ISSN: 2476-6909; Modares Mechanical Engineering. 2023;23(11):627-639.

Investigation of Iterative Solvers Parallelization for Systems of Linear Equations Resulting from the Poisson Equation

ARTICLE INFO

Article Type
Original Research

Authors
Mahmoodi Darian H.[1*]

How to cite this article
Mahmoodi Darian H, Investigation of Iterative Solvers Parallelization for Systems of Linear Equations Resulting from the Poisson Equation. Modares Mechanical Engineering; *2023;23(11):627-639*.

ABSTRACT

In the present article, a survey is carried out on the parallelization of several iterative solvers of the system of linear equations resulting from the discretization of the Poisson equation using the finite difference method. In particular, the point and line Gauss-Seidel successive over-relaxation methods, as well as the conjugate gradient and stabilized biconjugate gradient methods are investigated. For the over-relaxation methods, the optimum over-relaxation coefficient is used. The parallelization is first carried out on a multi-core central processor using C++ programming language and the OpenMP library, and then for a graphics processing unit using CUDA programming language. The results show, for both the two-dimensional and three-dimensional equations, the conjugate gradient methods due to a smaller number of iterations, have less computation time. Comparing the execution time of the different methods shows that for an 8-core processing, speedups of about 10 and 5 are achieved for the two- and three-dimensional equations, respectively. Furthermore, using a graphics processing unit leads to speedups between 5 and 10 in comparison to the 8-core processing.

Keywords Poisson Equation, Over-Relaxation, Conjugate Gradient Method, Parallel Processing, CUDA

CITATION LINKS

1- Iterative Solution of Large ... 2- Iterative methods for sparse ... 3- Linear solvers for power grid ... 4- A comparison of CPU and GPU ... 5- SADI approach programming on ... 6- An efficient GPU-based fractional-step ... 7- Parallel Thomas approach development ... 8- Assessment of WENO schemes for ... 9- Accelerating high-order WENO schemes ... 10- Assessment of WENO schemes for ... 11- A discontinuous Galerkin method with ... 12- Parallelizing SOR for GPGPUs using ... 13- Utilizing dynamic parallelism in ... 14- High Performance Implementation of Conjugate Gradient Method Using OpenCL on Graphics Processing Units 15- Parallel preconditioned conjugate gradient ... 16- A study of successive over-relaxation method ... 17- Parallel simulations for a ... 18- A new block parallel SOR method and ... 19- Computational fluid dynamics ... 20- Two-color Fourier analysis of ... 21- Fourier analysis of the ... 22- The optimal relaxation parameter for the ...

[1] Faculty of Engineering Science, College of Engineering, University of Tehran, Tehran, Iran.

***Correspondence**
Address: Faculty of Engineering Science, College of Engineering, University of Tehran, Tehran, Iran.

hmahmoodi@ut.ac.ir

Article History
Received: June 9, 2023
Accepted: December 11, 2023
ePublished: January 9, 2024

[DOI: 10.22034/mme.23.11.627]

بررسی موازی‌سازی حلگرهای تکراری دستگاه معادلات خطی حاصل از معادله پواسون

حسین محمودی‌داریان[۱]*

[۱] دانشکده علوم مهندسی، دانشکدگان فنی، دانشگاه تهران، تهران، ایران

چکیده

در مقاله حاضر یک بررسی بر موازی‌سازی چند حلگر تکراری دستگاه معادلات خطی حاصل از گسسته‌سازی معادله پواسون به روش تفاضل محدود انجام می‌شود. به طور خاص روش‌های تکراری فوق تخفیف گاوس سایدل نقطه‌ای و خطی و همچنین روش‌های گرادیان مزدوج و گرادیان دومزدوج پایدار شده بررسی می‌گردد. برای روش‌های فوق تخفیف از ضریب فوق تخفیف بهینه استفاده می‌شود. موازی‌سازی ابتدا برای یک پردازنده مرکزی چند هسته‌ای با زبان برنامه‌نویسی سی‌پلاس‌پلاس و کتابخانه اُپن اِم پی و سپس برای یک پردازنده گرافیکی با زبان برنامه‌نویسی کودا صورت می‌گیرد. نتایج حاصل از حل معادله دو بُعدی و همچنین معادله سه بُعدی نشان می‌دهد روش‌های گرادیان مزدوج در بیشتر موارد به علت تعداد تکرار کمتر زمان اجرای کمتری دارند. بررسی زمان اجرای روش‌های مختلف نشان می‌دهد در یک پردازش ۸ هسته‌ای نسبت به حالت تک هسته‌ای، افزایش سرعتی تا حدود ۱۰ و ۵ برابر به ترتیب در حل معادلات دو بُعدی و سه بُعدی حاصل می‌گردد. علاوه بر آن، استفاده از پردازنده گرافیکی نسبت به حالت ۸ هسته‌ای موجب افزایش سرعت بین ۵ تا ۱۰ برابر می‌شود.

کلیدواژه‌ها: معادله پواسون، فوق تخفیف، روش گرادیان مزدوج، پردازش موازی، کودا

تاریخ دریافت: ۱۴۰۲/۰۳/۱۹

تاریخ پذیرش: ۱۴۰۲/۰۹/۲۰

*نویسنده مسئول: hmahmoodi@ut.ac.ir

۱- مقدمه

معادله پواسون در بسیاری از معادلات حاکم در مسائل مهندسی نظیر انتقال حرارت، مکانیک سیالات، الکترواستاتیک و ... ظاهر می‌شود. حل عددی دستگاه معادلات جبری حاصل از گسسته‌سازی این معادله، با روش‌های تکراری انجام می‌شود. روش‌های تکراری مختلفی وجود دارند که از آنها می‌توان به روش‌های با قدمتی نظیر ژاکوبی، گاوس-سایدل و فوق تخفیف متوالی و روش‌های جدید نظیر گرادیان مزدوج و نسخه‌های مختلف آن اشاره کرد [1-3]. با ظهور پردازنده‌هایی با تعداد هسته‌های زیاد، به تدریج محققان به سمت تغییر کُدهای محاسباتی جهت استفاده از این پردازنده‌های جدید حرکت کردند. برای نمونه می‌توان به استفاده از پردازنده‌های گرافیکی برای حل معادله جابجایی-پخش [4] و حل معادلات سیال به روش‌های تفاضل محدود ضمنی [5-7]، روش‌های ضرورتاً غیرنوسانی وزن‌دار [8-10] و روش گالرکین ناپیوسته [11] اشاره نمود. در خصوص حل عددی معادله پواسون با روش‌های تکراری می‌توان به استفاده از پردازنده‌های گرافیکی برای روش فوق تخفیف [12, 13] و روش گرادیان مزدوج [14, 15] و همچنین استفاده از روش تقسیم دامنه برای پردازش چند هسته‌ای [16-18] اشاره کرد.

با وجود تحقیقات مختلفی که در حل عددی معادله پواسون صورت گرفته، مقایسه‌ای بین عملکرد روش‌های مختلف در پردازش موازی (تا جایی که نویسنده می‌داند) انجام نشده است. از سوی دیگر الگوریتم‌های مختلف قابلیت یکسانی جهت استفاده همزمان از چندین پردازنده را ندارند و در حل یک مسأله از راهکارهای متفاوتی باید استفاده نمود تا حداکثر بهره‌برداری از سخت افزار محاسباتی صورت گیرد. به همین علت، در مقاله حاضر یک بررسی و مقایسه میان روش‌های تکراری متداول در حل دستگاه معادلات خطی حاصل از گسسته‌سازی معادله پواسون به روش تفاضل محدود به صورت موازی و چالش‌های پیاده‌سازی آنها انجام می‌شود.

۲- گسسته‌سازی معادله پواسون

معادله پواسون در فضای دو بُعدی به صورت زیر است:

$$u_{xx} + u_{yy} = f(x,y) \qquad (۱)$$

گسسته‌سازی این معادله را به روش تفاضل محدود در یک ناحیه مربعی به ضلع واحد با شرایط مرزی دیریکله در نظر می‌گیریم [19]. شکل ۱ مشخصات شبکه عددی را نشان می‌دهد. متغیرهای گسسته را به صورت زیر تعریف می‌کنیم:

$$x_i = (i-۱)\Delta x, \quad y_j = (j-۱)\Delta y,$$

$$u_{i,j} \equiv u(x_i,y_j), \quad f_{i,j} \equiv f(x_i,y_j) \qquad (۲)$$

$$۱ \le i \le N_x, \quad ۱ \le j \le N_y,$$

$$\Delta x = \frac{۱}{N_x - ۱}, \quad \Delta y = \frac{۱}{N_y - ۱}$$

که در آن N_x و N_y تعداد نقاط شبکه و Δx و Δy اندازه شبکه به ترتیب در راستاهای افقی و عمودی هستند. با استفاده از روش تفاضل محدود مرکزی مرتبه دوم [19]، معادله به صورت گسسته زیر می‌شود.

$$\frac{u_{i+۱,j} - ۲u_{i,j} + u_{i-۱,j}}{\Delta x^۲} + \frac{u_{i,j+۱} - ۲u_{i,j} + u_{i,j-۱}}{\Delta y^۲} = f_{i,j} \qquad (۳)$$

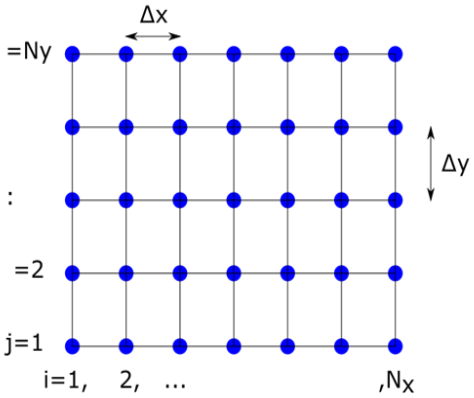
که با ضرب طرفین در $\Delta x^۲$ و تعریف $\beta = \Delta x/\Delta y$، به صورت زیر ساده می‌گردد.

$$u_{i+۱,j} + u_{i-۱,j} + \beta^۲(u_{i,j+۱} + u_{i,j-۱}) - ۲(۱+\beta^۲)u_{i,j} = \Delta x^۲ f_{i,j} \qquad (۴)$$

گسسته‌سازی معادله پواسون در فضای سه بُعدی به نحو مشابه انجام می‌شود. روابط (۵) و (۶) به ترتیب معادله پواسون و گسسته آن را در فضای سه بُعدی نشان می‌دهد.

$$u_{xx} + u_{yy} + u_{zz} = f(x,y,z) \qquad (۵)$$

[DOI: 10.22034/mme.23.11.627]



شکل ۱) مشخصات شبکه عددی

$$\begin{aligned} u_{i+1,j,k} + u_{i-1,j,k} + \beta^2\left(u_{i,j+1,k} + u_{i,j-1,k}\right) \\ + \gamma^2\left(u_{i,j,k+1} + u_{i,j,k-1}\right) \\ - 2\left(1 + \beta^2 + \gamma^2\right)u_{i,j,k} \\ = \Delta x^2 f_{i,j,k} \end{aligned} \qquad (۶)$$

که در آن $\gamma = \Delta x/\Delta z$ است. برد.

