مجلة جامعة تشرين للبحوث والدراسات العلمية – سلسلة العلوم الأساسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Basic Sciences Series Vol. (30) No. (1) 2008

خوارزميات مقترحة من نوع تحليل LU لحل مسائل البرمجة الخطية كثيرة الأصفار

الدكتور أحمد الكردي*
ختام حوى**

(تاريخ الإيداع **22 / 10 / 2007**. قُبِل للنشر في **2007/12/26**)

□ الملخّص □

يعتبر حل مسائل البرمجة الخطية باستخدام طرائق تحليل LU كثيرة الأصفار هي مسألة مفتوحة، مما يجعلها مجالا مهما للبحث و التطوير. لذلك نكرس اهتمامنا في هذه المقالة على تطوير خوارزميات من نوع تحليل LU كثيرة الأصفار عند تنفيذ خوارزمية سيمبليكس المعدلة. نصف في هذه المقالة خوارزميتين فعالتين تعتمدان على تحليل LU لحل مسألة البرمجة الخطية كثيرة الأصفار. نجري العديد من تجارب المحاكاة الحاسوبية لتوضيح فعالية هاتين الخوارزميتين. نقارن النتائج الحاصلة مع تلك الناتجة من تطبيق طريقة Golub – Bartels وطريقة Golub – Bartelsكثيرة الأصفار و طريقة Forrest – Tomlin و طريقة Reid لتبيان فعالية الخوارزميات المطورة. نبين من التجارب العددية أن الخوارزميتين المقترحتين هما الأفضل بين الخوارزميات المدروسة ولذلك نوصي باستخدامهما في المكتبات البرمجية الجاهزة كبديل عن الطرائق الحالية لحل مسائل البرمجة الخطية كثيرة الأصفار.

كلمات مفتاحية:
خوارزمية سيمبليكس المعدلة، تحليل LU كثيرة الأصفار، مسألة برمجة خطية كثيرة الأصفار، طريقة Golub – Bartels كثيرة الأصفار.

* مدرس – قسم الرياضيات – كلية العلوم – جامعة البعث – حمص – سورية.
** طالبة ماجستير – قسم الرياضيات – كلية العلوم – جامعة البعث – حمص – سورية.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الأساسية المجلد (30) العدد (1) 2008
Tishreen University Journal for Research and Scientific Studies - Basic Sciences Series Vol. (30) No. (1) 2008

Proposed LU-Decomposition - Based Algorithms for Sparse Linear Programming Problems

Dr.Ahmad Al-Kurdi [*]
Khitam Hawa [**]

(Received 22 / 10 / 2007. Accepted 26/12/2007)

□ ABSTRACT □

The solution of linear programming problems by sparse LU decomposition is an open problem providing room for further research. This paper tries to develop sparse LU decomposition – based algorithms when implementing Simplex algorithm.

In this paper, we describe efficient algorithms which are based on LU decomposition for solving sparse linear programming problems. Many numerical simulations are carried out to illustrate the efficiency of the proposed algorithms. We compare the obtained results with Golub – Bartels method, Sparse Golub – Bartels method, Forrest – Tomlin method and Reid method to show the efficiency of the proposed algorithms. From the numerical experiments carried out, it is shown that the proposed algorithms are much better than that of available methods. So we recommend using this method in software packages as new alternative for solving the sparse linear programming problems.

Key Words:
Revised Simplex Algorithm, Sparse LU-Decomposition, Sparse Linear Programming Problem, Sparse Golub–Bartels method.

[*] **Assistant Professor, Department of Mathematics, Faculty of Sciences, Al-Baath University, Homs, Syria.**
[**] **Postgraduate Student, Department of Mathematics, Faculty of Sciences, Al-Baath University, Homs, Syria.**

1. مقدمة (Introduction):

تعد مسألة البرمجة الخطية واحدة من مسائل عديدة تستخدم مزايا تقنيات اللاصفرية (**Sparsity**) في كل خطوة تكرار. تعتبر مسائل البرمجة الخطية بصفة عامة من مسائل الأمثليات التي تعرف بعبارة خطية تخضع لعدد من القيود الخطية. يتطلب حل مسائل البرمجة الخطية بطريقة سيمبليكس المعدلة التعبير عن المسألة وفق الشكل القياسي. إلا أن مسائل البرمجة الخطية لا تأتي جميعها في الشكل القياسي، إذ غالباً ما يُعبر عن القيود بمتراجحات (متباينات) بدلا من المعادلات. كما أن متغيرات القرار، في بعض المسائل، قد لا تكون جميعها غير سالبة. لذا فالخطوة الأولى في حل النموذج الخطي هي تحويلها إلى مسألة ذات شكل قياسي. إذا تحويل المتراجحات إلى معادلات يتم بإضافة متحولات جديدة لتمثل الفرق بين الطرفين الأيسر و الأيمن لكل متراجحة.

معظم مسائل البرمجة الخطية هي كثيرة الأصفار أي أن نسبة العناصر اللاصفرية لمصفوفة الأمثال صغيرة جدا. تعتمد الأهمية العلمية و النجاح التجاري لخوارزمية سيمبليكس المعدلة على استثمار البنية الصفرية. مع التكنولوجيا الحالية يمكن حل مسائل البرمجة الخطية كثيرة الأصفار بملايين المتغيرات والقيود باستخدام طريقة سيمبليكس المعدلة. من جهة أخرى، تصبح مسائل البرمجة الخطية الكثيفة غير جذابة إذا تجاوز عدد المتغيرات 10000. سنناقش في هذه المقالة جانباً هاماً من خوارزمية سيمبليكس المعدلة وهو تحليل LU كثيرة الأصفار. إن حل مسائل البرمجة الخطية كثيرة الأصفار يكون ممكنا باستخدام طرائق تحليل LU كثيرة الأصفار. إن تنفيذ البنية الصفرية لتحليل LU خلال تنفيذ خوارزمية سيمبليكس المعدلة هو مسألة صعبة تتطلب خبرة كبيرة. وإن تحليل LU كثيرة الأصفار هو جزء هام في تنفيذات خوارزمية سيمبليكس المعدلة. ونظراً لأن نسبة كبيرة من مسائل البرمجة الخطية كثيرة الأصفار تحل بطرائق أخرى، وفإن هذه المسألة لا تزال مفتوحة. نكرس اهتمامنا في هذه المقالة على تطوير خوارزميات LU كثيرة الأصفار عند تنفيذ خوارزمية سيمبليكس المعدلة.

2. أهمية البحث و مواده:

تعتبر مسألة البرمجة الخطية جزءا مركزيا لحقل بحوث العمليات نظرا لتطبيقاتها الكبيرة التي بدأت بالظهور منذ الحرب العالمية الثانية في مجالات إدارة المؤن و عمليات التصنيع و صولا إلى نظرية التحكم و بشكل خاص في نظرية إدارة العمليات و غيرها. و يمكن تعريف البرمجة الخطية بأنها عبارة عن طريق أو أسلوب رياضي يستخدم للمساعدة في التخطيط واتخاذ القرارات المتعلقة بالاستخدام الأمثل للموارد المتاحة وذلك بهدف زيادة الأرباح وتخفيض التكاليف. لقد اتسعت استخدامات البرمجة الخطية لتشمل معظم نواحي الحياة سواء كان ذلك بالقطاع العام أم الخاص، في مؤسسة إنتاجية أو خدمية.

2.1. طرائق حل مسائل البرمجة الخطية:
(Solution Methods of Linear Programming Problems)

إن معظم مسائل البرمجة الخطية العملية هي كثيرة الأصفار، أي أن النسبة المئوية للعناصر اللاصفرية في مصفوفة المعاملات منخفضة. تعتمد الأهمية العلمية و النجاح التجاري لخوارزمية سيمبليكس المعدلة على استثمار البنية الصفرية. سنناقش في هذه المقالة جانباً هاماً من خوارزمية سيمبليكس المعدلة وهو تحليل LU كثيرة الأصفار. يعتبر حل مسائل البرمجة الخطية كثيرة الأصفار باستخدام تحليل LU كثيرة الأصفار مسألة صعبة لأن المحافظة على

البنية الصفرية في تحليل LU خلال تنفيذ خوارزمية سيمبليكس المعدلة صعب للغاية. إن تحليل LU كثيرة الأصفار هو جزء هام في تنفيذات خوارزمية سيمبليكس المعدلة. نركز اهتمامنا في هذه المقالة على تطوير خوارزميات تحليل LU كثيرة الأصفار لنستفيد من الصفة الصفرية لمسائل البرمجة الخطية كثيرة الأصفار.

