Fast Numerical Solution of a Kind of Nonlinear Integral Equations—Dyson-Schwinger Equations for Quark Propagator in Hadron Physics

Jinghui Huang[1,2,3], Xiangyun Hu[1,3,*], Huan Chen[4], Xueying Duan[5] and Guangjun Wang[5]

[1] School of Geophysics and Geomatics, China University of Geosciences (Wuhan), Lumo Road 388, Wuhan, 430074, China
[2] Department of Physics and Centre for Research in Photonics, University of Ottawa, 25 Templeton Street, Ottawa, Ontario, Canada
[3] State Key Laboratory of Geological Processes and Mineral Resources, China University of Geosciences, Wuhan, Hubei, 430074, China
[4] School of Mathematics and Physics, China University of Geosciences (Wuhan), Lumo Road388, Wuhan, 430074, China
[5] School of Automation, China University of Geosciences (Wuhan), Lumo Road 388 Wuhan, 430074, China
Email: jinghuihuang@cug.edu.cn(Huang), xyhu@cug.edu.cn (Hu), huanchen@cug.edu.cn (Chen), CUGxyDuan@163.com (Duan), gjwang@cug.edu.cn (Wang)

Abstract. The nonlinear integral equation has been widely studied and has become the heart of the matter in many scientific and engineering fields, such as seismology, optical fiber evolution, radio astronomy, and hadron physics with Quantum Chromodynamics. The Dyson-Schwinger Equations (DSEs) approach provides an essential nonperturbative approach to investigating the properties of hadrons and hot/dense quark matter. Mathematically, the Dyson-Schwinger Equations are a group of coupled nonlinear integral equations of quark propagators, gluon propagators, ghost propagators, and various vertices. On account of the non-linearity and singularity of the coupled equations, it is almost impossible to solve the DSEs analytically. One has to resort to the numerical solution of the equations, in which efficient fast algorithms are key points in practice. In this work, two improvements for numerically solving the nonlinear and singular integral equation for quark propagator in a vacuum are proposed. One is a modified interpolation method for unknown functions in the integral with high degrees of freedom. The other is the parallelization on CPUs with OpenMP in GCC Comparing the CPU times with different algorithms, our results indicate that our proposed methods can greatly improve the efficiency and reduce the computation time of the CPU.

Keywords: Dyson-Schwinger Equations, Nonlinear Integral Equation, OpenMP

1. Introduction

The Dyson-Schwinger Equations (DSEs) of Quantum Chromodynamics (QCD) are mathematically nonlinear and singular integral equations. They are an infinite set of coupled equations about quark propagators, gluon propagators, ghost propagators, and various vertex functions. They provide a

successful description of hadrons in vacuum and phase transitions in a hot or dense medium [1-2]. In literature phenomenological models with effective gluon and vertex input are adopted to truncate the coupled equations. Requirement on computing power increases quickly when equations for gluons and vertexes are included or finite temperature and density effects are included. Therefore, it is imperative to come up with more efficient methods for improving computational efficiency.

The DSEs are nonlinear and singular integral equations with high degrees of freedom. Therefore, the numerical solutions are difficult to obtain. The equations are normally formed as [3-4]:

$$\begin{aligned} u(x) &= f_1(x) + \int [K_1(x,t)F_{11}(u(t),v(x)) + \bar{K}_1(x,t)F_{12}(u(t),v(x))]dt, \\ v(x) &= f_1(x) + \int [K_1(x,t)F_{21}(u(t),v(x)) + \bar{K}_1(x,t)F_{22}(u(t),v(x))]dt, \end{aligned} \tag{1}$$

where $F_{ij}(u(x),u(t)(i,j= 1, 2)$ are composite functions nonlinearly dependent on functions u(x) and u(t), the kernels K_1 and K_2 are the nonsingular kernels, while the kernels $\bar{K}_1$ and $\bar{K}_2$ are the singular kernels given by

$$\bar{K}_{1,2} = 1/(x-t)^{\alpha} \tag{2}$$

The integral will be singular when the denominator in the right hand of Eq. (2) becomes zero at one or more points in integration. In our previous work [5-6], the integral is approximated as discretized summation with Gauss-Legendre integral formula. The numerical results of the integral equation can be obtained by a common iterative method with given initial test functions for u(x) and v(x). Specifically, the numerical computation method introduces the recurrence relation:

$$\begin{aligned} u_0(x) &= arbitrary\ \ real\ \ function \\ v_0(x) &= arbitrary\ \ real\ \ function \end{aligned} \tag{3}$$

$$\begin{aligned} u_{n+1}(x) &= f_1(x) + \int [K_1(x,t)F_{11}(u_n(t),v_n(x)) + \bar{K}_1(x,t)F_{12}(u_n(t),v_n(x))]dt, \\ v_{n+1}(x) &= f_1(x) + \int [K_1(x,t)F_{11}(u_n(t),v_n(x)) + \bar{K}_1(x,t)F_{12}(u_n(t),v_n(x))]dt, \end{aligned} \tag{4}$$

Consequently, the solution are obtained as the convergent values of the sequences

$$\begin{aligned} u(x) &= \lim_{n\to\infty} u_n(x) \\ v(x) &= \lim_{n\to\infty} v_n(x) \end{aligned} \tag{5}$$

Note that the singular kernels $\bar{K}_1$ and $\bar{K}_2$ make it difficult to integrate numerically. To avoid the singularity, the integral nodes t_i need to be different from x, and the values of the functions $u(t)$ and $v(t)$ need to be interpolated with high precision from u(x) and v(x). In the traditional interpolation process [7], such as Newton interpolation, and spline interpolation, the step to find the location of t in the discrete sequence x cost a lot of CPU time.

In the following, we intend to optimize and speed up the numerical calculation with two algorithm improvements. One is to modify the interpolation method, and the other is to paralyze our code with OpenMP in GCC. OpenMP has been very successful in exploiting structured parallelism in applications [8-9]. Particularly article [10] introduced the fundamental design of the OpenMP specification v2.5 in GCC. The implementation supports all the programming languages (C, C++, and Fortran), and it is

generally available on any platform that supports Portable Operating System Interface (POSIX) threads.

This paper is organized as follows. In section 2, we briefly introduce the truncation scheme of DSE for quark propagator in vacuum. In section 3, we describe our modified algorithms to solve the nonlinear and singular integral equations. In section 4, we compare the numerical results, especially the cost CPU time with different algorithms. In section 5, a short summary and outlook are given.

2. Quark's Dyson-Schwinger Equation with BC Vertex in Vacuum

In the four-dimensional momentum (p) space, the Dyson-Schwinger equation for quark propagator in vacuum can be given as:

$$S(p)^{-1} = Z_2(i\gamma\cdot\tilde{p} + m_q) + Z_1 g^2(\mu)\int \frac{d^4q}{(2\pi)^4} \times D_{\rho\sigma}(k)\frac{\lambda^a}{2}\gamma_\rho S(q)\Gamma^a_\sigma(q,p), \tag{6}$$

where the $D_{\rho\sigma}(k=p-q)$ represents the full gluon propagator, $\Gamma^a_\sigma(q,p)$ represents the dressed quark-gluon vertex. Z_1 is the renormalization constant for the quark-gluon vertex, while Z_2 is the quark wave-function renormalization constant, μ is the renormalization point. Normally, the quark propagator can be decomposed into two scalar functions $A(p^2)\ and\ B(p^2)$:

$$S^{-1}(p) = i\gamma\cdot pA(p^2) + B(p^2) \tag{7}$$

With the model inputs of the gluon propagator and the effective quark-gluon vertex [6, 11], equation (6) can be solved within the simplest truncation scheme. In vacuum, the models are usually taken as

$$Z_1 g^2 D_{\rho\sigma}(k)\Gamma^a_\sigma(q,p) = \varsigma(k^2)D^0_{\rho\sigma}(k)\frac{\lambda^a}{2}\Gamma_\sigma(p,q) \tag{8}$$

where $D^0_{\rho\sigma}(k) = \frac{1}{k^2}[\delta_{\rho\sigma} - \frac{k_\rho k_\sigma}{k^2}]$ corresponds to the Landau gauge free gluon propagator, $\varsigma(k^2)$ represents a model effective interaction, and $\Gamma_\sigma(q,p)$ is an effective quark-gluon vertex.