۳- روش‌های تکراری

در این بخش روش‌های تکراری برای حل معادله گسسته دو بُعدی (۴) را معرفی می‌کنیم و برای حالت سه بُعدی به نحو مشابه خواهد بود. در روش‌های تکراری یک حدس اولیه برای هر یک از مجهولات مشخص می‌شود و سپس آن را بهبود می‌دهند. با تکرار این روند (البته با شرایطی) به جواب همگرا می‌شوند. روش‌های ژاکوبی و گاوس-سایدل شناخته‌شده‌ترین روش‌های تکراری هستند که جهت حل دستگاه معادلات خطی معرفی می‌شوند [1, 2]. در روش ژاکوبی در هر تکرار مقدار جدید هر متغیر (مرحله $n+1$) با استفاده از مقادیر قبلی (مرحله n) سایر متغیرها به دست می‌آید:

$$\begin{aligned} u_{i,j}^{n+1} &= \frac{u_{i+1,j}^{n} + u_{i-1,j}^{n} + \beta^2\left(u_{i,j+1}^{n} + u_{i,j-1}^{n}\right)}{2(1+\beta^2)} - s_{i,j} \\ s_{i,j} &= \frac{\Delta x^2 f_{i,j}}{2(1+\beta^2)} \end{aligned} \qquad (۷)$$

$$u_{i,j}^{n+1} = \frac{u_{i+1,j}^{n} + u_{i-1,j}^{n+1} + \beta^2\left(u_{i,j+1}^{n} + u_{i,j-1}^{n+1}\right)}{2(1+\beta^2)} - s_{i,j} \qquad (۸)$$

در این رابطه حین محاسبه مقدار جدید u در نقطه (i,j) مقادیر جدید $u_{i-1,j}$ و $u_{i,j-1}$ قبلاً به دست آمده است. لذا مقادیر جدید آنها استفاده می‌شود. اما برای $u_{i+1,j}$ و $u_{i,j+1}$ ناگزیر مقادیر قبلی‌شان استفاده می‌گردد. سرعت همگرایی روش گاوس-سایدل دو برابر روش ژاکوبی است. با این حال سرعت همگرایی هر دو روش برای شبکه‌های ریز بسیار کند است. به همین علت از روش فوق تخفیف متوالی (Successive Over-Relaxation) [1, 2] استفاده می‌شود.

۳-۱- روش فوق تخفیف متوالی نقطه‌ای

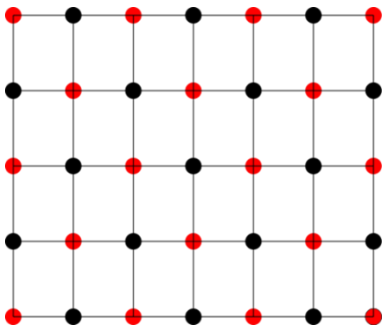
با اعمال فوق تخفیف [1, 2] می‌توان سرعت همگرایی را افزایش داد. روش فوق تخفیف به این صورت است که اختلاف مقدار جدید و قدیمی حاصل از رابطه تکرار (رابطه(۷) یا(۸)) با یک ضریب وزنی به مقدار قدیمی اضافه می‌شود:

$$u_{i,j}^{n+1} = u_{i,j}^{n} + \omega(u_{i,j}^{*} - u_{i,j}^{n}) \qquad (۹)$$

که در آن $u_{i,j}^{*}$ برابر $u_{i,j}^{n+1}$ در یکی از روابط (۷) یا (۸) است. به ضریب وزنی ω ضریب فوق تخفیف می‌گویند. محدوده همگرایی برای ضریب فوق تخفیف $0 < \omega < 2$ است [2]. مقدار بهینه ω بستگی به ابعاد شبکه عددی دارد. سرعت همگرایی روش فوق تخفیف گاوس-سایدل با مقدار بهینه ω بسیار بیشتر از روش فوق تخفیف ژاکوبی با مقدار بهینه ω است. اما مزیت روش ژاکوبی مناسب بودن آن برای پردازش موازی است. زیرا رابطه (۷) برای هر نقطه مستقل از نقاط دیگر قابل محاسبه است. اما روش گاوس-سایدل به صورت مرسوم رابطه (۸) امکان پردازش موازی ندارد. زیرا مقدار جدید هر نقطه به مقدار جدید دو نقطه دیگر ارتباط دارد. به همین علت روش گاوس-سایدل به صورت دیگری باید اعمال شود. یک روش، تقسیم‌بندی نقاط به دو دسته یا دو رنگ است [20]. به این صورت که نقاطی را که برای آنها مقدار $i+j$ زوج است، رنگ قرمز و نقاطی را که برای آنها مقدار $i+j$ فرد است، رنگ سیاه در نظر می‌گیریم. در این صورت مقدار جدید هر نقطه فقط وابسته به نقاط غیرهمرنگ و مستقل از نقاط همرنگ خودش است. بنابراین می‌توان ابتدا مقدار جدید نقاط قرمز را به صورت موازی محاسبه نمود و سپس به نحو مشابه مقدار جدید نقاط سیاه را به دست آورد:

$$\begin{aligned} u_{i,j}^{n+1} &= \frac{u_{i+1,j}^{n} + u_{i-1,j}^{n} + \beta^2\left(u_{i,j+1}^{n} + u_{i,j-1}^{n}\right)}{2(1+\beta^2)} \\ &\quad - s_{i,j} \qquad i+j = \text{زوج} \\ u_{i,j}^{n+1} &= \frac{u_{i+1,j}^{n+1} + u_{i-1,j}^{n+1} + \beta^2\left(u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1}\right)}{2(1+\beta^2)} \\ &\quad - s_{i,j} \qquad i+j = \text{فرد} \end{aligned} \qquad (۱۰)$$

شکل ۲چگونگی تقسیم‌بندی نقاط به دو رنگ قرمز و سیاه را نمایش می‌دهد. در رابطه (۱۰) برای نقاط قرمز (زوج) مقادیر مرحله n نقاط سیاه (فرد) استفاده شده است اما برای نقاط سیاه مقادیر مرحله $n+1$ نقاط قرمز استفاده می‌گردد.



شکل ۲) تقسیم‌بندی نقاط به دو رنگ قرمز و سیاه جهت موازی‌سازی روش گاوس-سایدل

[DOI: 10.22034/mme.23.11.627]

مقدار بهینه ω برای روش فوق تخفیف گاوس-سایدل برای هر دو رابطه (۸) و(۱۰) به صورت زیر به دست می‌آید [21]:

$$\omega = \frac{۲}{۱+\sqrt{۱-r^۲}} \qquad r = \frac{cos(\theta_x)+\beta^۲ cos\,(\theta_y)}{۱+\beta^۲} \tag{۱۱}$$

که در آن $\theta_x = \pi\Delta x$ و $\theta_y = \pi\Delta y$ است. شایان ذکر است در [20-22] مقدار بهینه ω برای شبکه مربعی $\Delta x = \Delta y$ ($\beta = ۱$) ارائه شده است و در اینجا ما با انجام تحلیل مشابه، حالت عمومی‌تر را ارائه کرده‌ایم. این رابطه نسبت به دو جهت تقارن دارد و جابجایی مقادیر N_x و N_y با یکدیگر، مقدار r و بالطبع مقدار ω را تغییر نمی‌دهد. نکته شایان ذکر این است که با استفاده از مقدار بهینه تعداد تکرارها به مراتب کمتر می‌شود. در این حالت می‌توان نشان داد تعداد تکرارها برای اینکه خطا یک دهم شود ($N_{۱۰}$)، به صورت زیر است:

$$N_{۱۰} = \frac{\sqrt{۱+\beta^۲}}{۲\sqrt{۲}\pi\Delta x} ln(۱۰) \tag{۱۲}$$

برای معادله سه بُعدی تقسیم‌بندی نقاط به دو رنگ قرمز و سیاه با توجه به زوج و فرد بودن مقدار $i+j+k$ انجام می‌شود. روابط ضریب فوق تخفیف بهینه نیز به نحو مشابه با تحلیل دو بُعدی [21, 22]حاصل می‌شوند:

$$\omega = \frac{۲}{۱+\sqrt{۱-r^۲}}$$
$$r = \frac{cos(\theta_x)+\beta^۲ cos\,(\theta_y)+\gamma^۲ cos\,(\theta_z)}{۱+\beta^۲+\gamma^۲} \tag{۱۳}$$

$$N_{۱۰} = \frac{\sqrt{۱+\beta^۲+\gamma^۲}}{۲\sqrt{۳}\pi\Delta x} ln(۱۰) \tag{۱۴}$$

جهت تمایز با روش فوق تخفیف خطی که در ادامه آورده می‌شود، روش‌های ذکر شده این بخش روش فوق تخفیف نقطه‌ای گفته می‌شود.