(I). طريقة سيمبلكس المعدلة (Revised Simplex Method)

تستخدم طريقة سيمبليكس المعدلة بشكل واسع في حل مسائل البرمجة الخطية من مراتب كبيرة على حواسيب رقمية. تعطى مسألة البرمجة الخطية عموما بالصيغة التالية [1]:

$$(i)\qquad \begin{cases} \min\ Z = cx & (1) \\ \text{subject to}\ Ax = b & (2) \\ x \geq 0 \end{cases}$$

حيث Z هو تابع الهدف (Objective Function) و c هي متجهة معاملات الكلفة (Cost Coefficients) و x هي متجهة متغيرات مجهولة تتضمن المتغيرات المصطنعة (Slacks) و المقادير المطلوب حسابها (Surplus quantities) و A هي مصفوفة معاملات القيود (Coefficient matrix of constraints) و b هو متجه الطرف الأيمن و هو معلوم.

يمكن أن تكون مسألة الأمثليات المعرفة بالمعادلة (1) هي مسألة تعظيم (Maximization) لتابع الهدف و ليس مسألة تصغير (Minimization) كما هو معطى في (1). يمكن صياغة هذا النوع من المسائل بتغيير إشارة تابع الهدف. لذلك نقصر دراستنا على مسألة التصغير كما هي معرفة في المعادلة (1).

في كل خطوة تكرار لحل المسألة (i) تجزأ المصفوفة A إلى مصفوفة مربعة نظامية B و مجموعة من المتغيرات الموافقة x_b. بهذا الشكل نجزئ (2) فنحصل على:

$$\begin{bmatrix} A' & B \end{bmatrix} \begin{bmatrix} x_a \\ x_b \end{bmatrix} = \begin{bmatrix} b \end{bmatrix} \qquad (3)$$

والتي تكتب بالشكل:

$$A'x_a + Bx_b = b \qquad (4)$$

بشكل مماثل نجزئ تابع الكلفة (1) فنحصل على:

$$Z = c_a x_a + c_b x_b \qquad (5)$$

بما أن B هي مصفوفة مربعة نظامية فيمكن قلبها و بالتالي يمكن حساب x_b من المعادلة (4) كما يلي:

$$x_b = B^{-1}\left(b - A'x_a\right) \qquad (6)$$

لتحقق القيود المعطاة بالجملة الخطية (2) يجب أن تكون عناصر $B^{-1}b$ موجبة. إذا كانت المتغيرات x_a معدومة عندئذ تصبح المعادلة (6) بالشكل:

$$x_b = B^{-1}b \qquad (7)$$

بما أن المعادلة (7) تحقق جميع قيود مسألة التصغير فإن الحل المعطى بهذه المعادلة هو حل محتمل للمسألة (Possible Solution)، يعرف بالحل الممكن الأساسي (Basic Feasible Solution). علاوة على ذلك تدعى المتغيرات x_b بالمتغيرات الأساسية (Basic Variables) و المتغيرات x_a بالمتغيرات غير الأساسية (Non-basic Variables) و تدعى المصفوفة B بمصفوفة الأساس (Basic Matrix).

بتعويض المعادلة (6) في (5) نجد أن:

$$Z = c_b B^{-1} b + c'_a x_a \qquad (8)$$

حيث

$$c'_a = c_a - c_b B^{-1} A' \qquad (9)$$

(1) تحسين الحل الممكن الأساسي (Improving a basic feasible solution):

على الرغم من أن الحل الممكن الأساسي المحدد بالمعادلة (7) يحقق القيود (2) فإنه يمكن أن لا يكون حلا أمثليا للمعادلة (1). تعتمد قيمة تابع الهدف المحدد بالجملة (2) على قيم المتغيرات x_a. لقد اشتققنا الحل الممكن الأساسي بتحديد قيم المتغيرات x_a على أن تكون معدومة. من الممكن أنه إذا كانت بعض قيم x_a أكبر من الصفر فإن قيمة تابع الهدف قد تتناقص كما هو مطلوب. لنفرض أن قيمة متغير غير أساسي و ليكن x_i قد ازدادت. فإذا كانت أمثال x_i و لتكن c'_i سالبة فإن قيمة تابع الهدف في (8) ستتناقص مما يشير إلى أن الحل الممكن الأساسي الناتج غير أمثلي. تمكن طريقة سيمبليكس المعدلة في اختيار المتغير x_i الموافق للأمثال c'_i و تبديله بمتغير أساسي. تكرر هذه العملية حتى تكون جميع الأمثال c'_i إما معدومة أو موجبة. في هذه الحالة لا تتناقص قيمة تابع الهدف و الحل الممكن الأساسي الناتج هو الحل الأمثل.

(2) تبديل المتغيرات الأساسية و غير الأساسية:

(Interchanging basic and non-basic variables)

إذا وجدنا من العملية الموصوفة في الفقرة السابقة أن تابع الهدف تناقص عند تبديل متغير غير أساسي x_i بمتغير أساسي فإن أحد متغيرات الأساس الموجودة يجب أن يستبدل بمتغير غير أساسي. تكمن المشكلة في إيجاد أي من متغيرات الأساس يجب أن يخرج من الأساس. يمكن كتابة المعادلة (6) بالشكل التالي:

$$\begin{aligned} x_b &= b' - B^{-1} A' x_a \\ b' &= B^{-1} b \end{aligned} \qquad (10)$$

بما أن قيمة متغير غير أساسي واحد فقط x_i قد ازدادت عن الصفر فإنه يمكن كتابة المعادلة (10) بدلالة x_i فقط كما يلي:

$$x_b = b' - B^{-1} p_i x_i \qquad (11)$$

حيث p_i هو العمود i من المصفوفة A'. يمكن التعبير عن المعادلة (11) بالشكل:

$$\begin{aligned} x_b &= b' - p'_i x_i \\ p'_i &= B^{-1} p_i \end{aligned} \qquad (12)$$

عندما يدخل المتغير x_i إلى الأساس فإنه ينبغي أن تكون قيمة متغير الأساس الخارج مساوية الصفر دون جعل أي من المتغيرات الأخرى سالبة. لتحديد المتغير الذي نحذفه و يحقق هذا المعيار نوجد دليلاً j يوافق السطر j من الأساس و يحقق الشروط:

$$\min \frac{b'_j}{p'_{ji}} \;\&\; p'_{ji} > 0 \qquad (13)$$

حيث b'_j هو العنصر j من b' و p'_{ji} هو العنصر j من p'_i.

المتغير الذي يحقق الشروط المحددة بالمعادلة (13) هو متغير غير أساسي و قيمته تساوي الصفر. في الخطوة التكرارية r يمكن حساب معكوس مصفوفة الأساس الجديدة B_r^{-1} من معكوس مصفوفة الأساس السابقة B_{r-1}^{-1} باستخدام مصفوفة التحويلات (Transformations matrix) T_r كما يلي:

$$B_r^{-1} = T_r B_{r-1}^{-1} \qquad (14)$$

حيث

$$T_r = \begin{bmatrix} 1 & & & & & & \\ & 1 & & -\frac{p'_{1i}}{p'_{ji}} & & & \\ & & 1 & \vdots & & & \\ & & & \frac{1}{p'_{ji}} & & & \\ & & & \vdots & \ddots & & \\ & & & -\frac{p'_{mi}}{p'_{ji}} & & 1 & \\ & & & & & & 1 \end{bmatrix} \qquad (15)$$

حيث m هو عدد القيود. باستخدام المتغيرات الاصطناعية إن كان ضروريا يمكن أن نبدأ العملية التكرارية بمصفوفة أساس واحدية. في نهاية التكرارات تصبح المتغيرات الاصطناعية غير أساسية. باستخدام هذه الطريقة فإن معكوس الأساس في أي خطوة تكرارية يمكن التعبير عنه على شكل جداء مصفوفات التحويلات. بالفعل معكوس الأساس r هو:

$$B_r^{-1} = T_r T_{r-1} .. T_2 T_1 \qquad (16)$$

إذا كانت مصفوفة القيود A كثيرة الأصفار فإن مصفوفة التحويلات يمكن أن تكون أيضا كثيرة الأصفار. لذلك يمكن استثمار طرائق كثيرة الأصفار في مثل هذه الحالات أي فقط العناصر اللاصفرية في العمود المعني من مصفوفة التحويلات يطلب تخزينها و التعامل معها. هذه النقطة هي إحدى أهم أفكار هذه المقالة.