In our work, we use the well-known Ball-Chiu (BC) ansätz for the effective quark-gluon vertex, which satisfies the nonperturbative Ward-Takahashi identity. The BC vertex in vacuum was given in Ref [12].

$$\Gamma^{BC}_\sigma(q,p;\mu) = \lambda_1\gamma_\mu + \lambda_2(p+q)_\mu + \lambda_3(p+q)_\mu(p\cdot\gamma + q\cdot\gamma) + \lambda_4(p+q)_\nu\delta_{\mu\nu} \tag{9}$$

where $k = q - p,\ t = q + p$, λ_i (i=1,2,3,4) are composite functions of the scalar functions $A(p^2)$, $A(q^2)$, $B(q^2)$ and $B(p^2)$ of the quark propagator:

$$\lambda_1 = \frac{A(p^2)+A(q^2)}{2}, \lambda_2 = -i\frac{B(p^2)-B(q^2)}{p^2-q^2}, \lambda_3 = -\frac{1}{2}\frac{A(p^2)-A(q^2)}{p^2-q^2}, \lambda_4 = 0. \tag{10}$$

Note that the model effective interaction has been widely investigated [13]. In this paper, we employ a popular infrared-dominant model, denoted as the "HF" model, which indicates the long-range behavior of Qin-Chang (QC) model and is defined as [14]:

$$\varsigma^{QC}(k^2) = \frac{8\pi^2}{\omega^4}De^{-k^2/\omega^2} \tag{11}$$

Hence, the renormalization constants in Eq. (6) can be set to 1. With inputs of the effective quark-gluon vertex and the effective interaction model, the nonlinear and singular integral equations for the scalar functions $A(p^2)$ and $B(p^2)$ can be obtained:

$$\begin{aligned} A(p^2) &= z_1 + \int \frac{dq^4}{(2\pi)^4}\frac{\varsigma^{QC}(k^2)}{k^2p^2(p^2A^2(p^2)+B^2(p^2))}(I_{A1}+I_{A2}+I_{A3}), \\ B(p^2) &= m_0z_1 + \int \frac{dq^4}{(2\pi)^4}\frac{\varsigma^{QC}(k^2)}{k^2(p^2A^2(p^2)+B^2(p^2))}(I_{B1}+I_{B2}+I_{B3}), \end{aligned} \tag{12}$$

with

$$I_{A1} = -\frac{B(q^2)}{p^2}\frac{k^2(p\cdot t)-(k\cdot p)(k\cdot t)}{k^2p^2}\frac{B(q^2)-B(p^2)}{q^2-p^2},$$
$$I_{A2} = -\frac{A(q^2)}{2p^2}\frac{p^2(p\cdot t)k^2+q^2(p\cdot t)k^2-q^2(p\cdot k)(t\cdot k)-p^2(q\cdot k)(t\cdot k)}{k^2}\frac{A(q^2)-A(p^2)}{q^2-p^2},$$
$$I_{A3} = \frac{A(q^2)}{p^2}\frac{k^2(p\cdot q)+2(k\cdot q)(k\cdot p)}{k^2}\frac{A(q^2)+A(p^2)}{2},$$
$$I_{B1} = -A(q^2)\frac{(q\cdot t)k^2-(q\cdot k)(t\cdot k)}{k^2}\frac{B(q^2)-B(p^2)}{q^2-p^2},$$
$$I_{B2} = 3B(q^2)\frac{A(q^2)+A(p^2)}{2},$$
$$I_{B3} = B(q^2)\frac{k^2t^2-(k\cdot t)^2}{2k^2}\frac{A(q^2)-A(p^2)}{q^2-p^2}.$$

(13)

We could find the singular kernel $1/(q^2-p^2)$ in the term I_{A1}, I_{A2}, I_{B1} and I_{B3}. In traditional numerical calculation, the integral and singular equation can be solved by discretized with Gauss-Legendre integral formula. The initial functions are set to be: $A_0(p) = 1$, $B_0(p) = 1$. The convergence criteria of the iterated sequence is given by an accuracy ξ that at every momentum point:

$$A(p) = \lim_{n\to\infty} A_n(p) \to abs(A_{n+1}(p) - A_n(p)) < \xi,$$
$$B(p) = \lim_{n\to\infty} B_n(p) \to abs(B_{n+1}(p) - B_n(p)) < \xi,$$