۳-۲- روش فوق تخفیف متوالی خطی

در روش گاوس-سایدل خطی، متغیرهای هر خط از شبکه (خط سطری یا خط ستونی) همزمان به روز می‌شوند [19]. برای هر خط یک دستگاه معادله سه قطری حل می‌گردد. همگرایی این روش سریع‌تر از روش نقطه‌ای است، اما نیاز به محاسبات بیشتری دارد. روابط (۱۵)و (۱۶) به ترتیب روش گاوس-سایدل خطی سطری و ستونی را نشان می‌دهند.

$$-u_{i+۱,j}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - u_{i-۱,j}^{n+۱} = \beta^۲\left(u_{i,j+۱}^{n} + u_{i,j-۱}^{n+۱}\right) - \Delta x^۲ f_{i,j} \tag{۱۵}$$

$$-\beta^۲ u_{i,j+۱}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - \beta^۲ u_{i,j-۱}^{n+۱} = u_{i+۱,j}^{n} + u_{i-۱,j}^{n+۱} - \Delta x^۲ f_{i,j} \tag{۱۶}$$

با تحلیلی مشابه آنچه در [21, 22] انجام شده است، ضرایب فوق تخفیف بهینه به دست می‌آید. روابط (۱۷)و (۱۸)مقدار بهینه ω را به ترتیب برای روابط (۱۵)و (۱۶)نمایش می‌دهند:

$$\omega = \frac{۲}{۱+\sqrt{۱-r^۲}} \qquad r = \frac{\beta^۲ cos\,(\theta_y)}{۱+\beta^۲-cos\,(\theta_x)} \tag{۱۷}$$

$$\omega = \frac{۲}{۱+\sqrt{۱-r^۲}} \qquad r = \frac{cos\,(\theta_x)}{۱+\beta^۲-\beta^۲ cos\,(\theta_y)} \tag{۱۸}$$

با این حال استفاده از روش فوق تخفیف خطی به صورت روابط (۱۵)و (۱۶) برای پردازش موازی مناسب نیست. برای پردازش موازی می‌توان مجدد سطرها (یا ستونها) را به دو رنگ قرمز و سیاه تقسیم‌بندی نمود. برای نقاط قرمز i زوج و برای نقاط سیاه i فرد است. در این صورت ابتدا سطرهای زوج و سپس سطرهای فرد همزمان حل می‌شوند. روابط (۱۹) روش گاوس-سایدل خطی سطری قرمز-سیاه را نشان می‌دهند:

$$\begin{aligned} &-u_{i+۱,j}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - u_{i-۱,j}^{n+۱} \\ &= \beta^۲\left(u_{i,j+۱}^{n} + u_{i,j-۱}^{n}\right) - \Delta x^۲ f_{i,j} \quad j = \text{زوج} \\ &-u_{i+۱,j}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - u_{i-۱,j}^{n+۱} \\ &= \beta^۲\left(u_{i,j+۱}^{n+۱} + u_{i,j-۱}^{n+۱}\right) - \Delta x^۲ f_{i,j} \quad j = \text{فرد} \end{aligned} \tag{۱۹}$$

و روابط (۲۰) روش گاوس-سایدل خطی ستونی قرمز-سیاه را نشان می‌دهند:

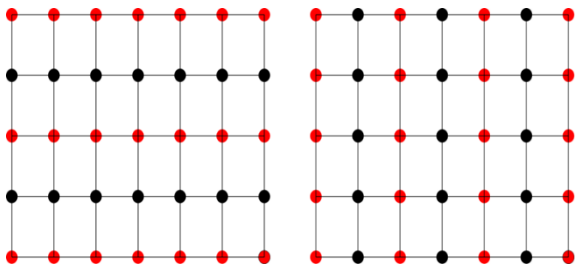
$$\begin{aligned} &-\beta^۲ u_{i,j+۱}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - \beta^۲ u_{i,j-۱}^{n+۱} \\ &= u_{i+۱,j}^{n} + u_{i-۱,j}^{n} - \Delta x^۲ f_{i,j} \quad i = \text{زوج} \\ &-\beta^۲ u_{i,j+۱}^{n+۱} + ۲(۱+\beta^۲)u_{i,j}^{n+۱} - \beta^۲ u_{i,j-۱}^{n+۱} \\ &= u_{i+۱,j}^{n+۱} + u_{i-۱,j}^{n+۱} - \Delta x^۲ f_{i,j} \quad i = \text{فرد} \end{aligned} \tag{۲۰}$$

شکل۳ چگونگی تقسیم‌بندی سطرها و ستون‌ها به دو رنگ قرمز و سیاه را نمایش می‌دهد.

مقدار بهینه ω مشابه روابط (۱۵)و (۱۶) است. در حالت سطری و ستونی تعداد تکرارها برای اینکه خطا یک دهم شود، به ترتیب به صورت زیر است:

$$N_{۱۰} = \frac{۱}{۲\sqrt{۲}\pi\Delta y} ln(۱۰) \tag{۲۱}$$

$$N_{۱۰} = \frac{۱}{۲\sqrt{۲}\pi\Delta x} ln(۱۰) \tag{۲۲}$$



شکل ۳) تقسیم‌بندی ستون‌ها (راست) و سطرها (چپ) به دو رنگ قرمز و سیاه جهت موازی‌سازی روش فوق تخفیف خطی

[DOI: 10.22034/mme.23.11.627]

به عبارت دیگر در حالت سطری تعداد تکرارها فقط وابسته به تعداد سطرها (N_y) و در حالت ستونی فقط وابسته به تعداد ستونها (N_x) است.

از آنجا که تفاوت خاصی میان نسخه سطری و ستونی روش فوق تخفیف خطی وجود ندارد، در اینجا صرفاً نسخه ستونی را در نظر می‌گیریم. جدول۱ روش‌های فوق تخفیف استفاده شده را برای معادله دو بُعدی در تحقیق حاضر نمایش می‌دهد.

جدول ۱) روش‌های فوق تخفیف استفاده شده

نام روش	رابطه	ضریب بهینه	$N_{۱۰}$
فوق تخفیف نقطه‌ای	(۸)	(۱۱)	(۱۲)
فوق تخفیف نقطه‌ای قرمز-سیاه	(۱۰)	(۱۱)	(۱۲)
فوق تخفیف خطی ستونی	(۱۶)	(۱۸)	(۲۲)
فوق تخفیف خطی ستونی قرمز-سیاه	(۲۰)	(۱۸)	(۲۲)

مقایسه رابطه (۱۲)و (۲۲) نشان می‌دهد در صورتی که β کوچک باشد، تعداد تکرار روش نقطه‌ای و روش خطی ستونی تقریباً یکسان خواهد شد. همچنین برای حالت شبکه مربعی ($\beta = ۱$) تعداد تکرار روش نقطه‌ای $\sqrt{۲} \cong ۱/۴۱$ بار بیشتر از روش خطی است.

برای معادله سه بُعدی، روش فوق تخفیف خطی را می‌توان برای هر یک از سه راستا استفاده نمود که در اینجا راستای z را در نظر می‌گیریم. تقسیم‌بندی خط‌ها به دو رنگ قرمز و سیاه با توجه به زوج و فرد بودن مقدار $i+j$ انجام می‌شود. روابط ضریب فوق تخفیف بهینه نیز به نحو مشابه حاصل می‌شوند:

$$\omega = \frac{۲}{۱+\sqrt{۱-r^۲}} \qquad r = \frac{cos(\theta_x)+\beta^۲ cos(\theta_y)}{۱+\beta^۲+\gamma^۲-\gamma^۲ cos(\theta_z)} \qquad (۲۳)$$

$$N_{۱۰} = \frac{\sqrt{۱+\beta^۲}}{۲\sqrt{۳}\pi\Delta x} ln(۱۰) \qquad (۲۴)$$

مقایسه رابطه (۱۴)و (۲۴) نشان می‌دهد برای شبکه مکعب مربعی ($\beta = \gamma = ۱$) تعداد تکرار روش نقطه‌ای $\sqrt{۱/۵} \cong ۱/۲۲$ بار بیشتر از روش خطی است.

۳-۳- روش گرادیان مزدوج

روش گرادیان مزدوج یک روش تکراری است که برای حل دستگاه معادلات خطی با ماتریس ضرایب متقارن و معین مثبت به کار می‌رود [2]. با توجه به اینکه ماتریس حاصل از رابطه (۴) متقارن و مثبت معین است، لذا می‌توان از این روش استفاده کرد. شایان ذکر است یکی از ویژگی‌های مهم این روش آن است که برای یک دستگاه با N معادله و N مجهول، در صورت نبودن خطای گرد کردن، حداکثر در N مرحله تکرار پاسخ دقیق معادله به دست می‌آید. اما در عمل با تعداد تکرار کمتر، نُرم خطا به حد مطلوب می‌رسد. یک مزیت این روش نسبت به روش فوق تخفیف گاوس-سایدل آن است که نیازی به تنظیم پارامتری نظیر ضریب فوق تخفیف برای همگرایی سریعتر ندارد. الگوریتم این روش برای حل دستگاه معادله خطی $Ax = b$ به صورت زیر است:

$$r^0 = b - Ax$$
$$p^0 = r^0$$
Loop
$$\alpha = \frac{\langle r^n, r^n \rangle}{\langle Ap^n, p^n \rangle}$$
$$x^{n+1} = x^n + \alpha p^n$$
$$r^{n+1} = r^n - \alpha A p^n \qquad (۲۵)$$
$$\text{if } \sqrt{\langle r^{n+1}, r^{n+1} \rangle} < \varepsilon$$
break
$$\beta = \frac{\langle r^{n+1}, r^{n+1} \rangle}{\langle r^n, r^n \rangle}$$
$$p^{n+1} = r^{n+1} + \beta p^n$$
End Loop

که در آن x^n بردار حدس مرحله n و r^n بردار مانده است. همچنین $\langle ۰,۰ \rangle$ بیانگر ضرب داخلی است.