تستخدم هذه الطريقة نفس المبادئ الأساسية التي تستخدمها الطريقة النظامية، إلا أن الحسابات في كل تكرار للحل لا تجرى على كامل الجدول، فالمعلومات اللازمة للانتقال من الحل الأساسي النافذ إلى الحل التالي يمكن استنتاجها من المعادلات الأصلية، و هذه النقطة هي محور البحث عن خوارزميات مطورة لحل هذه النقطة.

للانتقال من الحل الأساسي النافذ إلى الحل الذي يليه تتطلب طريقة سيمبليكس المعدلة ما يلي:

1. السطر c: (معاملات coefficients الربح النسبي) لتعيين المتغير غير الأساسي للدخول إلى أساس الحل.

2. عمود التمحور W : العمود الموافق للمتغير غير الأساسي الداخل إلى أساس الحل.

3. ثوابت الطرف الأيمن لإنجاز اختبار النسبة الأصغرية و لتعيين المتغير الأساسي للخروج من أساس الحل.

4. مصفوفة قاعدة الحل B التي عناصرها هي الأعمدة الأصلية الموافقة للمتغيرات الأساسية من المرتبة $m \times m$. و يرمز لمقلوب مصفوفة أساس الحل بالرمز B^{-1} و تحسب بالطريقة الموسعة و يعطي b قيم المتغيرات الأساسية. و V هي مصفوفة غير مربعة تشمل الأعمدة غير الأساسية الموافقة للمتغيرات غير الأساسية. توجد خطوتان أساسيتان في الطريقة المعدلة و هي اختيار المتغير غير الأساسي للدخول إلى أساس الحل و تعيين المتغير الأساسي للخروج من أساس الحل تنجز هذه الخطوات الأساسية بالطريقة المعدلة دون إجراء الحسابات على كامل الجدول في كل تكرار بحساب: $D = c_B \times B^{-1} \times p_j - c_V$ وهو سطر c (حيث p_j العمود من V). يظهر التحليل الخوارزمي لطريقة سيمبلكس المعدلة أن الخطوة التي لها أكبر قيمة هو حساب معكوس مصفوفة الأساس.

(II). طرائق تحليل LU لحل مسائل البرمجة الخطية:

(LU decomposition methods for linear programming problems)

أصبحت طريقة استخدام تحليل LU في البرمجة الخطية واضحة المعالم في معظم برامج حل مسائل البرمجة الخطية باستخدام المعكوس (Linear programming inversion routines). لقد توضحت أفضلية هذا الشكل للمعكوس على شكل الجداء الأصلي بدلالة السرعة و الدقة و الذاكرة في التطبيقات العملية و قد أصبحت هذه الطريقة هي الخوارزمية القياسية (Standard algorithm). يعد ماركويتز (Markowitz) [2] أول من اقترح استخدام تحليل LU عام 1957 بينما يعتبر دانتزيغ (Dantzig) [3] أول من نفذ الأساس بالشكل المثلثي من تكرار إلى آخر من أجل مصفوفات ذات بنية محددة لتوزع العناصر اللاصفرية في العام 1963. لم يظهر هذا الاقتراح إلى حيز النور لحين قريب بسبب التبسيطات الكبيرة في شكل الجداء القياسي [4] و الصعوبات الأساسية في تعديل و تحديث المعاملات المثلثية L و U لمصفوفات كبيرة كثيرة الأصفار بتقنية مصفوفة كثيرة الأصفار.

لقد طور العديد من الباحثين طرائق متعددة لتحديث المعاملات المثلثية L و U. من هذه الطرائق: طريقة Golub – Bartels [5] و طريقة Golub – Bartels – Saunders [6] التي تعتمد على حذف غاوس و هي مشابهة في جوانبها لطريقة دانتزيغ (Dantzig) [3]. و طريقة Gill – Murrary [7] التي تستخدم الخواص المفضلة للتحويلات المتعامدة [8]. لقد قدم Brayton – Gustafson - Willoughby [9] عدة طرائق لتحديث المعاملات المثلثية التي صممت خصيصا للمحافظة على البنية اللاصفرية لمصفوفة الأساس. طرأت تطويرات كبيرة على هذه الطرائق منها طريقة Forrest – Tomlin [10] والتي استخدمت بشكل واسع في حل مسائل البرمجة الخطية من مراتب عليا باستخدام UMPIRE [11] و طريقة Reid [1].

1. خوارزميات تحليل LU المقترحة لحل مسائل البرمجة الخطية

إن طريقة تحليل LU كثيرة الأصفار هي إحدى الطرائق المباشرة المعروفة تماما لحل (7). لنفرض أننا نريد حل (7) وأنه يمكن أن نعبر عن مصفوفة الأساس B في الشكل المسمى تحليل LU أي $B = LU$. عندئذ يمكن حل (7) بالإجراء التالي:

المرحلة 1: نكتب $Bx_b = LUx_b = b$.

المرحلة 2: نضع $y_b = Ux_b$ بحيث أن $Bx_b = Ly_b = b$. عندئذ نستخدم التعويض التقدمي على $Ly_b = b$ لإيجاد $y_1, y_2, .. , y_m$ على الترتيب.

المرحلة 3: نستخدم التعويض التراجعي على $Ux_b = y_b$ لإيجاد x_1 , ...x_m على الترتيب.

(A) طريقة تحليل LU المباشرة كثيرة الأصفار (Direct Sparse LU- Decomposition).

تعتبر طريقة تحليل LU المباشرة كثيرة الأصفار مفيدة جدا لحل (7) كثيرة الأصفار الكبيرة بأطراف يمنى مختلفة. إن هذه الطريقة في هذه الحالة هي أكثر فعالية من الطرائق التقليدية. حالما نجد تحليل LU للمصفوفة B نستخدم فقط التعويضين التقدمي والتراجعي لحل الجملة من أجل أي b.

إن المفهوم الهام في تحليل LU كثيرة الأصفار هو تحديد البنية الصفرية للمعاملين L و U. علاوة على ذلك فإن تحديد العناصر fill-ins (العناصر fill-ins هي عناصر تساوي الصفر في المصفوفة B و تصبح عناصر مغايرة للصفر عند تحليل المصفوفة B) لمصفوفة كثيرة الأصفار B هو مسألة مركزية في حل (7) كثيرة الأصفار. إن ظهور العناصر fill-ins في تحليل LU ليس هو المشكلة. فالمشكلة الرئيسية هي أن المواقع التي تظهر فيها العناصر fill-ins غير معلومة مسبقا. الفكرة الأساسية من طريقة تحليل LU كثيرة الأصفار هي تحديد البنية الصفرية لمعاملي LU في طريقة جيدة تتفادى عوائق تلك الطرائق المتيسرة في المراجع. نقدم خوارزمية فعالة وغير مكلفة لتحديد البنية الصفرية للمعاملين L وU. تعتمد هذه الخوارزمية الجديدة على قوى المصفوفة البوليانية R التي تنتج من مصفوفة الأساس B بحيث أنه يمكن أن نعرف مسبقا العناصر fill-ins في بنية المصفوفة B. لقد طورنا نظام تخزين اقتصادي للجملة (7) بحيث أنه ليست هناك عناصر fill-ins في تحليل المصفوفة B. تحلل المصفوفة B وفق نظام التخزين الجديد. حالما نحصل على البنية اللاصفرية للمعاملين L وU فإن بناء تحليل LU يكون مباشرا. تم اختيار طريقة تحليل LU كثيرة الأصفار لأنها تسمح بحلول مكررة للمتجه x من أجل قيم عديدة للمتجه b دون إعادة التحليل. نحصل على حل (7) ببنية صفرية كيفية باستخدام طريقة تحليل LU كثيرة الأصفار المحددة في هذه المقالة.