3. Modified Interpolation Algorithm and Parallelization in GCC

The tricky problem in numerical computation is that the values of momentum p and q always take different values for eliminating the singularity. Therefore, the unknown functions $A(q^2)$ and $B(q^2)$ inside the integral need to be interpolated with high precision from $A(p^2)$ and $B(p^2)$, *which is quite time-consuming.* In the following, we optimize the numerical calculation with two algorithm improvements.

3.1. Modified Interpolation Algorithm

```
void main()
{
double Ap[N],Bp[N];
!!... some initialization code;
for(int i=0;i<N:i++)
{
Ap[i]=functionA(p[i]);
Bp[i]=functionB(p[i]);
}
}
```

```
void main()
{
double Ap[N],Bp[N];
!!... some initialization code;
# pragma omp parallel for
for(int i=0;i<N:i++)
{
Ap[i]=functionA(p[i]);
Bp[i]=functionB(p[i]);
}
}
```

Figure 1. Left : code of the "for" loop in a sequential process. Right: code of the "for" loop in a paralyzed process with OpenMP.

First we briefly introduce the traditional interpolation method. With traditional interpolation [7], such as Newton interpolation, and spline interpolation, one needs to find the location of q on the discrete data array $p[i] = (i = 1,2,3 \dots N)$, where N is the number of the discrete points of the outside momentum p.

$$q$$
$$\downarrow where?$$
$$p[1] < p[2] < \cdots < p[i-1] < p[i] < p[i+1] < \cdots < p[N]$$

Then with $p[i] < q < p[i+1]$ unknown function value $F(q^2)$= $A(q^2)$, $B(q^2)$ can be obtained with linear interpolation

$$F(q) = \begin{cases} F(p[i]) + (q - p[i]) \times \dfrac{F(p[i+1]) - F(p[i])}{p[i+1] - p[i]} & i < N \\ F(p[i]) & i = N \end{cases} \tag{14}$$

Normally the "searching step" costs a lot of CPU time due to many logical operations. In this work, we will use a modified interpolation method. In particular, the integral nodes-momentum q are set and saved as array $q[i] = (\mathrm{i} = 1,2,3\ldots \mathrm{N})$, with the relationship between array $p[i]$, and array $q[i]$:

$$p[1] < q[1] < p[2] < \cdots < p[i-1] < q[i-1] < p[i] < \cdots q[N-1] < p[N] < q[N] \tag{15}$$

In this modified interpolation method, we use equation (15) to replace the traditional "searching step", then a lot of CPU time for logical operations will be saved and the unknown function $F(q^2)$= $A(q^2)$, $B(q^2)$ can be obtained directly.

$$F(q[i]) = \begin{cases} F(p[i]) + (q[i] - p[i]) \times \dfrac{F(p[i+1]) - F(p[i])}{p[i+1] - p[i]} & i < N, \\ F(p[i]) & i = N, \end{cases} \tag{16}$$

3.2. Automatic Parallelization with OpenMP

In each step of the iteration, we need to calculate functions $A_{n+1}(p)$ and $B_{n+1}(p)$at every momentum p[i] which is done by a "for loop" as shown in the left panel of Figure 1. However, because N - the number of momentum p[i] is very large, it is quite time-consuming to complete the loop. In this work, we use OpenMP to paralyze our code, following the standard method in the article[10]. The code of a normal "for" loop is shown in the left panel of figure 1, where function A(p[i]) and function B(p[i]) are subroutines computing the right hand of the equation (13). Clearly, function A(p[i]) and function B(p[i]) for each p[i] run independently from those for other momenta. Therefore, we can split the "for" loop and assign them to multiple CPU cores. The right panel of Figure 2 shows how OpenMP helps us do it. The parallelization of the code helps us to split the "for" loop workload across multiple threads, with each thread running on different cores independently. Consequently, the total CPU running time can be significantly decreased.