برای استفاده از روش گرادیان مزدوج نیازی به تشکیل ماتریس ضرایب A نیست و فقط کافی است که نتیجه حاصل ضرب آن در یک بردار مشخص باشد. در حقیقت ماتریس A به مانند یک عملگر خطی در نظر گرفته می‌شود که ورودی و خروجی آن یک بردار است. در تحقیق حاضر برای معادله (۴) بردار مجهولات x شامل $u_{i,j}$ است و بنابراین خواهیم داشت:

$$Au_{i,j} = u_{i+۱,j} + u_{i-۱,j} + \beta^۲(u_{i,j+۱} + u_{i,j-۱}) - ۲(۱+\beta^۲)u_{i,j} \qquad (۲۶)$$
$$b_{i,j} = \Delta x^۲ f_{i,j}$$

موازی‌سازی رابطه (۲۶) به سهولت انجام می‌شود. اما چالش اصلی موازی‌سازی رابطه ضرب داخلی است که در قسمت نتایج توضیح داده می‌شود.

۴-۳- روش گرادیان دومزدوج پایدار شده

روش گرادیان دومزدوج پایدار شده از مشتقات روش گرادیان مزدوج است که برای ماتریس ضرایب غیرمتقارن نیز قابل استفاده است [2]. هرچند حجم محاسبات آن بیش از دو برابر روش گرادیان مزدوج است، اما همگرایی هموارتری دارد. الگوریتم این روش به صورت زیر است:

[DOI: 10.22034/mme.23.11.627]

$r^0 = b - Ax$

Choose $\hat{r}^0$ such that $\langle \hat{r}^0, r^0 \rangle \neq 0$

$p^0 = r^0$

$\rho^0 = \langle \hat{r}^0, r^0 \rangle$

Loop

$$\alpha = \frac{\rho^n}{\langle \hat{r}^0, Ap^n \rangle}$$
$$s = r^n - \alpha Ap^n$$
$$\omega = \frac{\langle As, s \rangle}{\langle As, As \rangle} \qquad (۲۷)$$
$$x^{n+1} = x^n + \alpha p^n + \omega s$$
$$r^{n+1} = s - \omega As$$
$$\text{if } \sqrt{\langle r^{n+1}, r^{n+1} \rangle} < \varepsilon$$
break
$$\rho^{n+1} = \langle \hat{r}^0, r^{n+1} \rangle$$
$$\beta = \frac{\alpha}{\omega}\frac{\rho^{n+1}}{\rho^n}$$
$$p^{n+1} = r^{n+1} + \beta(p^n - \omega Ap^n)$$

End Loop

۴- نتایج عددی

جدول ۲ مشخصات سخت افزار استفاده شده جهت ارائه نتایج را نشان می‌دهد. پردازنده مرکزی دارای ۸ هسته کارایی و ۸ هسته بهره‌وری است. هسته‌های کارایی هر یک دارای دو نخ هستند. به همین علت به صورت ۸+۱۶ در جدول ذکر شده است. از زبان برنامه‌نویسی سی‌پلاس‌پلاس (C++) و کتابخانه اُپن اِم پی (OpenMP) برای پردازنده مرکزی و زبان برنامه‌نویسی کودا (CUDA) برای پردازنده گرافیکی استفاده شده است.

جمله چشمه و حدس اولیه به صورت زیر در نظر گرفته شده است:

$$f(x,y) = x(۱ - x)y(۱ - y) \qquad (۲۸)$$

$$u^{۰}_{i,j} = ۳f(x_i, y_j) \qquad (۲۹)$$

شرط توقف یا معیار همگرایی، کاهش خطا به کمتر از $۱۰^{-۱۲}$ در نظر گرفته می‌شود. تعریف خطا با بیشینه اختلاف دو مقدار متوالی برای تمام نقاط شبکه، تعریف می‌شود.

$$E_n = \max_{i,j} |u^{n+۱}_{i,j} - u^n_{i,j}| \qquad (۳۰)$$

جدول ۳ برای چند شبکه مختلف تعداد تکرارهای صورت گرفته جهت همگرایی را برای روش‌های فوق تخفیف مختلف با ضریب بهینه متناظر نمایش می‌دهد. تعداد تکرار روش‌های گرادیان مزدوج نیز در جدول آورده شده است. مشاهده می‌شود که تعداد تکرارهای دو روش فوق تخفیف نقطه‌ای با یکدیگر تقریباً یکسان هستند. همین امر برای دو روش فوق تخفیف خطی نیز مشاهده می‌گردد. همان طور که پیش تر ذکر شد، در اینجا روش‌های فوق تخفیف خطی فقط به صورت ستونی در نظر گرفته شده است. در این حالت تعداد ستون‌ها (N_x) برابر تعداد دستگاه معادلات سه‌قطری و تعداد سطرها (N_y) برابر تعداد مجهولات هر دستگاه است و بنابراین β برابر نسبت تعداد مجهولات هر دستگاه به تعداد دستگاه‌ها می‌باشد. در جدول مشاهده می‌شود که تعداد تکرارهای روش‌های فوق تخفیف خطی صرفاً وابسته به تعداد دستگاه معادلات است. هرچه β بزرگ‌تر باشد، تعداد تکرارهای روش‌های نقطه‌ای نسبت به روش‌های خطی ستونی افزایش می‌یابد. از سوی دیگر، در صورتی که β کوچک باشد، همان طور که پیش‌تر با مقایسه رابطه (۱۲)و (۲۲) ذکر شد، تعداد تکرار روش نقطه‌ای و روش خطی ستونی تقریباً یکسان خواهد شد که این مورد در جدول ۳ نیز مشاهده می‌گردد.

بررسی روش‌های گرادیان مزدوج نشان می‌دهد که تعداد تکرار آنها کمتر از روش‌های فوق تخفیف نقطه‌ای است. با توجه به مقادیر جدول، به طور میانگین تعداد تکرارهای روش فوق تخفیف نقطه‌ای ۱/۶ برابر روش گرادیان مزدوج است. همچنین تعداد تکرارهای روش گرادیان دومزدوج ۱/۴ برابر روش گرادیان مزدوج پایدار شده است.

جدول ۴ زمان اجرای روش‌های فوق تخفیف قرمز-سیاه نقطه‌ای و خطی را با پردازنده مرکزی و با تعداد هسته‌های مختلف به صورت موازی نشان می‌دهد. در نیمه بالای جدول تعداد ستونها ثابت ($N_x = ۵۱۳$) در نظر گرفته شده و برای تعداد سطرهای مختلف زمان اجرا گزارش گردیده است. در این حالت برای اجرای تک هسته‌ای مشاهده می‌کنیم که برای روش فوق تخفیف نقطه‌ای با افزایش تعداد سطرها (N_y) به علت افزایش تعداد تکرارها زمان اجرا نیز به سرعت افزایش می‌یابد. شایان ذکر است با افزایش تعداد سطرها، علاوه بر افزایش حجم محاسبات در هر تکرار، تعداد تکرارها نیز افزایش می‌یابد طوری که با ۲ برابر شدن تعداد سطرها، زمان اجرا ۳ تا ۴ برابر افزایش می‌یابد. برای روش فوق تخفیف خطی تعداد تکرارها ثابت می‌ماند و البته فقط به علت افزایش تعداد سطرها، افزایش زمان اجرا با سرعت کمتری نسبت به روش نقطه‌ای رخ می‌دهد. تا پیش از $N_y = ۲۰۴۹$ زمان اجرای روش نقطه‌ای (در حالت تک هسته‌ای) با اینکه تعداد تکرارهای بیشتری دارد، کمتر از روش خطی است. اما با افزایش تعداد هسته‌ها مشاهده می‌شود که کاهش زمان اجرای بیشتری برای روش خطی در مقایسه با روش نقطه‌ای حاصل می‌گردد.

در نیمه پایین جدول تعداد سطرها ثابت ($N_y = ۵۱۳$) و تعداد ستون‌ها تغییر داده شده است. مقایسه روش نقطه‌ای در این حالت با حالت تعداد سطرهای ثابت نشان می‌دهد جابجایی تعداد سطرها با تعداد ستونها اثری بر زمان اجرا ندارد که البته به علت یکسان بودن تعداد تکرارها (تقارن رابطه ۱۱) قابل انتظار است. اما زمان اجرای روش خطی با افزایش تعداد سطرها به سرعت افزایش می‌یابد که ناشی از افزایش تعداد تکرارها است.

[DOI: 10.22034/mme.23.11.627]

جدول ۲) مشخصات پردازنده‌ها

	مدل	هسته‌ها	بسامد	حافظه	بسامد
پردازنده مرکزی	Core i۹ ۱۲۹۰۰K	۸+۱۶	۵۱۰۰ MHz	۳۲ GB	**۳۶۰۰ MHz**
پردازنده گرافیکی	GeForce RTX ۳۰۷۰ Ti	۶۱۴۴	۱۹۲۰ MHz	۸ GB	**۹۵۰۰ MHz**