☒ خوارزمية تحليل LU كثيرة الأصفار المقترحة:

نشرح في هذه الفقرة كيف نحل (7) باستخدام تحليل LU كثيرة الأصفار. نستخدم في بناء المعاملين L و U طريقة قوى مصفوفة بوليانية (PBS) المحددة في هذه الطريقة لتحديد البنية اللاصفرية لمعاملي LU لمصفوفة معطاة B. نقدم أيضا خوارزمية لحساب المعاملين L و U. تعطي تقريبا هذه الخوارزمية مع PBS تحليل LU الدقيق للمصفوفة B.

(1) طريقة قوى مصفوفة بوليانية **(PBS)**:

نصف في هذه الفقرة خوارزمية مطورة فعالة لتحديد البنية الصفرية P لمعاملي LU التي ليس لها قيود على الإطلاق بالنسبة للبنية الصفرية للمصفوفة B. تعتمد هذه الخوارزمية على قوى مصفوفة بوليانية R تنتج من المصفوفة B. توصف البنية الصفرية إما مسبقا أو ضمنا بطريقة ما. علاوة على ذلك من المرغوب أن نعرف البنية اللاصفرية لمعاملي LU لتحديث بنية المعطيات لاستيعاب العناصر اللاصفرية والعناصر fill-ins أيضا. فوق ذلك تعطي PBS كما سنرى المعلومات الضرورية لحساب البنية اللاصفرية لمعاملي LU. بعد تحديد البنية الصفرية لمعاملي LU فإن حساب L و U يكون مباشرا. المشكلة الأساسية هي أن نوجد المجموعة P من العناصر التي يكون من أجلها المعاملان L و U كثيري أصفار وبحيث أن المصفوفة LU تكون تقريبا جيدا للمصفوفة B قدر الإمكان.

لنعتبر البيان الموجه $G_B=(V,E)$ الموافق للمصفوفة المعطاة B. إذا كانت R هي المصفوفة البوليانية التي تمثل البيان الموجه G_B عندئذ يعرف البيان الموجه المعدل $G_{R^m}=(V,E_m)$ بالشكل $E_m=E\cup\{(v_i,v_k)\}$ أي أن كل قوس موجه جديد (v_i,v_k) يضاف إلى البيان G_B لتشكيل البيان G_{R^m} حيث m هو عدد صحيح موجب ما. في البداية البنيتان الصفريتان للمصفوفتين B و R هما نفساهما أي أن B و R تملكان بالضبط العناصر اللاصفرية في نفس المواقع. لكن المشكلة هي أن نحصل على البيان الموجه المعدل G_{R^m} في المستوى m. علاوة على ذلك عند إيجاد قوى المصفوفة R فإن بعض العناصر الصفرية في المصفوفة R تصبح 1 في R^m. كل عنصر كان في البداية صفرا في R ويصبح 1 في R^m يعطي قوسا موجها جديدا وليكن (v_i,v_k)، والذي يضاف إلى G_B لتشكيل البيان G_{R^m}. هذه العناصر هي بالضبط مواقع العناصر fill-ins في B. نحاول الآن إيجاد البنية الصفرية للمعاملين L و U بلغة نظرية المجموعات.

لتحديد البنية اللاصفرية للمعاملين L و U نعرف المجموعة

$$P=\{(i,j):\ position\ (i,j)\ is\ non-zero\ including\ fill-in,\ 1\le i,j\le n\} \qquad (17)$$

عندئذ بوضوح $P=\{(i,j):r_{ij}^{(m)}=1\}$ حيث $1\le m\le n-1$, $R^m=\left[r_{ij}^{(m)}\right]$ و m لايتجاوز المسار الأطول في G_B. لاحظ أن كلتا المجموعتين P و E_m لهما نفس العناصر. وهكذا تعطي المجموعة P البنية الصفرية لمعاملي LU.

لتكن $R=\left[r_{ij}\right]$ تعطى بالشكل:

$$r_{ij}=\begin{cases}1, & if\ \ b_{ij}\neq 0\\ 0, & otherwise\end{cases} \qquad (18)$$

تلخص الطريقة لإيجاد المجموعة P في الخوارزمية التالية:

خوارزمية 1: تقنية قوى مصفوفة بولیانية (PBS):

الخطوة 1:

نشكل المصفوفة R كما هي معرفة في (18)

الخطوة 2:

نحسب R^{2^k} $(k\ge 1)$

إذا كان $R^{2^k}=R^{2^{k+1}}$ عندئذ

نشكل المجموعة $P=\{(i,j):r_{ij}^{2^k}=1\}$

وإلا

$k=k+1$ ونذهب إلى الخطوة 2.

ينتج من الخوارزمية أن البنية الصفرية لمعاملي LU هي تقريبا مساوية لتلك من المصفوفة R^{2^k}. في أي تكرار معطى إذا كانت المصفوفة البوليانية المحسوبة متوافقة مع المصفوفة في التكرار السابق أي أن $R^{2^k}=R^{2^{k+1}}$ فإن العملية تتقارب ونحصل على البنية الصفرية لمعاملي LU. إذا كان $R^{2^k}=R^{2^{k+1}}$ فإننا نحصل على تحليل LU التام. وهكذا يمكن أن تستخدم بسهولة طريقة تحليل LU كثيرة الأصفار لحل (7).

(2) تحليل الخوارزمية 1:

يفترض تحليل أي خوارزمية أن عمليات معينة يمكن أن تنفذ بكلفة ثابتة. في الأساس تحليل خوارزمية هو التعقيد الزمني لخوارزمية الذي يحدد كيف تحل الخوارزمية مسألة كبيرة. لذلك تعطينا مرتبة زمن تنفيذ الخوارزمية تقديرا عن إمكانيتها العملية. يجب أن لا ننسى أن مرتبة دالة تعقيد خوارزمية هو مقياس لتأدية الخوارزمية عندما يسعى بعد الدخل إلى اللانهاية.

تعريف 1:

لنفرض أن $B' = \lfloor b'_{ij} \rfloor$ و $B'' = \lfloor b''_{ij} \rfloor$ هما مصفوفتان بوليانيتان مربعتان من المرتبة m وأن "AND" هو مؤثر بولياني معرف على عناصر B' و B'' . الجداء البولياني للمصفوفتين B' و B'' ، ونرمز له بالرمز $B'B''$ ، هو المصفوفة البوليانية $C = \lfloor c_{ij} \rfloor$ المربعة من المرتبة m المعرفة كما يلي:

$$c_{ij} = \begin{cases} 1 \ (True) & if \ b'_{ik} = 1 \ AND \ b''_{kj} = 1 \ for \ some \ k, \ 1 \le k \le m \\ 0 \ (False) & otherwise. \end{cases} \qquad (19)$$

ينتج من التعريف أن $c_{ij} = 1$ إذا وفقط إذا كان $b'_{ik} = 1 \ AND \ b''_{kj} = 1$. بما أننا بصدد مشكلة محددة فإننا نستخدم المؤثر البولياني "AND" الذي يقيد النتائج على تلك العناصر التي نبحث عنها في مشكلتنا وهكذا تنفذ بسهولة المقارنات ونتأكد من كل موقع للجداء البولياني.

لإيجاد البنية الصفرية للمعاملين L و U بحيث أن $B = LU$ يكفي أن نوجد العدد الأصغر l بحيث أن $R^{2^l} = R^{2^{l+1}}$ حيث $k = l$ هو المرة الأولى التي تصبح فيها المصفوفة R^{2^k} مساوية $R^{2^{k+1}}$. زمن التنفيذ المصروف في حساب R^{2^k} هو $O\left(k \, s(m)\right)$ حيث $s(m)$ هو الزمن المطلوب لحساب الجداء المصفوفي البولياني لمصفوفتين بوليانيتين من المرتبة $m \times m$. يتطلب الجداء البولياني لمصفوفتين بوليانيتين من المرتبة $m \times m$ على الأكثر $O\left(m^2\right)$ عملية مقارنة (المؤثرات البوليانية AND). وهكذا لحساب المصفوفات البولينية R^{2^k} نحتاج إلى $O\left(km^2\right)$ عملية مقارنة.

