

$$\widehat{\delta t}(1 - \rho \hat{c}_n) - \rho(\hat{t}_n + 2\hat{c}_n)\,\widehat{h_n \delta t} = F_3(t_n, c_n) \tag{3}$$

where $F_3(t_n, c_n) = \rho(\hat{t}_n + 2\hat{c}_n)\,\hat{F}_1(t_n, c_n) + F_2(t_n, c_n)$.

For the operator A, we naturally select the left-hand side of (3), acting on $\widehat{\delta t}$. B is identified with F_3 and X with $\widehat{\delta t}$.

To compute easily the adjoint of A, we write it as a sum of two operators,

$$A = A_1 + A_2$$

$$A_1 : X \to (1 - \rho \hat{c}_n)\, X$$

$$A_2 : X \to -\rho(t_n + 2\hat{c}_n)\,\widehat{h_n \hat{X}}.$$

The scalar product is defined by : $(X, Y) = \int X(r)\, Y(r)\, r^2\, dr$ for which the Fourier transform is self-adjoint.

Now, for this scalar product, it is easy to compute the adjoint of A: we perform the sequence of Fourier transforms and products that constitutes A_1 and A_2 in reverse order,

$$\tilde{A} = \tilde{A}_1 + \tilde{A}_2$$

$$\tilde{A}_1 : Y \to (1 - \rho\hat{c}_n)\, Y$$

$$\tilde{A}_2 : Y \to Z = -\rho(\hat{t}_n + 2\hat{c}_n)\, Y \to \widehat{h_n \hat{Z}}.$$

In other words, $\tilde{A}_2 Y = \widehat{h_n \hat{Z}}$, where $Z = -\rho(\hat{t}_n + 2\hat{c}_n)\, Y$.

The whole algorithm may be summarized as follows:

Starting with a first estimate t_0, c_0, we can solve the system (3) using the method described in Section II: one generates a sequence of approximations of δt, until the remainder R_N is small enough. Then, with the help of the first line of (2), or from the closure relation, one easily gets δc, and one can thus construct

$$t_1 = t_0 + \delta t, \qquad c_1 = c_0 + \delta c.$$

The process is repeated until F_3 vanishes to the desired accuracy. One should note that at high density, the condition number of A is quite high, and we can reduce it by multiplying each member of (3) by $(1 - \rho\hat{c})^{-1}$. This is tantamount to work with $(1 - \rho\hat{c})^{-1} A$ instead of A, which was actually done.

V. Final Form and Results

For computational purposes, we need to discretize every function. We represent each function on a regular grid and compute Fourier transforms with the help of a fast Fourier transform routine.

TABLE I

Grid of 128 Points $\rho^* = 1.2$, $T^* = 1$[a]

N	Error[b]	Number of iterations Needed for the C.G. method
1	$\simeq 4.10^4$	26
2	$\simeq 1.10^4$	24
3	$\simeq 30$	23
4	$\simeq 1.10^{-2}$	21
	$< 2.10^{-8}$	

[a] Starting point: interpolation of solutions obtained at $\rho^* = 0.9$ and $\rho^* = 1.05$.

[b] Here, Error denotes $(F_3, F_3) = \sum_{I=1}^{N-1} F_3^2(I)\ I^2$, before the iteration number N.

TABLE II

Grid of 1024 Points $\rho^* = 1.2$, $T^* = 1$[a]

N	Error[b]	Number of Iterations needed for the C.G. method
1	$\simeq 2.10^4$	24
2	$\simeq 3.10^{-3}$	21
3	$\simeq 2.10^{-7}$	10
	$< 1.10^{-10}$	

[a] Starting point: interpolation on $\mathbf{r}$ of the solution t obtained in I on the grid of 128 points.

[b] Here, Error denotes $(F_3, F_3) = \sum_{I=1}^{N-1} F_3^2(I)\ I^2$, before the iteration number N.

In order to maintain orthogonality, each integral is computed via a trapezoidal rule. The specification of a discrete F.T. rule is sufficient to determine the discrete form of A and $\tilde{A}$ through the formulae (4) and (5).

Now, we illustrate our method with typical examples: Consider first an inverse power potential

$$v(r) = \varepsilon(\sigma/r)^{12}.$$

In reduced form, we have $X = r/\sigma$, $T^* = k_b T/\varepsilon$, and $\rho^* = \rho\sigma^3$, and we solve the Percus–Yevick equation for a density $\rho^* = 1.2$, and temperature $T^* = 1$.

We proceed by solving the equation first on a grid of 128 points for reduced densities ranging from $\rho^* = 0.$ to $\rho^* = 1.2$ at $T^* = 1$, using for each density the two previous solutions linearly extrapolated as starting point. At $\rho^* = 1.2$, we inter-

TABLE III

Grid of 1024 Points $\rho^* = 0.85$, $T^* = 0.719$[a]

N	Error[b]	Number of iterations needed for the C.G. method
1	$\simeq 3.10^5$	20
2	$\simeq 5.10^3$	21
3	$\simeq 2$	17
4	$\simeq 1.10^{-4}$	15
	$< 10^{-10}$	

[a] Starting point: interpolation of solation obtained at $T^* = 3$ and $T^* = 1.16$.

[b] Here, Error denotes $(F_3, F_3) = \sum_{I=1}^{N-1} F_3^2\ (I)\ I^2$, before iteration number N.

polate the solution obtained with 128 points on a grid of 1024 points and use this function as a new initial guess and then iterate until convergence is achieved to a prescribed accuracy. We present the typical results in Tables I and II. To allow comparison of the efficiencies of our procedure and Gillan's combination of Picard's and Newton's methods we have also investigated the case of a Lennard–Jones potential $v(r) = 4\varepsilon\,[(\sigma/r)^{12} - (\sigma/r)^6]$ for the HNC equation, on a grid of 1024 points in Table III.

VI. Conclusion

In our opinion the proposed method has three main advantages:

(1) *It is stable.* The reason for this is clear: first, Newton's method has a very high rate of convergence at any density, giving, when the solution is approached, an improvement of about three to five orders of magnitude at *each* step, and second, the conjugate gradient method is a safe method as it guarantees convergence in a *finite* number of steps. Those two ingredients are the reasons of the success of the method: it is essentially problem independent and no difficult parameter adjustment has to be made.

(2) *It is very well suited for computation on array processors.* Most of computation time is spent in Fourier transformations and scalar products. On array processors, those two operators are done in a very short time, making the whole process very fast. For instance, the aforementioned computation involving the initial computation on a 128-point grid and refinement on a 1024-point grid is performed in about 4 s on a CRAY 1 S computer for ten successive densities.

(3) *Its extension to mixtures is straightforward.* One has just to introduce the indices corresponding to each species and generalize the equation. Computation time is increased by a factor of four to five. Results for mixtures will be the object of a future publication.

References

1. J. P. Hansen and I. R. Mc Donald, "Theory of Simple Liquids," Academic Press, New York, 1976.
2. R. O. Watts, "Statistical mechanics," Chemical Society Specialist Periodical Report, Vol. 1, Chap. 1, Burlington House, London, 1973.
3. K. C. Ng, *J. Chem. Phys.* **61** (1974), 2680.
4. M. J. Gillan, *Mol. Phys.* **38** (1979), 1781.
5. F. J. Rogers and D. A. Young, *Phys. Rev. A* **30** (1984), 999.
6. J. Meurant and G. Golub, "Résolution numerique des grands systèms linéaires," Editions Eyrolles, Paris, 1983.