جدول ۳) تعداد تکرار برای روش‌های فوق تخفیف با ضریب بهینه و روش‌های گرادیان مزدوج

گرادیان دومزدوج پایدار شده	گرایان مزدوج	فوق تخفیف خطی قرمز-سیاه	فوق تخفیف خطی	ω	فوق تخفیف نقطه‌ای قرمز-سیاه	فوق تخفیف نقطه‌ای	ω	β	$N_x \times N_y$
۲۴	۳۱	۵۱	۵۳	۱/۵۷۲۳۲	۶۵	۷۶	۱/۶۷۳۵۱	۱	**۱۷×۱۷**
۴۹	۷۱	۵۱	۵۳	۱/۵۷۲۱۳	۱۰۶	۱۱۸	۱/۷۷۹۶۵	۲	**۱۷×۳۳**
۱۰۰	۱۳۷	۵۱	۵۳	۱/۵۷۲۰۸	۱۹۳	۲۱۳	۱/۸۷۳۹۷	۴	**۱۷×۶۵**
۱۹۳	۲۷۱	۵۱	۵۳	۱/۵۷۲۰۷	۳۸۲	۴۰۵	۱/۹۳۳۴۷	۸	**۱۷×۱۲۹**
۵۰	۷۱	۹۹	۱۰۳	۱/۷۵۷۴۱	۱۰۵	۱۱۸	۱/۷۷۹۶۵	۰/۵	**۳۳×۱۷**
۵۱	۷۰	۹۹	۱۰۳	۱/۷۵۷۲۹	۱۲۹	۱۴۸	۱/۸۲۱۴۷	۱	**۳۳×۳۳**
۱۰۶	۱۴۷	۹۹	۱۰۳	۱/۷۵۷۲۵	۲۰۸	۲۳۰	۱/۸۸۳۱۶	۲	**۳۳×۶۵**
۱۹۵	۲۷۶	۹۹	۱۰۳	۱/۷۵۷۲۵	۳۸۵	۴۱۴	۱/۹۳۴۸۷	۴	**۳۳×۱۲۹**
۱۰۱	۱۳۷	۱۹۱	۱۹۹	۱/۸۷۰۴۲	۱۹۰	۲۱۳	۱/۸۷۳۹۷	۰/۲۵	**۶۵×۱۷**
۱۰۶	۱۴۷	۱۹۱	۱۹۹	۱/۸۷۰۳۵	۲۰۱	۲۳۰	۱/۸۸۳۱۶	۰/۵	**۶۵×۳۳**
۱۰۵	۱۴۴	۱۹۱	۱۹۹	۱/۸۷۰۳۳	۲۵۷	۲۸۹	۱/۹۰۶۴۵	۱	**۶۵×۶۵**
۲۱۴	۲۹۰	۱۹۱	۱۹۹	۱/۸۷۰۳۳	۴۰۷	۴۴۸	۱/۹۳۹۷۹	۲	**۶۵×۱۲۹**
۱۹۴	۲۷۱	۳۷۱	۳۸۸	۱/۹۳۲۹۸	۳۶۸	۴۰۵	۱/۹۳۳۴۷	۰/۱۲۵	**۱۲۹×۱۷**
۱۹۸	۲۷۶	۳۷۱	۳۸۷	۱/۹۳۲۹۴	۳۶۷	۴۱۴	۱/۹۳۴۸۷	۰/۲۵	**۱۲۹×۳۳**
۲۱۳	۲۹۰	۳۷۱	۳۸۷	۱/۹۳۲۹۳	۳۹۳	۴۴۸	۱/۹۳۹۷۹	۰/۵	**۱۲۹×۶۵**
۱۹۸	۲۹۱	۳۷۱	۳۸۷	۱/۹۳۲۹۳	۵۱۳	۵۶۲	۱/۹۵۲۰۹	۱	**۱۲۹×۱۲۹**

برای اینکه میزان بهره حاصل از افزایش تعداد هسته‌ها بهتر مشخص گردد در شکل ۴ افزایش سرعت نسبت به حالت تک هسته‌ای برای شبکه ۵۱۳ × ۵۱۳ نمایش داده شده است. ابتدا مشاهده می‌شود که افزایش سرعت بیشتری برای روش فوق تخفیف خطی نسبت به روش فوق تخفیف نقطه‌ای حاصل می‌شود. به طور خاص در حالت ۸ هسته‌ای مشاهده می‌شود که افزایش سرعت برای روش فوق تخفیف نقطه‌ای حدود ۲ و برای روش فوق تخفیف خطی حدود ۴ است. از سوی دیگر مشاهده می‌شود پس از ۸ هسته (برای پردازنده استفاده شده در تحقیق حاضر) افزایش سرعتی حاصل نمی‌شود.

جدول ۵ زمان اجرا برای روش‌های گرادیان مزدوج را با پردازنده مرکزی و با تعداد هسته‌های مختلف به صورت موازی نمایش می‌دهد. با اینکه تعداد تکرار روش گرادیان مزدوج حدود ۲۰ تا ۴۰ درصد بیشتر از روش گرادیان دومزدوج پایدار شده است، زمان اجرای آن (با بررسی حالت تک هسته‌ای) به علت عملیات محاسباتی کمتر، حدود ۴۰ تا ۸۰ درصد کمتر است.

شکل ۵ افزایش سرعت نسبت به حالت تک هسته‌ای را برای شبکه ۵۱۳ × ۵۱۳ نمایش می‌دهد. مشاهده می‌شود که افزایش سرعت هر دو روش گرادیان مزدوج تا حدودی برابر هستند. بر خلاف روش‌های فوق تخفیف (شکل ۴) مشاهده می‌شود با افزایش تعداد هسته‌ها افزایش سرعت ادامه می‌یابد.

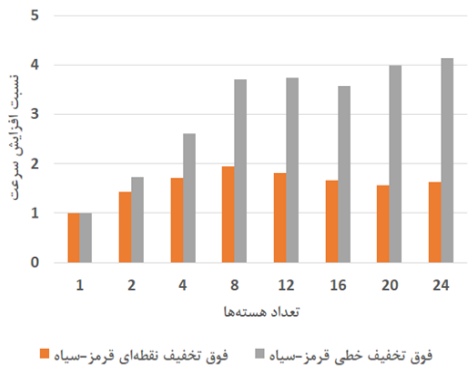
[DOI: 10.22034/mme.23.11.627]

جدول ۴) زمان اجرا (بر حسب ثانیه) برای روش‌های فوق تخفیف مختلف با ضریب بهینه

$N_x \times N_y$	β	ω	تکرار	تعداد هسته‌های فعال پردازنده مرکزی							
				۱	۲	۴	۸	۱۲	۱۶	۲۰	۲۴
$N_x = ۵۱۳$											
فوق تخفیف نقطه‌ای قرمز-سیاه											
۵۱۳×۱۲۹	۰/۲۵	۱/۹۸۳۳	۱۴۹۵	۰/۴۸	۰/۳۶	۰/۳۲	۰/۲۸	۰/۳۰	۰/۲۵	۰/۲۶	۰/۲۴
۵۱۳×۲۵۷	۰/۵	۱/۹۸۴۶	۱۶۱۶	۱/۰۳	۰/۷۳	۰/۶۰	۰/۵۶	۰/۵۶	۰/۶۹	۰/۶۸	۰/۵۷
۵۱۳×۵۱۳	۱	۱/۹۸۷۸	۲۰۲۳	۲/۵۶	۱/۸۰	۱/۴۹	۱/۳۱	۱/۴۱	۱/۵۴	۱/۶۴	۱/۵۷
۵۱۳×۱۰۲۵	۲	۱/۹۹۲۲۷	۳۱۳۸	۸/۱۴	۵/۶۴	۴/۶۰	۴/۱۵	۴/۲۶	۴/۸۸	۵/۲۹	۴/۱۶
۵۱۳×۲۰۴۹	۴	۱/۹۹۵۸	۵۶۳۲	۲۹/۹۶	۲۱/۶۲	۱۷/۰۷	۱۵/۱۵	۱۶/۱۸	۱۸/۴۰	۱۹/۵۵	۱۶/۷۱
فوق تخفیف خطی قرمز-سیاه											
۵۱۳×۱۲۹	۰/۲۵	۱/۹۸۲۷۹	۱۴۵۲	۱/۳۵	۰/۸۴	۰/۵۹	۰/۴۴	۰/۳۹	۰/۳۹	۰/۳۳	۰/۳۴
۵۱۳×۲۵۷	۰/۵	۱/۹۸۲۷۹	۱۴۵۲	۲/۷۰	۱/۶۳	۱/۰۷	۰/۷۸	۰/۷۳	۰/۷۴	۰/۷۰	۰/۶۶
۵۱۳×۵۱۳	۱	۱/۹۸۲۷۹	۱۴۵۲	۵/۳۹	۳/۱۳	۲/۰۶	۱/۴۶	۱/۴۴	۱/۵۱	۱/۳۵	۱/۳۰
۵۱۳×۱۰۲۵	۲	۱/۹۸۲۷۹	۱۴۵۲	۱۰/۵۳	۶/۰۴	۳/۹۸	۲/۸۴	۲/۸۰	۲/۸۹	۲/۹۵	۲/۳۶
۵۱۳×۲۰۴۹	۴	۱/۹۸۲۷۹	۱۴۵۲	۲۲/۰۷	۱۲/۵۶	۸/۲۲	۵/۹۲	۵/۶۲	۵/۷۸	۵/۹۴	۴/۹۵
$N_y = ۵۱۳$											
فوق تخفیف نقطه‌ای قرمز-سیاه											
۱۲۹×۵۱۳	۴	۱/۹۸۳۳	۱۴۹۵	۰/۴۸	۰/۳۴	۰/۳۴	۰/۲۸	۰/۲۸	۰/۲۶	۰/۲۵	۰/۲۷
۲۵۷×۵۱۳	۲	۱/۹۸۴۶	۱۶۱۶	۱/۰۳	۰/۷۳	۰/۶۱	۰/۶۰	۰/۶۱	۰/۶۲	۰/۶۳	۰/۵۳
۵۱۳×۵۱۳	۱	۱/۹۸۷۸	۲۰۲۳	۲/۵۷	۱/۷۸	۱/۴۸	۱/۳۷	۱/۴۳	۱/۵۶	۱/۵۷	۱/۳۳
۱۰۲۵×۵۱۳	۰/۵	۱/۹۹۲۲۷	۳۱۳۸	۹/۴۱	۶/۹۴	۵/۲۸	۴/۶۶	۴/۷۵	۵/۰۱	۵/۱۲	۴/۴۳
۲۰۴۹×۵۱۳	۰/۲۵	۱/۹۹۵۸	۵۶۳۲	۳۰/۱۳	۲۰/۹۶	۱۷/۰۸	۱۵/۶۱	۱۶/۰۶	۱۸/۵۷	۱۹/۴۳	۱۶/۰۸
فوق تخفیف خطی قرمز-سیاه											
۱۲۹×۵۱۳	۴	۱/۹۳۲۹۳	۳۸۵	۰/۳۶	۰/۲۲	۰/۱۶	۰/۱۱	۰/۱۰	۰/۰۹	۰/۱۰	۰/۰۸
۲۵۷×۵۱۳	۲	۱/۹۶۵۸۸	۷۴۷	۱/۳۷	۰/۷۹	۰/۵۳	۰/۴۰	۰/۳۷	۰/۴۰	۰/۳۳	۰/۳۴
۵۱۳×۵۱۳	۱	۱/۹۸۲۷۹	۱۴۵۲	۵/۳۸	۳/۱۳	۲/۰۷	۱/۴۶	۱/۴۷	۱/۴۲	۱/۳۶	۱/۱۷
۱۰۲۵×۵۱۳	۰/۵	۱/۹۹۱۳۶	۲۸۲۰	۲۱/۰۰	۱۲/۱۶	۷/۹۷	۵/۷۵	۵/۵۸	۵/۴۸	۵/۶۳	۴/۹۳
۲۰۴۹×۵۱۳	۰/۲۵	۱/۹۹۵۶۷	۵۴۷۱	۸۳/۸۲	۴۷/۹۵	۳۱/۳۴	۲۲/۵۰	۲۱/۴۱	۲۲/۱۵	۲۲/۳۶	۱۸/۶۰