يمكن تقدير متطلبات الذاكرة لمصفوفة بوليانية R بـ m^2 بيتا بصرف النظر عن كون المصفوفة R كثيرة الأصفار. إذا كانت المصفوفة B ضعيفة الشرطية فإن الخوارزمية لا تعاني من هذا العائق لأن قوى مصفوفة بوليانية لا تعتمد على قيم العناصر في المصفوفة B . بالنتيجة فإن الخوارزمية فعالة جدا وغير مكلفة لحساب العناصر fill-ins. ولذلك نستخدم الخوارزمية لنعرف مسبقا البنية اللاصفرية للمعاملين. هناك تقابل طبيعي بين العلاقات الثنائية والمصفوفات البولينية المربعة.

تعريف 2:

لتكن E_1 و E_2 علاقتين ثنائيتين على مجموعة $V = \{v_1, v_2, ..., v_n\}$. عندئذ $E = E_1.E_2$ هي العلاقة الثنائية:

$$E = \left\{ (v_i, v_j) : (v_i, v_k) \in E_1 \ AND \ (v_k, v_j) \in E_2 \ \ for \ some \ k \right\}$$

مبرهنة 1:

ليكن $G_1(V, E_1)$ و $G_2(V, E_2)$ بيانين موجهين مرتبطين بالمصفوفات البولينية B' و B'' على الترتيب. عندئذ يكون الجداء المصفوفي $C = B'B''$ هو المصفوفة البولينية للبيان الموجه $G(V, E_1.E_2)$.

البرهان:

لنفرض أن $E = E_1.E_2$ حيث $E = \{(v_i,v_k):(v_i,v_j)\in E_1 \ AND \ (v_j,v_k)\in E_2 \ for \ some \ j\}$. وهكذا إذا كانت C هي المصفوفة البوليانية للبيان $G(V,E)$ عندئذ

$$c_{ik} = 1 \ iff \ for \ some \ j, b'_{ij} = 1 \ AND \ b''_{jk} = 1$$

هذا بالضبط نفس الشيء كأن نقول $c_{ik} = b'_{ij} \ AND \ b''_{jk}$.

حالما نوجد المجموعة P باستخدام الطريقة المحددة فإنه يمكن حساب تحليل LU التام باستخدام الخوارزمية المعطاة في الفقرة التالية.

☒ خوارزمية حساب LU كثيرة الأصفار:

حالما نحصل على البنية اللاصفرية للمعاملين L و U نحلل المصفوفة B ونحصل على العناصر اللاصفرية بطريقة دوليتل [6,7] حيث تكون كل العناصر القطرية للمصفوفة L مساوية 1. تعطى $B = LU$ بالصيغة:

$$b_{ij} = \sum_{k=1}^{\min(i,j)} l_{ik}u_{kj} \qquad (20)$$

تعطي هذه العلاقة الصيغ الصريحة التالية من أجل l_{ij} و u_{ij}:

$$l_{ik} = \frac{b_{ik} - \sum_{j=1}^{k-1} l_{ij}u_{jk}}{u_{kk}}, \qquad i > k \qquad (21)$$

$$u_{ik} = b_{ik} - \sum_{j=1}^{i-1} l_{ij}u_{jk}, \qquad i \le k \qquad (22)$$

نقدم الآن كيف نبني تحليل LU التام لمصفوفة المعاملات في المعادلة (7). حالما نحصل على البنية اللاصفرية للمعاملين L و U، أي عندما تحدد المجموعة

$$P = \{(i,j): r_{ij}^{(m)} = 1\} \qquad (23)$$

بالخوارزمية 1 فإن حساب تحليل LU يكون مباشرا.

يعطى جداء مصفوفة بمتجه $u = Qx$ باستخدام القائمة المرتبة السطرية لطريقة التخزين المتبعة بالخوارزمية التالية:

خوارزمية2: جداء مصفوفة بمتجه $u = Qx$

```
For i = 1 To m Do
        {
    u [ i ] = 0 ;
    For v = IQ [ i ] To IQ [i+1]-1 Do
      u [ i ]  = u [ i ] + VQ [v] * x [ JQ [v] ] ;
  }//End For i..
```

عند حساب تحليل LU التام نحتاج إلى تخزين العناصر اللاصفرية للمعاملين L و U. نعرف متجهة مساعدة **Diag** $[1...n]$ التي تشير إلى العناصر القطرية للمصفوفة U في المتجهة VB. تخزن البنية اللاصفرية P للمعاملين L و U في JB و IB و VB والتي تحتوي $b_{ij} \neq 0$ والعناصر fill-ins أيضا.

تحسب الخوارزمية التالية التحليل التام. المتغير البولياني revise هو false من أجل التحليل التام القياسي وtrue من أجل التحليل المعدل بحيث أن المجاميع السطرية لمصفوفة الخطأ Error $(=E)=B-LU$ تساوي الصفر.

المتجه Point$[1...n]$ هو متجهة من الأعداد الصحيحة التي تشير إلى العناصر في L و U من السطر i.

خوارزمية 3: تحليل LU التام

```
For i = 1 To m Do
   Point [ i ] = 0;
For i = 2 To m Do
  {
    For v = IB [ i ]+1 To IB [ i+1 ]-1 Do
      Point [ JB [ v ] ] = v;
    For v = IB [ i ] To Diag [ i ]-1 Do
     {
       j = JB [ v ] ;
      VB [ v ]  = VB [ v ] / VB [ Diag [ j ] ] ;
      For w = Diag [ j ]+1 To IB [ j+1 ] -1 Do
        {
          k = Point [ JB [ w ] ] ;
          If ( k>0 ) then
            VB [ k ] = VB [ k ] – VB [ v ] * VB [ w ] ;
          Else
            If ( revised ) then
              VB [ Diag [ i ] ] = VB[ Diag [ i ] ] – VB [ v ] * VB [ w ] ;
        }//End For w.
     }//End For v.
    For v = IB [ i ]+1 To IB [ i+1 ]-1 Do
      Point [ JB [ v ] ] = 0 ;
  }//End For i.
```

☒ حل (7) بطريقة تحليل LU كثيرة الأصفار:

حالما تحدد المجموعة

$$P=\left\{(i,j):r_{ij}^{(m)}=1\right\} \qquad (24)$$

بالخوارزمية 1 فإنه يمكن حساب تحليل LU باستخدام الخوارزمية 3. بالنتيجة نحصل على الحل x للجملة (7) بالخوارزمية التالية:

خوارزمية 4 حل الجملة $Bx_b=b$

تنفذ في الخطوات التالية:

1. تحديد العناصر fill-ins باستخدام الخوارزمية 1.
2. استدعاء الخوارزمية 3 لبناء تحليل **LU**.
3. حساب الحل x_b باستخدام:

(i) طريقة التعويض التقدمي لحل $Ly_b=b$.

(ii) طريقة التعويض التراجعي لحل $Ux_b = y_b$.

حيث تعطى الخطوة 3 في الخوارزميتين التاليتين:

خوارزمية 5: طريقة التعويض التقدمي

```
  y[ 1 ] = b [ 1 ] ;
For i = 2 To m Do
    {
     Sum = 0 ;
        For j = IB [ i ] To  Diag [ i ] - 1  Do
         Sum = Sum + VB [ j ] * y[ JB [ j ] ] ;
      y[ i ] = b [ i ] - Sum ;
  }//End For i.
```

خوارزمية 6: طريقة التعويض التراجعي

```
 x [m ] = y [m ] / VB [ Diag [ m ] ] ;
For i = m-1 Down To 1 Do
   {
     Sum1 = 0 ;
      For j = Diag [ i ] To IB [ i+1] -1 Do
        Sum1 = Sum1 + VB [ j ] * x [ JB [ j ] ] ;
     x[ i ] = ( y [ i ] – Sum1 ) / VB [ Diag [ i ] ] ;
  }//End For i.
```

المصفوفات المثلثية مهمة جدا في حسابات المصفوفة. تعزى أهميتها إلى حقيقة أن كل الطرائق المباشرة (والعديد من الطرائق التكرارية) لحل جمل المعادلات الخطية تتضمن حل جمل مثلثية والتي يمكن حلها بسهولة باستخدام التعويض التقدمي و التراجعي باستخدام الخوارزميات 5 و 6 على الترتيب. تتجنب هذه الطريقة مشكلة قلب المصفوفة فبدلا من قلب المصفوفة بشكل كامل تم تحليل الأساس إلى مصفوفة مثلثية علوية U و سفلية L.