4. Numerical Results

For numerically solving the coupled system in equation (13), two parameters D and ω in the effective interaction in equation (11) need to be fixed. We take the parameters D = 0.550 and ω = 0.678 from reference [5] which are determined by fitting meson properties with the BSE approach. With the above parameters and the ansätz, the DSE for quark propagator in vacuum with four different algorithms are investigated: Algorithm 1) traditional interpolation with sequential processing. Algorithm 2) modified interpolation with sequential processing. Algorithm 3) traditional interpolation with parallel processing. Algorithm 4) modified interpolation with parallel processing. For each algorithm, there are three parameters for controlling the computational accuracy and CPU time: the length of array $p[i] = (\mathrm{i} = 1,2,\ldots \mathrm{N})$, the number M of the nodes in the Gauss-Legendre integral integration formula, and the accuracy ξ for iteration convergence.

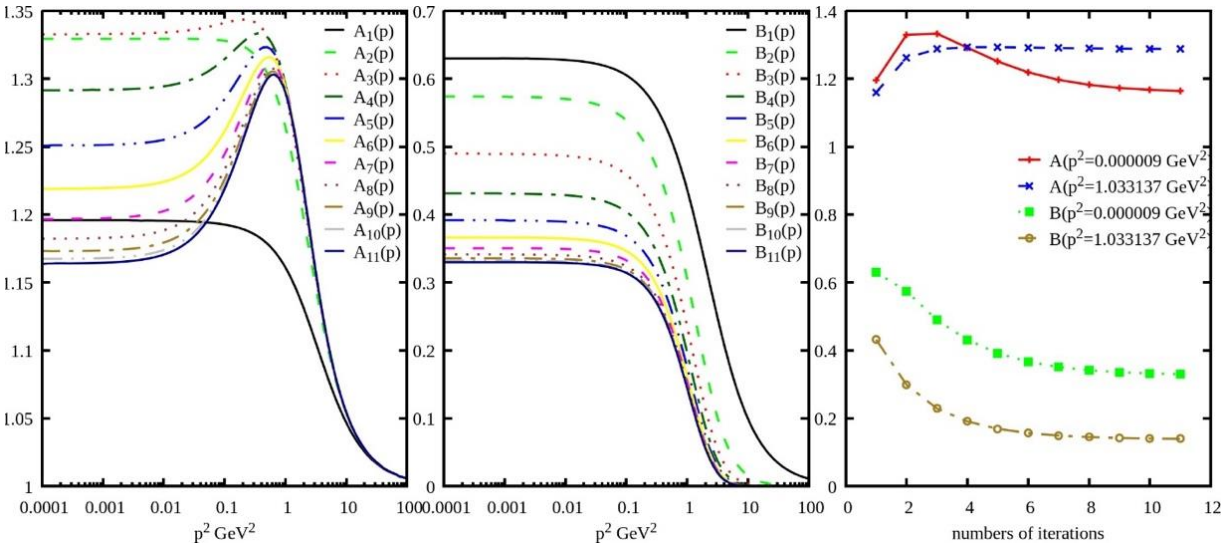

Figure 2. Left: quark's scalar function A(p) evolving with iterations. Middle: quark's scalar function B(p) evolving with iterations. Right: the quark's scalar function A(log p=-2.5), A(log p=0.0), B(log p=-2.5) and B(log p=0.0) evolving with iterations. The results are the same with all four algorithms.

Table 1. Costed CPU time with different algorithms. All codes are run on a computer with Inter(R) Core(TM) i7- @1.00GHz on GCC version 9.3.0 (Ubuntu 9.3.0-10).

N	M	ξ	algorithm	%CPU	iteration times	CPU time(s)
150	100	0.005	algorithm.1	100.0	11	65
150	100	0.005	algorithm.2	100.0	11	6
150	100	0.005	algorithm.3	782.0	11	9
150	100	0.005	algorithm.4	792.0	11	1

It is noted that the four algorithms lead to the same solution with the same number of iterations. In Figure 2 we show the quark's scalar functions A(p) and B(p) evolving in each iteration step. All are the same with different algorithms Our new algorithms just improve the efficiency of numerical computation, while the process and the result remain the same.