به طور خاص برای حالت ۸ هسته‌ای، نسبت افزایش سرعت حدود ۶ و برای حالت ۲۰ هسته‌ای حدود ۸ است.

در جدول ۶ زمان اجرای روش‌های مختلف با استفاده از پردازنده گرافیکی آورده شده است. پیاده‌سازی روش فوق تخفیف نقطه‌ای روی پردازنده گرافیکی به سهولت انجام می‌شود. اما برای روش فوق تخفیف خطی، نیاز به حل دستگاه معادلات سه‌قطری به ازای هر ستون است. برای این کار از کتابخانه کواسپارس (cuSPARSE) استفاده می‌شود. می‌توان همانند پردازنده مرکزی حل دستگاه‌ها را بین هسته‌های پردازنده گرافیکی توزیع نمود و سپس با استفاده از الگوریتم توماس هر دستگاه را حل کرد.

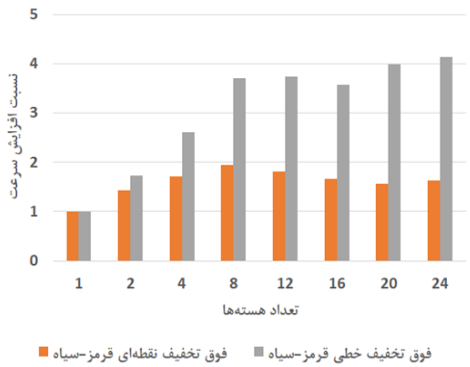


شکل ۴) نسبت افزایش سرعت برای روش‌های فوق تخفیف برای شبکه ۵۱۳ × ۵۱۳

[DOI: 10.22034/mme.23.11.627]

جدول ۵) زمان اجرا برای روش‌های گرادیان مزدوج

$N_x \times N_y$	β	تکرار	تعداد هسته‌های فعال پردازنده مرکزی: ۱	۲	۴	۸	۱۲	۱۶	۲۰	۲۴
$N_x = ۵۱۳$										
گرادیان مزدوج										
۵۱۳×۱۲۹	۰/۲۵	۸۵۷	۰/۲۵	۰/۱۳	۰/۰۷	۰/۰۵	۰/۰۴	۰/۰۴	۰/۰۳	۰/۰۳
۵۱۳×۲۵۷	۰/۵	۸۹۵	۰/۵۳	۰/۲۷	۰/۱۵	۰/۰۸	۰/۰۹	۰/۰۷	۰/۰۶	۰/۰۵
۵۱۳×۵۱۳	۱	۹۳۶	۱/۰۶	۰/۵۷	۰/۳۱	۰/۱۷	۰/۱۸	۰/۱۴	۰/۱۲	۰/۱۱
۵۱۳×۱۰۲۵	۲	۱۷۸۸	۴/۲۲	۲/۱۸	۱/۱۸	۰/۶۵	۰/۸۳	۰/۷۵	۰/۶۵	۰/۵۵
۵۱۳×۲۰۴۹	۴	۳۴۵۰	۱۹/۰۸	۹/۸۲	۵/۲۴	۲/۸۱	۳/۵۵	۲/۹۸	۲/۴۴	۲/۱۲
گرادیان دومزدوج پایدار شده										
۵۱۳×۱۲۹	۰/۲۵	۶۸۶	۰/۳۸	۰/۲۲	۰/۱۲	۰/۰۷	۰/۰۷	۰/۰۶	۰/۰۵	۰/۰۴
۵۱۳×۲۵۷	۰/۵	۷۴۱	۰/۸۸	۰/۵۳	۰/۲۵	۰/۱۵	۰/۱۴	۰/۱۲	۰/۱۱	۰/۰۹
۵۱۳×۵۱۳	۱	۶۵۸	۱/۴۴	۰/۷۳	۰/۴۱	۰/۲۵	۰/۲۸	۰/۲۶	۰/۱۸	۰/۱۸
۵۱۳×۱۰۲۵	۲	۱۳۸۹	۶/۴۵	۳/۶۸	۲/۳۸	۱/۰۱	۱/۴۶	۱/۲۱	۰/۸۹	۰/۸۳
۵۱۳×۲۰۴۹	۴	۳۱۸۴	۳۳/۷۵	۱۷/۸۰	۱۱/۸۲	۶/۹۷	۸/۰۴	۶/۳۱	۶/۹۰	۷/۹۳
$N_y = ۵۱۳$										
گرادیان مزدوج										
۱۲۹×۵۱۳	۴	۹۰۷	۰/۲۶	۰/۱۴	۰/۰۸	۰/۰۶	۰/۰۵	۰/۰۴	۰/۰۳	۰/۰۳
۲۵۷×۵۱۳	۲	۹۵۱	۰/۵۴	۰/۳۱	۰/۱۵	۰/۰۹	۰/۰۹	۰/۰۷	۰/۰۶	۰/۰۶
۵۱۳×۵۱۳	۱	۹۳۶	۱/۰۷	۰/۵۸	۰/۳۱	۰/۱۶	۰/۱۷	۰/۱۵	۰/۱۲	۰/۱۱
۱۰۲۵×۵۱۳	۰/۵	۱۷۳۲	۳/۹۴	۲/۰۸	۱/۰۶	۰/۵۹	۰/۸۲	۰/۷۲	۰/۶۲	۰/۵۳
۲۰۴۹×۵۱۳	۰/۲۵	۳۱۵۲	۱۵/۶۷	۸/۰۴	۴/۲۸	۲/۳۹	۳/۰۳	۲/۶۶	۲/۲۰	۱/۹۰
گرادیان دومزدوج پایدار شده										
۱۲۹×۵۱۳	۴	۷۷۹	۰/۴۳	۰/۲۰	۰/۱۲	۰/۰۷	۰/۰۷	۰/۰۶	۰/۰۵	۰/۰۴
۲۵۷×۵۱۳	۲	۷۱۴	۰/۷۹	۰/۴۰	۰/۲۴	۰/۱۳	۰/۱۳	۰/۱۲	۰/۱۰	۰/۱۰
۵۱۳×۵۱۳	۱	۶۵۸	۱/۴۴	۰/۷۷	۰/۴۴	۰/۲۵	۰/۲۷	۰/۲۸	۰/۱۷	۰/۱۸
۱۰۲۵×۵۱۳	۰/۵	۱۴۵۴	۶/۸۴	۴/۱۲	۱/۹۲	۱/۰۶	۱/۲۸	۱/۱۱	۰/۸۱	۰/۷۸
۲۰۴۹×۵۱۳	۰/۲۵	۲۷۷۱	۳۰/۰۲	۲۰/۱۴	۱۰/۸۴	۵/۷۸	۷/۵۰	۵/۳۳	۴/۷۵	۴/۷۶



شکل ۵) نسبت افزایش سرعت برای روش‌های گرادیان مزدوج برای شبکه ۵۱۳ × ۵۱۳

. اما با توجه به تعداد هسته‌های زیاد پردازنده گرافیکی، بهتر است از الگوریتم کاهش چرخشی موازی (Parallel Cyclic Reduction) استفاده شود. برای این کار، تمام دستگاه‌های سه‌قطری در یک دستگاه سه‌قطری بزرگ جایگذاری و سپس حل می‌شود. نتایج ارائه شده در جدول ۶ با استفاده از این روش است.

مهم‌ترین بخش‌های پیاده‌سازی روش‌های گرادیان مزدوج روی پردازنده گرافیکی، بخش عملیات ضرب ماتریسی و سپس عملیات ضرب داخلی است. پیاده‌سازی عملیات ضرب ماتریسی (۲۶) به سهولت انجام می‌شود.