(B) الطريقة التكرارية (Iterative Method):

تتلخص هذه الطريقة لحل المسألة (7) كما يلي:

يمكن كتابة (7) بالشكل $BB^Ty=b$; $x=B^Ty$. ينتج الحل في فضاء كريلوف:

$$x_0 + B^T K_m(BB^T, r_0) = x_0 + span\left\{B^T r_0, (B^T B)B^T r_0, ..., (B^T B)^{m-1} B^T r_0\right\}$$

و ذلك بهدف تخفيض $x_* - x$ حيث $x_* = B^{-1}b,\ r_0 = b - Bx_0$ where. الخطأ الجديد $b - Bx_{m+1}$ متعامد مع فضاء كريلوف المولد بالأساس $v_1, v_2, ..., v_m$ بحيث أن r_{m+1} يقع في اتجاه v_{m+1}.

إذا رمزنا بالرمز R_j للمصفوفة التي أعمدتها r_j نحصل على العلاقة التالية $BR_m = R_{m+1}T_{m+1,m}$ حيث T هي مصفوفة ثلاثية الأقطار. بما أن الحل x_m في $K_m\left(B, r_0\right)$ فإنه يمكن كتابته كتركيب خطي لمتجهات أساس فضاء كريلوف و لذلك $x_m = R_m y$. ينص شرط كلاركين– ريتز (Ritz-Galarkin) أن الخطأ الناتج من الحل x_m متعامد مع المتجهات $r_0, ..., r_{m-1}$: $R_m^T\left(Bx_m - b\right) = 0 \Rightarrow R_m^T BR_m - R_m^T b$. حسب $BR_m = R_{m+1}T_{m+1,m}$ ينتج أن $R_m^T R_m T_m y \ \|r_0\|_2^2 e_1$ حيث $R_m^T R_m$ هي مصفوفة قطرية بالعناصر القطرية $\|r_0\|_2^2 \ \ldots \ \|r_{m-1}\|_2^2$. يمكن حساب الحل المنشود كما يلي $T_{mm} y = e_1 \Rightarrow x_m = R_m y$. لقد استخدمنا حقيقة أن B متناظرة و أن T_m ليست شاذة. إذا كانت المصفوفة B موجبة محددة و باستخدام العلاقة $R_m^T BR_m = R_m^T R_m T_m$. فإنه يمكن تحويل المصفوفة T_m باستخدام المصفوفة $R_m^T R_m$ إلى مصفوفة ثلاثية الأقطار متناظرة موجبة محددة. هذا يقتضي أنه يمكن تحليل T_m إلى

LU ودون استخدام الارتكاز $T_m = L_m U_m$. حيث L_m هي مصفوفة مثلثية سفلية و U_m هي مصفوفة مثلثية علوية واحدية. تعريف $P_m = R_m U_m^{-1}$ يقودنا إلى متجهات إضافية p_j التي تشكل $(Bp_i, p_j) = 0$ من أجل $i \neq j$. هذا ينتج من حقيقة أن $P_m^T BP_m$ هي مصفوفة قطرية $P_m^T BP_m = U_m^{-T} V_m^T B V_m U_m^{-1} = U_m^{-T} T_m U_m^{-1} = U_m^{-T} L_m$. لاحظ أن $U_m^{-T} L_m$ هي مصفوفة مثلثية سفلية متناظرة لأنها تساوي المصفوفة المتناظرة $P_m^T BP_m$ التي يجب أن تكون قطرية. يمكن اشتقاق الخوارزمية النهائية بافتراض تعامد متجهات الخطأ كل منها مع بعضها الآخر وبافتراض شروط ترافق p_i. يمكن التعبير عن المتجه x_{j+1} بالصيغة التالية $x_{j+1} = x_j + \alpha_j p_j$. و لذلك يجب أن تحقق متجهات الخطأ العلاقة التالية $r_{j+1} = r_j - \alpha_j Bp_j$. إذا كانت r_j متعامدة عندئذ يلزم أن يكون $(r_j - \alpha_j Bp_j, r_j) = 0$ وبالنتيجة $\alpha_j = \dfrac{(r_j, r_j)}{(Bp_j, r_j)}$. من المعلوم أن متجه البحث التالي p_{j+1} هو تركيب خطي للمتجهين r_{j+1} و p_j و بعد إعادة سلمية المتجهات p (Rescaling) بشكل مناسب ينتج أن $p_{j+1} = r_{j+1} + \beta_j p_j$. و هكذا فإن النتيجة الأولى من العلاقة أعلاه هي أن $(Bp_j, r_j) = (Bp_j, p_j - \beta_{j-1} p_{j-1}) = (Bp_j, p_j)$ لأن Bp_j متعامدة مع p_{j-1}. عندئذ تصبح علاقة α_j بالشكل $\alpha_j = \dfrac{(r_j, r_j)}{(Bp_j, p_j)}$. بالإضافة لذلك فإن كتابة p_{j+1} متعامدة مع Bp_j يقتضي أن $\beta_j = -\dfrac{(r_{j+1}, Bp_j)}{(p_j, Bp_j)}$

مما سبق يمكننا أن نكتب $Bp_j = -\dfrac{1}{\alpha_j}(r_{j+1} - r_j)$

و لذلك $$\beta_j = \frac{1}{\alpha_j} \frac{(r_{j+1}, (r_{j+1} - r_{mj}))}{(Bp_j, p_j)} = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}$$

بوضع هذه العلاقات معا نحصل على الخوارزمية التالية:

خوارزمية 7 (Algorithm 7):

1- Compute $r_0 = b - Bx_0$, $p_0 = B^T r_0$

2- for $k = 0, 1, \ldots$. until convergence do

- $\alpha_k = \dfrac{(r_k, r_k)}{(p_k, p_k)}$
- $x_{k+1} = x_k + \alpha_k p_k$
- $r_{k+1} = r_k - \alpha_k Bp_k$
- $\beta_k = \dfrac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}$
- $p_{k+1} = B^T r_{k+1} + \beta_k p_k$

End for

ينتج من الخوارزمية 7 أنه يتم تحديث x_m من x_{m-1} و أن الخوارزمية 7 لا تعتمد على تطوير أو تحديث V .

2.3 . نظام التخزين (Storage Scheme):

يمكن تصنيف المصفوفات إلى صفين طبقا لنوع الخوارزميات التي تناسبها. الصنف الأول هو المصفوفات الكثيفة أو الممتلئة أو دون عناصر صفرية. الصنف الثاني هو المصفوفات كثيرة الأصفار التي تكون معظم عناصرها أصفارا. تخزن المصفوفات الكثيفة في ذاكرة الحاسب باستخدام متجهتين ببعدين. من أجل مصفوفة من المرتبة $m \times m$ فإنها تخزن باستخدام متجهة $m \times m$ من الأعداد الحقيقية. علاوة على ذلك، استخدام نفس المتجهة ببعدين لتخزين المصفوفات كثيرة الأصفار له عائقين هامين جدا. الأول بما أن معظم العناصر في المصفوفة كثيرة الأصفار هي أصفار فيمكن أن يفقد نظام التخزين كثيرا من الذاكرة. الثاني الحسابات التي تتضمن مصفوفات كثيرة الأصفار تتطلب العمل مع العناصر اللاصفرية فقط للمصفوفة. لهذين السببين تخزن المصفوفات كثيرة الأصفار باستخدام بنى معطيات مختلفة. التخزين المطلوب لمصفوفة كثيرة الأصفار يمكن أن يخفض إلى تخزين العناصر اللاصفرية في التخزين المطلوب. هناك طرائق عديدة يمكن أن تخزن فيها العناصر اللاصفرية.

يعتمد التمثيل الموجز لتقنية التخزين الموصوفة هنا على الفكرة المحددة في المراجع [12]. يدعى هذا النظام بنظام التخزين غير المتراص. شكل التخزين عام ولا يتطلب أي افتراضات عن البنية الصفرية للمصفوفة ولا يخزن أي عناصر غير ضرورية [12]. علاوة على ذلك، يستخدم هذا النظام بشكل واسع لتخزين المصفوفات كثيرة الأصفار بنماذج صفرية كيفية [12]. النسخة المقدمة هنا هي القائمة المرتبة السطرية التي تخزن فيها العناصر اللاصفرية سطرا سطرا بكل سطر تخزن العناصر اللاصفرية في الترتيب المتزايد لأدلة العمود. لتحديد عناصر أي سطر من الضروري أن نعرف متى يبدأ السطر وكم عنصرا لاصفريا يحوي وفي أية أعمدة تقع العناصر اللاصفرية.