Table 1 shows the costed CPU time of our code with various algorithms. Clearly, both the modified interpolation method and parallelization with OpenMP can save a lot of computation time. Comparing the result of algorithm 2 and algorithm 1, the modified interpolation method speeds up the code 10 times. Comparing the result of algorithm.3 and algorithm.1, the parallelization with OpenMP speeds up the code about 7 times, which identifies with the computation ability of my computer (Intel i7). Comparing the result of algorithm.4 and algorithm.1, the combination of the modified interpolation method and the parallelization with OpenMP speeds up the code about 60 times.

5. Summary

In summary, two improved methods to numerically solve the nonlinear and singular integral equation of the quark propagator in vacuum are proposed. One is t a modified interpolation method, the other is the parallelization of our code with OpenMP in GCC. With our new algorithms, we can obtain the same results with the same accuracy, but speeds up to several tens of times, with our present hardware of computers. Our new algorithms can also be used in a more complex system, such as the coupled Dyson-Schwinger equations for both the quark propagator and the gluon propagator [6], or for the quark propagator at finite chemical potential and/or temperature [13]. In the latter case, the propagator functions will depend on more variables, and two-dimensional interpolation will cost more computation

time. Then the application of the two new algorithms will be more efficient for solving the complex system.

Acknowledgments
This study is supported by the National Science Foundation of China (No. 41630317), MOST Special Fund from the State Key Laboratory of Geological Processes and Mineral Resources, the China University of Geosciences (No. MSFGPMR01-4). J-H. Huang. acknowledges support from the China Scholarship Council (CSC No. 202206410063).

References
[1] Roberts C D, Richards D G, Horn T and Chang L 2021 Insights into the emergence of mass from studies of pion and kaon structure *Prog. Part. Nucl. Phys.* **120** 103883
[2] Eichmann G, Sanchis-Alepuz H, Williams R, Alkofer R and Fischer C S 2016 Baryons as relativistic three-quark bound states *Prog. Part. Nucl. Phys.* Pp **91** 1-100
[3] Erdogan F, Gupta G D and Cook T S 1973 Numerical solution of singular integral equations In: Sih, G.C. (eds) Methods of analysis and solutions of crack problems *Mechanics of Fracture* vol **1** Springer, Dordrecht
[4] Wazwaz A M 2011 *Linear and Nonlinear Integral Equations: Methods and Applications* (Berlin: Springer)
[5] Huang J H, Sun T T and Chen H 2020 valuation of pion-nucleon sigma term in Dyson-Schwinger equation approach of QCD *Phys. Rev. D* **101**(5) 054007
[6] Huang J H, Hu X Y, Wang Q, Duan X Y, Wang G J and Chen H 2022 Influence of light quark loops on the Wigner phase with Dyson–Schwinger equations approach *Eur. Phys. J. A* **58** 124
[7] Peckover R S 1971 Methods in computational physics *Physics Bulletin* **22** pp158-158
[8] Dagum L and Menon R 1998 Openmp: an industry standard api for shared-memory programming. *IEEE Computational ence and Engineering* **5** pp 46-55
[9] Ayguade E, Copty N, Duran A, Hoeflinger J, Lin Y, Massaioli F, Teruel X, Unnikrishnan P and Zhang G 2009 The design of openmp tasks. *IEEE Transactions on Parallel and Distributed Systems* **20** pp 404-418
[10] Klemm M, Bezold M, Gabriel S, Veldema R and Philippsen M 2007 Reparallelization and Migration of OpenMP Programs *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)* pp 529-540
[11] Fischer C S and Williams R 2009 Probing the gluon self-interaction in light mesons *Phys. Rev. Lett.* **103** 122001
[12] Ball J S and Chiu T W 1980 Analytic Properties of the Vertex Function in Gauge Theories *Phys. Rev*. D **22** 2542
[13] Roberts C 2008 Hadron properties and Dyson–Schwinger equations *Prog. Part. Nucl. Phys*. **61** pp 50-65
[14] Qin S X, Chang L, Liu Y X, Roberts C D and Wilson D J 2011 Interaction model for the gap equation *Phys. Rev. C* **84** 042202