[DOI: 10.22034/mme.23.11.627]

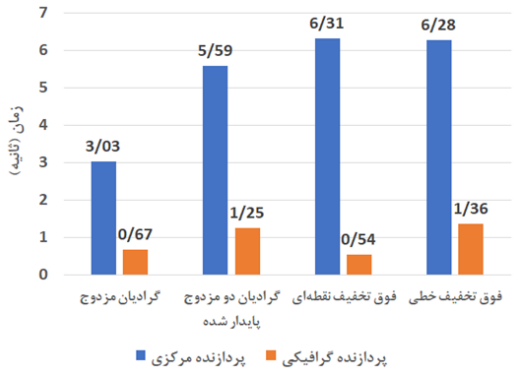
جدول ۶) زمان اجرا برای روش‌های مختلف با پردازنده گرافیکی

گرادیان دومزدوج پایدار شده		گرادیان مزدوج		فوق تخفیف خطی قرمز-سیاه		فوق تخفیف نقطه‌ای قرمز-سیاه			
زمان	تکرار	زمان	تکرار	زمان	تکرار	زمان	تکرار	β	$N_x \times N_y$
$N_x = ۵۱۳$									
۰/۱۶۳	۶۸۶	۰/۰۸۲	۸۵۷	۰/۶۱۵	۱۴۵۲	۰/۲۹	۱۴۹۵	۰/۲۵	۵۱۳×۱۲۹
۰/۱۸۷	۷۴۱	۰/۱۰۶	۸۹۵	۰/۶۴۳	۱۴۵۲	۰/۲۹۵	۱۶۱۶	۰/۵	۵۱۳×۲۵۷
۰/۲۳۹	۶۵۸	۰/۱۵۹	۹۳۶	۰/۸۴۷	۱۴۵۲	۰/۸۴۲	۲۰۲۳	۱	۵۱۳×۵۱۳
۰/۸۶۴	۱۳۸۹	۰/۴۶۲	۱۷۸۸	۱/۱۳۴	۱۴۵۲	۱/۵۷۲	۳۱۳۸	۲	۵۱۳×۱۰۲۵
۲/۶۶۳	۳۱۸۴	۱/۵۰۵	۳۴۵۰	۱/۷۰۴	۱۴۵۲	۳/۴۷۱	۵۶۳۲	۴	۵۱۳×۲۰۴۹
$N_y = ۵۱۳$									
۰/۱۶۷	۷۷۹	۰/۰۹۱	۹۰۷	۰/۱۶۴	۳۸۵	۰/۲۹	۱۴۹۵	۴	۱۲۹×۵۱۳
۰/۱۹۹	۷۱۴	۰/۱۱۳	۹۵۱	۰/۳۴۲	۷۴۷	۰/۳۲۴	۱۶۱۶	۲	۲۵۷×۵۱۳
۰/۲۳۱	۶۵۸	۰/۱۶۲	۹۳۶	۰/۸۳۹	۱۴۵۲	۰/۸۳	۲۰۲۳	۱	۵۱۳×۵۱۳
۰/۹۰۸	۱۴۵۴	۰/۴۴۱	۱۷۳۲	۲/۱۵	۲۸۲۰	۱/۴۹۲	۳۱۳۸	۰/۵	۱۰۲۵×۵۱۳
۲/۳۱	۲۷۷۱	۱/۳۷۶	۳۱۵۲	۶/۶۴۱	۵۴۷۱	۳/۵۳۲	۵۶۳۲	۰/۲۵	۲۰۴۹×۵۱۳

برای پیاده‌سازی عملیات ضرب داخلی، می‌توان از کتابخانه تراست (Thrust) استفاده کرد. اما از آنجا که در هر بار استفاده از تابع ضرب داخلی این کتابخانه، حافظه مورد نیاز تخصیص و حذف می‌گردد، سربار اجرایی بسیار زیادی ایجاد می‌شود. به همین علت تابع ضرب داخلی جداگانه پیاده‌سازی شده است، طوری که فقط یک بار در ابتدای اجرای برنامه حافظه مورد نیاز تخصیص می‌یابد و در انتهای برنامه آزاد می‌شود. به این ترتیب زمان اجرای روش‌های گرادیان مزدوج به یک سوم کاهش می‌یابد.

در مقایسه با جدول ۴ و جدول ۵ مشاهده می‌شود برای شبکه‌های درشت نه تنها زمان اجرا کاهش نیافته، بلکه افزایش نیز یافته است. اما برای شبکه‌های ریز افزایش سرعت ۲ تا ۳ برابر حاصل شده است. علت این امر کم بودن عملیات محاسباتی است که برای شبکه‌های درشت زمان انجام محاسبات در مقایسه با زمان سربار حاصل از اجرای تابع روی پردازنده گرافیکی، کمتر است.

از آنجا که برای شبکه‌های جدول ۶ افزایش سرعت برای شبکه‌های ریز مشاهده شده است، به همین علت برای شبکه بسیار ریز ۴۰۹۷ × ۴۰۹۷ در شکل ۶ مقایسه‌ای بین زمان اجرای پردازنده مرکزی با ۸ هسته و زمان اجرای پردازنده گرافیکی برای روش‌های مختلف برای ۱۰۰ تکرار انجام شده است. مشاهده می‌شود برای روش‌های گرادیان مزدوج افزایش سرعت حاصل حدود ۴ برابر است. همچنین افزایش سرعت حاصل برای



شکل ۶) مقایسه زمان اجرای روش‌های مختلف برای ۱۰۰ تکرار برای شبکه ۴۰۹۷ × ۴۰۹۷

روش‌های فوق تخفیف نقطه‌ای و خطی به ترتیب حدود ۱۲ و ۵ برابر است. به عبارت دیگر برای روش فوق تخفیف نقطه‌ای نسبت به سایر روش‌ها افزایش سرعت بیشتری حاصل می‌شود.

جدول ۷ زمان اجرای روش تکراری مختلف را برای معادله سه بُعدی نمایش می‌دهد. در اینجا تعداد نقاط شبکه در هر سه راستا را یکسان در نظر گرفته‌ایم. نتایج برای هر دو پردازنده مرکزی و پردازنده گرافیکی گزارش شده است. مقایسه روش فوق تخفیف نقطه‌ای و خطی نشان می‌دهد با اینکه روش فوق خطی در تعداد تکرار کمتری همگرا می‌گردد، در حالت تک هسته‌ای زمان اجرای آن حدود ۱/۵ برابر بیشتر از روش فوق تخفیف نقطه‌ای است.

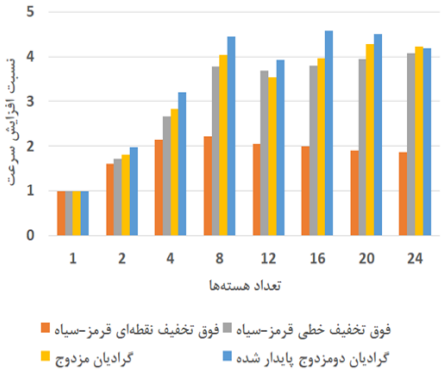
[DOI: 10.22034/mme.23.11.627]

جدول ۷) زمان اجرا برای روش‌های تکرار مختلف برای معادله سه بُعدی

تعداد هسته‌های فعال پردازنده مرکزی											
N^3	ω	تکرار	۱	۲	۴	۸	۱۲	۱۶	۲۰	۲۴	GPU
فوق تخفیف نقطه‌ای قرمز-سیاه											
65^3	۱/۹۰۶۴۵	۲۶۲	۰/۵۹	۰/۳۸	۰/۲۸	۰/۲۲	۰/۲۵	۰/۲۹	۰/۲۵	۰/۲۱	۰/۱۳
129^3	۱/۹۵۲۰۹	۵۰۸	۹/۴۷	۵/۸۷	۴/۳۶	۳/۶۱	۴/۲۸	۴/۳۴	۴/۲۹	۴/۳۳	۰/۶۴
257^3	۱/۹۷۵۷۵	۹۸۴	۱۴۸/۲	۹۲/۵۳	۶۹/۰۸	۶۶/۸۱	۷۲/۰۵	۷۴/۰۹	۷۸/۰۸	۷۹/۴۹	۶/۸۶
فوق تخفیف خطی قرمز-سیاه											
65^3	۱/۸۸۶۶۷	۲۱۶	۰/۸۷	۰/۶۳	۰/۴۴	۰/۳۱	۰/۳۲	۰/۲۷	۰/۳۰	۰/۳۰	۰/۱۶
129^3	۱/۹۴۱۶۵	۴۱۸	۱۴/۲۱	۸/۵۳	۵/۷۵	۴/۰۶	۴/۱۲	۳/۹۲	۳/۸۷	۳/۷۵	۱/۰۱
257^3	۱/۹۷۰۳۹	۸۱۱	۲۲۶/۴	۱۳۱/۳	۸۴/۹۷	۶۰/۰۵	۶۱/۳۲	۵۹/۷۱	۵۷/۳۸	۵۵/۶۷	۱۲/۴۳
گرادیان مزدوج											
65^3	-	۱۶۶	۰/۳۶	۰/۲۰	۰/۱۲	۰/۰۷	۰/۰۶	۰/۰۵	۰/۰۴	۰/۰۴	۰/۰۳
129^3	-	۳۳۰	۶/۲۶	۳/۳۳	۲/۰۱	۱/۳۰	۱/۴۳	۱/۲۳	۱/۱۰	۱/۱۵	۰/۳۰
257^3	-	۶۶۳	۱۰۱/۹	۵۶/۲۹	۳۶/۰۰	۲۵/۲۵	۲۸/۸۱	۲۵/۷۹	۲۳/۷۹	۲۴/۱۸	۴/۳۸
گرادیان دومزدوج پایدار شده											
65^3	-	۹۹	۰/۴۲	۰/۲۳	۰/۱۴	۰/۰۸	۰/۰۷	۰/۰۶	۰/۰۵	۰/۰۵	۰/۰۴
129^3	-	۲۱۰	۷/۹۹	۴/۵۴	۲/۴۷	۱/۵۹	۱/۸۳	۱/۵۷	۱/۴۳	۱/۵۲	۰/۳۶
257^3	-	۴۵۷	۱۴۰/۴	۷۱/۲۸	۴۳/۸۸	۳۱/۵۲	۳۵/۷۱	۳۰/۶۵	۳۱/۲۰	۳۳/۵۰	۵/۷۲

اما با افزایش تعداد هسته‌ها زمان اجرای آن با نسبت بیشتری کاهش می‌یابد و در نتیجه سریع‌تر از روش فوق تخفیف نقطه‌ای اجرا می‌گردد. مقایسه همین دو روش برای پردازنده گرافیکی نشان می‌دهد که زمان اجرای روش نقطه‌ای حدود نصف زمان اجرای روش خطی است.