لنفرض أنه لدينا مصفوفة B كثيرة الأصفار من المرتبة $m \times m$ تملك n-nz عنصرا لاصفريا يتطلب تخزين المصفوفة B وفق نظام التخزين غير المتراص ثلاثة متجهات أحادية البعد IB, JB, VB بأطوال n-nz و n-nz و $m+1$ على الترتيب. تحوي المتجهة VB العناصر اللاصفرية في B المخزنة سطرا – سطرا وتحوي المتجهة JB أدلة العمود التي توافق العناصر اللاصفرية في المتجهة VB وتحوي IB مؤشرات $m+1$ تحدد أسطر العناصر اللاصفرية في المتجهة VB. إذن $2\text{(n-nz)}+\text{m}+1$، كلمة تخزين مطلوبة لتمثيل B بدلا من $m \times m$ كلمة في نظام مصفوفة كثيفة الموافق أي قد خفضت متطلبات التخزين من $m \times m$ وحدة إلى $2\text{(n-nz)}+\text{m}+1$ وحدة. لندرس المصفوفة B المعرفة كما يلي:

$$B = \begin{bmatrix} b_{11} & 0 & b_{13} & 0 & 0 \\ b_{21} & b_{22} & b_{23} & 0 & 0 \\ 0 & 0 & b_{33} & 0 & 0 \\ b_{41} & 0 & 0 & b_{44} & b_{45} \\ 0 & b_{52} & 0 & 0 & b_{55} \end{bmatrix} \qquad (25)$$

عندئذ تكون القائمة المرتبة السطرية لنظام تخزين المصفوفة B كما يلي:

$$VB = \overbrace{b_{11}\ b_{13}}^{row\ 1}\ \overbrace{b_{21}\ b_{22}\ b_{23}}^{row\ 2}\ \overbrace{b_{33}}^{row\ 3}\ \overbrace{b_{41}\ b_{44}\ b_{45}}^{row\ 4}\ \overbrace{b_{52}\ b_{55}}^{row\ 5}$$

$$JB = 1\ \ 3\ \ \ 1\ \ 2\ \ 3\ \ \ 3\ \ 1\ \ 4\ \ \ 5\ \ 2\ \ 5 \qquad (26)$$

$$IB = 1\ \ 3\ \ \ 6\ \ 7\ \ 10\ \ \ 12$$

2.4. التجارب العددية (Numerical Experiments):

تنفذ جميع الطرائق المدروسة والمقترحة الجديدة و هي طريقة سيمبليكس المعدلة (Revised Simplex Method) و طريقة LU كثيرة الأصفار المقترحة (New Direct LU) و طريقة Bartels-Golub و طريقة Bartels-Golub كثيرة الأصفار و طريقة Forrest-Tomlin و طريقة Reid و الطريقة التكرارية الجديدة New (Iterative Method) المحددة في هذه المقالة بلغة C++ النسخة 4.9 Devcpp على حاسوب بنتيوم 4 بمعالج Athlom 3500 و بذاكرة RAM 512 DDR.

بنية معطيات مصفوفة الأمثال كثيرة الأصفار A في التنفيذات الواقعية للطرائق المحددة مختلفة. في تنفيذاتنا لطريقة تحليل LU كثيرة الأصفار و طريقة Bartels-Golub كثيرة الأصفار تخزن مصفوفة الأمثال باستخدام القائمة المرتبة السطرية لنظام التخزين المقترح في هذه المقالة. تجري مقارناتنا الرئيسية لجميع الطرائق المذكورة على خمس مسائل اختبار من حجوم مختلفة اختيرت عشوائيا. يوضح الجدول (1) خواص مصفوفة الأمثال A في مسائل الاختبار المدروسة.

الجدول (1): يعطي المعلومات المتعلقة بالمسائل المدروسة المولدة عشوائيا المعدة لاختبار الطرائق المقترحة.

Problem	n-Row	m-Col	n-nz	Density
1	10	15	59	88.5%
2	20	30	220	36.67%
3	50	75	1300	34.67%
4	30	65	1065	54.62%
5	100	200	10286	51.43%

يمثل العمود الأول رقم المسألة المولدة عشوائيا، إذ إن هناك خمس مسائل اختبار. بينما يمثل العمودين الثاني والثالث عدد الأسطر n-Row و عدد الأعمدة m-Col في المصفوفة A الموافقة للبرنامج الخطي (1) على الترتيب. و أخيرا، يعطي العمود الرابع عدد العناصر اللاصفرية n-nz في المصفوفة A الموافقة للمسألة المدروسة. أما العمود الأخير فيعطي نسبة العناصر اللاصفرية المئوية في كل مصفوفة A من المسائل الخمس المدروسة.

الجدول (2): يوضح زمن تنفيذ (مأخوذا بالثواني) كل خوارزمية من الخوارزميات المقترحة: طريقة سيمبليكس المعدلة و طريقة LU المباشرة كثيرة الأصفار الجديدة و طريقة Bartels – Golub و طريقة Bartels – Golub كثيرة الأصفار و طريقة – Forrest Tomlin و طريقة Reid و طريقة تكرارية جديدة من أجل كل مسألة اختبار من المسائل المبينة في الجدول (1).

Problem	New Iterative	Reid	Forrest Tomlin	Sparse Bartels Golub	Bartels Golub $L^{-1}U^{-1}$	New Direct LU	Revised Simplex
1	0.0	0.67	0.001	0.047	0.0	0.0	0.0
2	0.0	1.65	0.031	1.594	0.31	0.01	0.015
3	0.015	*	0.484	13.5	0.312	0.015	0.016
4	0.015	13.088	0.109	12.078	0.172	0.001	0.015

5	0.076	*	4.047	*	8.87	0.078	2.593

من أجل كل مسألة تنفذ الطرائق باستخدام طريقة الجداء القياسي مبتدئين من أساس كثيف و محاولين إعادة القلب. يبين الجدول التالي العدد الكلي للعناصر اللاصفرية في مجالات التكرار كاملة.

الجدول (3): (نمو العناصر اللاصفرية في جميع الطرائق) يبين عدد العناصر اللاصفرية في كل تكرار من تكرارات الحلقة للحصول على الجدول الذي يعطي الحل المحسن.

Problem n-nz	New Iterative	Reid	Forrest Tomlin	Sparse Bartels Golub	Bartels Golub $L^{-1}U^{-1}$	New Direct LU	Revised Simplex
1	59	69 67	50	62 67 79	59	59	68
2	220 220	240 251 259 274	240 221	240 248 257 274	240 240	240 210	220 239
3	1300 1300	*	1350 1301	1350 1358 1365 1380	1350 1350 1350	1350 1301	1349 1396
4	1050 1050	1134 1191 1243 1300	1095 1066	1134 1181 1233 1290	1095	1065	1094 1123
5	10088 10088	*	10188 10089	*	10188 10188	10088 1088	10286 10305

2.5. نتائج و مناقشات (Results and Discussions)

تدون النتائج المحسوبة الناتجة باستخدام الطرائق المدروسة والمقترحة الجديدة في جداول. تدون تفاصيل نتائج المحاكاة الحاسوبية من تطبيق جميع الطرائق على خمس مسائل اختبار في الجدولين (2) و (3). تبين الأعمدة في الجدول (2) زمن التنفيذ المأخوذ بالثواني للحصول على حل البرنامج الخطي (1) التي تعطى مصفوفة أمثاله في الجدول (1) باستخدام جميع الطرائق. يبين العمود الأول في الجدول (3) رقم المسألة المدروسة بينما تبين جميع الأعمدة المتبقية من الجدول (3) عدد العناصر اللاصفرية التي تظهر في كل تكرار من تكرارات الحلقة في كل جدول يعطي تحسينا للحل.

يمكن أن نرى من الجدولين (2) و (3) ما يلي:

1. طريقة سيمبليكس المعدلة: تعمل هذه الطريقة على تغيير كل الجداول في كل تكرار. بمقارنة هذه الطريقة مع الطرائق الأخرى نجد أنها:

- أسرع من طريقة Bartels-Golub و طريقة Bartels-Golub كثيرة الأصفار و طريقة Forrest-Tomlin و طريقة Reid.
- تزيد من عدد التكرارات لإيجاد الحل الأمثل.
- يزداد عدد العناصر اللاصفرية بشكل طفيف في كل تكرار من تكرارات الحلقة.
- على الرغم من أن هذه الطريقة تعتمد مباشرة على مصفوفة الأساس والعمود الداخل (عمود التمحور) و على العمود الخارج إلا أنها مكلفة من جهة حساب معكوس مصفوفة الأساس و تزيد من عدد العناصر اللاصفرية بشكل ملحوظ كل تكرار مما يزيد من حجم الذاكرة لدى تنفيذ الخوارزمية حاسوبيا.

2. الطرائق الأخرى المدروسة: طريقة Bartels-Golub و طريقة Bartels-Golub كثيرة الأصفار و طريقة Forrest-Tomlin و طريقة Reid: تعتمد هذه الطرائق على تحليل مصفوفة الأساس إلى جداء مصفوفتين LU حيث L هي مصفوفة مثلثية سفلية و U هي مصفوفة مثلثية علوية ثم إيجاد معكوس هاتين المصفوفتين المثلثيتين. من حيث الزمن فإنها:
   - تتطلب زمنا كبيرا جدا إذ أنها أبطأ من خوارزمية سيمبليكس المعدلة.
   - تتوقف طريقة Bartels-Golub كثيرة الأصفار و طريقة Reid عن حل بعض المسائل كما هو الحال في المسألتين 3 و 5 التي أشرنا إليهما في الجدول (3) بالرمز (*). وهذا ناجم عن ظهور عدد كبير من العناصر اللاصفرية مما لم تقدر الذاكرة على الاتساع المتزايد لها.
   - لا نلحظ تخفيض في عدد العناصر اللاصفرية التي ترافق كل تكرار من تكرارات الحلقة.
   - أبطأ بحوالي 8−10 مرات من حيث سرعة التنفيذ من الطريقتين الجديدتين: طريقة تحليل LU كثيرة الأصفار و الطريقة التكرارية.
   - بالنتيجة لا تصلح لأن تكون عموما خوارزميات فعالة في حل مسائل البرمجة الخطية.
3. طريقة LU المباشرة كثيرة الأصفار المقترحة: هذه الطريقة هي إحدى الطرائق الجديدة المقترحة في هذه المقالة لحل مسألة البرمجة الخطية (1). إنها:
   - أسرع بحوالي 10 مرات من جميع الطرائق المدروسة.
   - نلاحظ تخفيضاً كبيراً في عدد العناصر اللاصفرية في كل تكرار من تكرارات الحلقة و خاصة في المسألة 5 كما هو واضح في الجدول (3). مما يظهر تفوقها أيضا على جميع الطرائق الأخرى.
   - بالنتيجة هي أسرع من جميع الطرائق و تخفض عدد العناصر اللاصفرية أي أنها فعالة جدا لذا ننصح باستخدامها في حل جميع مسائل البرمجة الخطية أو التطبيقات الواقعية التي تؤول إلى مسألة برمجة خطية.
4. الطريقة التكرارية: و هي طريقة جديدة اقترحناها أيضا في هذه المقالة. إنها:
   - توازي من حيث سرعة زمن التنفيذ طريقة LU الجديدة و أسرع من بقية الطرائق.
   - الأفضل من حيث حجم الذاكرة لأنه منذ بداية التكرار الأول تحافظ الطريقة على نفس العدد من العناصر اللاصفرية و هذا يقود بالتأكيد إلى تخفيض هائل في حجم الذاكرة المطلوبة لتنفيذ الخوارزمية.
   - بالنتيجة هي منافس قوي لطريقة LU الجديدة و أفضل من الطرائق المتبقية. لذا ننصح باستخدامها كخيار آخر لحل مسائل البرمجة الخطية.

أخيرا، يتضح من الجدولين (2) و (3) أيضا نتائج هامة للغاية و هي أن العناصر Fill-ins تؤثر على معدل نمو العناصر اللاصفرية و تزيد بالتالي كمية عمل معالجة المعطيات لدخول عدد من العناصر في الملف الذي يحوي المعطيات. كانت الأخطاء في جميع الطرائق بين 10^{-22} و 10^{-18} و بالتالي النتائج الحاصلة من التجارب المنفذة كافية لإعطاء تبصر واضح عن الطرائق المقترحة و غيرها.

2.6. توصيات (Recommendations):

درسنا في هذه المقالة العديد من الطرائق التي تعتمد على تحليل LU و هي طريقة سيمبليكس المعدلة و طريقة Bartels-Golub و طريقة Bartels-Golub كثيرة الأصفار و طريقة Forrest-Tomlin و طريقة Reid كما اقترحنا طريقتين جديدتين تعتمدان أيضا على تحليل LU دعوناهما طريقة LU المباشرة كثيرة الأصفار و الطريقة التكرارية لحل مسألة البرمجة الخطية و جميع التطبيقات الواقعية التي تؤول إلى مسألة برمجة خطية. تبين من نتائج المحاكاة الحاسوبية مايلي:

1. نوصي باعتماد الطريقتين الجديدتين في حل مسائل البرمجة الخطية لأنهما الأسرع من حيث الزمن و الأقل حجما في الذاكرة لعدم ظهور عناصر لاصفرية. لذا نوصي باعتمادهما في البرامج و المكتبات الحاسوبية المعدة لحل مسائل البرمجة الخطية لأنها تعطي نتائج ممتازة من وجهة نظر كثيرة الأصفار كونها تحافظ على العدد الأصلي للعناصر اللاصفرية.
2. طريقة سيمبليكس المعدلة أسرع و أقل حجما في الذاكرة من طريقة Bartels-Golub وطريقة Bartels-Golub كثيرة الأصفار و طريقة Forrest-Tomlin و طريقة Reid.
3. الطرائق Bartels-Golub وBartels-Golub كثيرة الأصفار و Forrest-Tomlin و طريقة Reid أبطأ من طريقة سيمبليكس بحوالي 10 مرات و تتطلب ذاكرة أكبر و أحيانا تتوقف بسبب ظهور عدد كبير من العناصر اللاصفرية.

المراجع:

[1] ROSE, D. J. *Sparse matrices and their applications*, 1st Edition, Plenum press, New York-London, 1972, 457.

[2] MARKOWITZ, H. M. *The elimination form of the inverse and its application to linear programming*, Management Sci. U. S. A., Vol. 3, No. 21, 1957, 255-269.

[3] DANTZIG, G. B. *Compact basis triangulation for the simplex method*, Recent Advances in Mathematical Programming, Graves, Vol. 22, No. 5, 1963, 125-132

[4] DANTZIG, G. B.; ORCHARD-HAYS, W. *The product form of inverse in the simplex method*, Math. Tables Aids Comput. U. S. A., Vol. 8, No. 9, 1954, 64-67.

[5] BARTELS, R. H.; GOLUB, G. H. *The simplex method of linear programming using LU decomposition*, Comm. ACM U. S. A., Vol. 12, No. 22, 1969, 266-268.

[6] BARTELS, R. H.; GOLUB, G. H.; SAUNDERS, M. A. *Numerical techniques in mathematical programming*, Non-linear Programming U. S. A., Vol. 32, No. 25, 1970, 123-176.

[7] GILL, P. E.; MURRAY, W. *A numerically stable form of the simplex algorithm*, J. Linear Algebra Appl. U. S. A., Vol. 5, No. 18, 1972, 224-2236.

[8] DANTZIG, B. *Linear Programming and Extensions*, 1st Edition, Princeton University Press, Princeton, 1963, 498.
[9] BRAYTON, R. K.; GUSTAFSON, F. G.; WILLOUGHBY, R. A. *Some results on sparse matrices*, Math. Comp. U. S. A., Vol. 24, No. 12, 1970, 937-954.
[10] FORREST, J. J. H.; TOMBLIN, J. A. *Updating the triangular factors of the basis to maintain sparsity in the product form simplex method*, Proceedings NATO Conference on Large Scale Mathematical Programming, Elsinor, Denmark, 1971, 658.
[11] TOMBLIN, J. A. *Modifying factors of the basis in the simplex method*, In *Sparse matrices and their applications*, Plenum press, New York-London, 1972, 77-85.
[12] REID, J. K. *Large Sparse Sets Of Linear Equations*, 1st Edition, Academic Press, London and New York, 1971, 284.