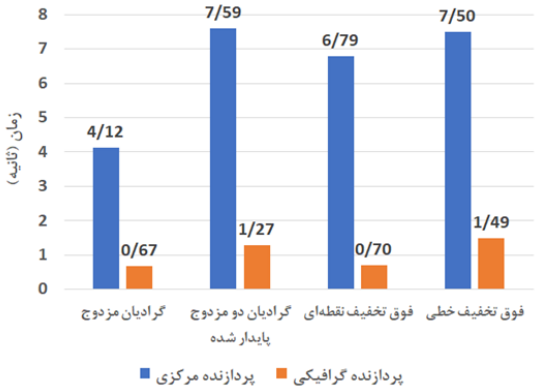
شکل ۷ افزایش سرعت نسبت به حالت تک هسته‌ای را برای شبکه 257^3 نمایش می‌دهد. مشاهده می‌شود که پس از ۸ هسته، افزایش سرعتی حاصل نمی‌شود. برای روش فوق تخفیف نقطه‌ای، همانند نتایج معادله دو بُعدی (شکل ۴) افزایش سرعت حاصل کمی بیش از ۲ برابر و برای روش فوق تخفیف خطی حدود ۴ برابر است.



شکل ۷) نسبت افزایش سرعت برای روش‌های مختلف برای شبکه 257^3

برای روش‌های گرادیان مزدوج نیز افزایش سرعت در بهترین حالت به ۴/۵ برابر رسیده است که البته در مقایسه با نتایج معادله دو بُعدی (شکل ۵) کمتر است.

در شکل ۸ مقایسه‌ای بین زمان اجرای پردازنده مرکزی با ۸ هسته و زمان اجرای پردازنده گرافیکی برای روش‌های مختلف برای ۱۰۰ تکرار انجام شده است. مشاهده می‌شود برای روش‌های گرادیان مزدوج افزایش سرعت حاصل حدود ۶ برابر است. همچنین افزایش سرعت حاصل برای روش‌های فوق تخفیف نقطه‌ای و خطی به ترتیب حدود ۱۰ و ۵ برابر است.



شکل ۸) مقایسه زمان اجرای روش‌های مختلف برای ۱۰۰ تکرار برای شبکه 257^3

[DOI: 10.22034/mme.23.11.627]

۵- نتیجه‌گیری

در تحقیق حاضر یک بررسی بر موازی‌سازی چند حلگر تکراری دستگاه معادلات خطی حاصل از گسسته‌سازی معادله پواسون دو بُعدی و سه بُعدی به روش تفاضل محدود انجام شد. به طور خاص روش‌های تکراری فوق تخفیف گاوس سایدل نقطه‌ای و خطی و همچنین روش‌های گرادیان مزدوج و گرادیان دومزدوج پایدار شده بررسی گردید. نتایج نشان داد، موازی‌سازی با پردازنده مرکزی چند هسته‌ای برای روش فوق تخفیف نقطه‌ای افزایش سرعت محسوسی ایجاد نمی‌کند، در حالی که برای سایر روش‌ها افزایش سرعت حدود ۴ برابر حاصل شد. این افزایش سرعت برای دو روش گرادیان مزدوج برای معادله دو بُعدی به ۱۰ نیز رسید. نتایج موازی‌سازی با پردازنده گرافیکی نشان داد که برای معادله دو بُعدی و شبکه‌های درشت افزایش سرعت محسوسی حاصل نمی‌شود. اما برای شبکه‌های ریز افزایش سرعت ۴ الی ۵ برابر نسبت به پردازنده مرکزی در حالت ۸ هسته‌ای مشاهده گردید. برای معادله سه بُعدی نیز افزایش سرعت متناظر در شبکه‌های متوسط و ریز حدود ۶ برابر بود. بیشترین افزایش سرعت با پردازنده گرافیکی نیز برای روش فوق تخفیف نقطه‌ای برای هر دو معادله دو بُعدی و سه بُعدی مشاهده گردید که حدود ۱۰ الی ۱۲ برابر بود.

در مجموع می‌توان گفت روش‌های گرادیان مزدوج در بیشتر موارد به علت تعداد تکرار کمتر زمان کمتری برای همگرایی نیاز دارند و در نتیجه کمترین زمان اجرا را دارند. از سوی دیگر روش فوق تخفیف خطی با ضریب بهینه در شبکه‌هایی که تعداد دستگاه معادلات سه‌قطری (تعداد خط‌ها) از تعداد مجهولات هر دستگاه (خط) مقدار قابل توجهی کوچک‌تر باشد، در کمترین تکرار همگرا می‌گردد و به همین علت در این موارد سریع‌ترین زمان اجرا را دارد. روش فوق تخفیف نقطه‌ای با ضریب بهینه، هر چند برای همگرایی تعداد تکرار بیشتری نیاز دارد، اما به علت عملیات محاسباتی کم، زمان اجرای چندان بیشتری نسبت به سایر روش‌ها ندارد. این مطلب خصوصاً در استفاده از پردازنده گرافیکی مشاهده می‌گردد که در کنار پیاده‌سازی راحت آن می‌تواند موجب گردد که انتخاب اول در توسعه کُدهای محاسباتی باشد.

تاییدیه اخلاقی: این مقاله تاکنون در نشریه دیگری به چاپ نرسیده و همچنین برای بررسی یا چاپ به نشریه دیگری فرستاده نشده است.

تعارض منافع: مقاله حاضر هیچگونه تعارض منافعی با سازمان‌ها و اشخاص دیگر ندارد.

تقدیر و تشکر: از حمایت مالی دانشگاه تهران از این تحقیق در قالب طرح پژوهشی شماره ۲۸۷۴۵/۱/۰۳ قدردانی می‌گردد.

منابع

1- D. M. Young, “Iterative Solution of Large Linear Systems,” Academic Press, New York, 1971.
2- Saad Y. Iterative methods for sparse linear systems. Society for Industrial and Applied Mathematics; 2003 Jan 1.
3- Świrydowicz K, Darve E, Jones W, Maack J, Regev S, Saunders MA, Thomas SJ, Peleš S. Linear solvers for power grid optimization problems: a review of GPU-accelerated linear solvers. Parallel Computing. 2022 Jul 1;111:102870.
4- Cotronis Y, Konstantinidis E, Louka MA, Missirlis NM. A comparison of CPU and GPU implementations for solving the Convection Diffusion equation using the local Modified SOR method. Parallel Computing. 2014 Jul 1;40(7):173-85.
5- Taghavi SM, Akbarzadeh P, Mahmoodi Darian H. SADI approach programming on GPU: convective heat transfer of nanofluids flow inside a wavy channel. Journal of Thermal Analysis and Calorimetry. 2021 Oct;146:31-46.
6- Foadaddini A, Zolfaghari SA, Mahmoodi Darian H, Saadatfar H. An efficient GPU-based fractional-step domain decomposition scheme for the reaction–diffusion equation. Computational and Applied Mathematics. 2020 Dec;39:1-35.
7- Souri M, Akbarzadeh P, Darian HM. Parallel Thomas approach development for solving tridiagonal systems in GPU programming– steady and unsteady flow simulation. Mechanics & Industry. 2020;21(3):303.
8- Darian HM, Esfahanian V. Assessment of WENO schemes for multi-dimensional Euler equations using GPU. International Journal for Numerical Methods in Fluids. 2014 Dec 30;76(12):961-81.
9- Mahmoodi Darian H. Accelerating high-order WENO schemes using two heterogeneous GPUs. Journal of Computational Applied Mechanics. 2017 Dec 1;48(2):161-70.
10- Esfahanian V, Darian HM, Gohari SI. Assessment of WENO schemes for numerical simulation of some hyperbolic equations using GPU. Computers & Fluids. 2013 Jul 10;80:260-8.8
11- Baghapour B, Esfahanian V, Torabzadeh M, Mahmoodi Darian H. A discontinuous Galerkin method with block cyclic reduction solver for simulating compressible flows on GPUs. International journal of computer mathematics. 2015 Jan 2;92(1):110-31.
12- Di P, Wu H, Xue J, Wang F, Yang C. Parallelizing SOR for GPGPUs using alternate loop tiling. Parallel Computing. 2012 Jun 1;38(6-7):310-28.
13- Bozorgmehr B, Willemsen P, Gibbs JA, Stoll R, Kim JJ, Pardyjak ER. Utilizing dynamic parallelism in CUDA to accelerate a 3D red-black successive over relaxation wind-field solver. Environmental Modelling & Software. 2021 Mar 1;137:104958.
14- Mossaiby F. High Performance Implementation of Conjugate Gradient Method Using OpenCL on Graphics Processing Units. Journal of Computational Methods in Engineering. 1401; 33(1): 1-13.

[DOI: 10.22034/mme.23.11.627]

15- Helfenstein R, Koko J. Parallel preconditioned conjugate gradient algorithm on GPU. Journal of Computational and Applied Mathematics. 2012 Sep 1;236(15):3584-90.
16- Mittal S. A study of successive over-relaxation method parallelisation over modern HPC languages. International journal of high performance computing and networking. 2014 Jan 1;7(4):292-8.
17- Zapata MU, Van Bang DP, Nguyen KD. Parallel simulations for a 2D x/z two-phase flow fluid-solid particle model. Computers & Fluids. 2018 Sep 15;173:103-10.
18- Xie D. A new block parallel SOR method and its analysis. SIAM Journal on Scientific Computing. 2006;27(5):1513-33.
19- Hoffmann KA, Chiang ST. Computational fluid dynamics volume I. Engineering education system. 2000 Aug.
20- Kuo CC, Chan TF. Two-color Fourier analysis of iterative algorithms for elliptic problems with red/black ordering. SIAM Journal on Scientific and Statistical Computing. 1990 Jul;11(4):767-93.
21- LeVeque RJ, Trefethen LN. Fourier analysis of the SOR iteration. IMA journal of numerical analysis. 1988 Jul 1;8(3):273-9.
22- Yang S, Gobbert MK. The optimal relaxation parameter for the SOR method applied to the Poisson equation in any space dimensions. Applied mathematics letters. 2009 Mar 1;22(3):325-31.

[DOI: 10.22034/mme.23.11.627]