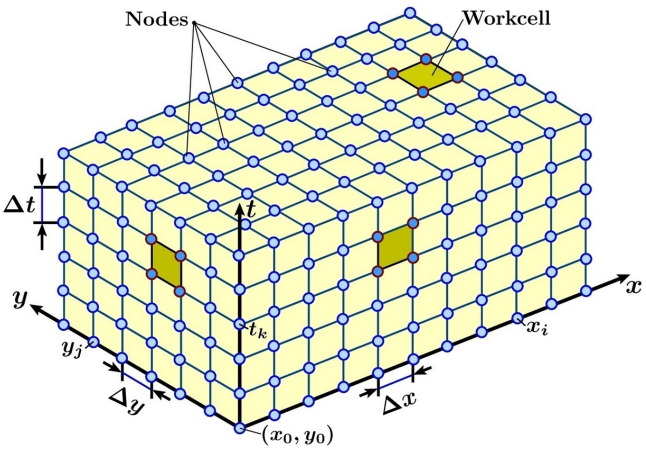
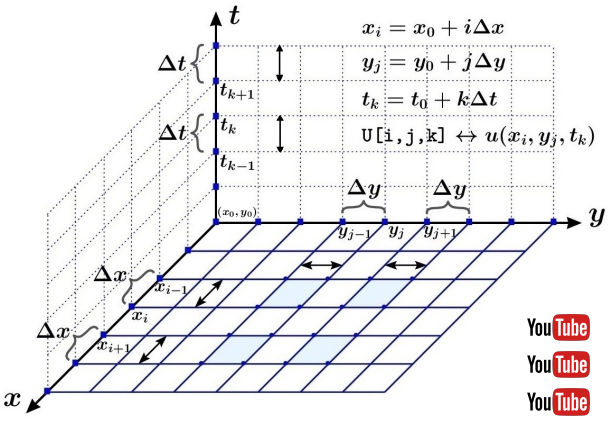
פרוייקט: שימוש בשיטת ההפרשים הסופיים לפתרון משוואות דיפרנציאליות

דניאל רבייב (danielr@technion.ac.il), סמי זעפרני (samyz@technion.ac.il)

09.10.2025.a





תוכן עניינים

1. מבוא כללי

בניגוד למקובל בענפי המתמטיקה הקלאסית המתבססים על מודלים **רציפים** של מרחב, זמן, מסה, חום, וגדלים פיזיקאליים אחרים, לשם ייצוג ופתרון בעיות מדעיות והנדסיות, שיטת ההפרשים הסופיים (Finite Difference Method) מתבססת על **מודלים בדידים (discrete)** בכדי לפתור מכלול רחב יותר של בעיות שלא ניתן לפתור בשיטות הקלאסיות. המודל הבדיד הכי פשוט הוא **שריג (grid)** כמו זה שרואים באיורים 1, 3, 4. מדובר במטריצה חד-מימדית או רב-מימדית של ערכים בדידים שמפוזרים בצורה אחידה בתחום הבעייה. שיטת ההפרשים הסופיים מתעלמת מהרצף האינסופי של המרחב והזמן שבו הבעייה טופלה קודם לכן, ובוחרת להתמקד במדגם קטן של ערכים בדידים מתחום הבעייה.

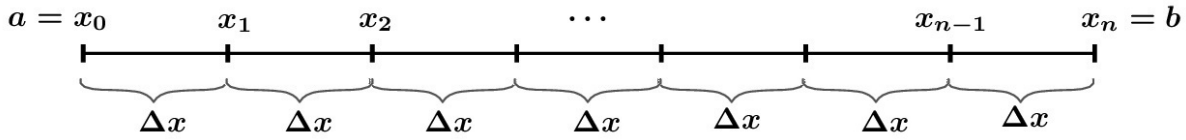
לדוגמה, בנושא משוואות דיפרנציאליות רגילות, המשתנה x יוגבל ל**שריג חד-מימדי** של ערכים בדידים x_i, $i = 0, 1, 2, \ldots, n$ (ראה איור 1). ה**שריג** (החד-מימדי) נוצר על ידי חלוקת קטע נתון $[a, b]$ על ציר-x ל-n תתי-קטעים שווים בגודלם באורך

$$\Delta x = \frac{b - a}{n}$$

נקודות החלוקה

$$x_i = a + i\Delta x, \quad i = 0, 1, 2, \ldots, n$$

נקראים **צמתים (nodes)** או **נקודות חלוקה**, והמספר הטבעי n נקרא **מספר החלוקה** בציר x.



איור 1: שריג חד-מימדי: חלוקה אחידה של קטע סופי ל-n תתי-קטעים ו-$n+1$ צמתים

נבחין כי חלוקת קטע ל-n תתי-קטעים יוצרת $n + 1$ צמתים. הצומת הראשונה מסומנת x_0 והאחרונה x_n. חקירת פונקציה $y = f(x)$, תתבצע על ידי חקירת הערכים $y_i = f(x_i)$

$$y_i = f(x_i), \quad i = 0, 1, 2, \ldots, n$$

בניגוד לערכי השריג x_i, ההפרשים $\Delta y = y_{i+1} - y_i$ אינם בהכרח אחידים (אלא אם הפונקציה $y = f(x)$ ליניארית). מושג גבולי כמו נגזרת מסדר ראשון נוכל להגדיר מחדש על ידי מנת **הפרשים** בדידים (ומכאן מקור השם "שיטת ההפרשים")

$$\frac{dy}{dx} = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} = \frac{y_{i+1} - y_i}{\Delta x}$$

בהנחה שחלוקת תחום ההגדרה של f מספיק צפופה, הביטוי $\frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$ עשוי להיות קירוב מספיק טוב לנגזרת "האמיתית" עבור מטרות פרקטיות.

לדוגמה, משוואה דיפרנציאלית רגילה כגון

$$\begin{cases} y' = 4y + 3x, & 0 < x < 2 \\ y(0) = -2 \end{cases} \tag{1}$$

תתורגם ל**משוואת הפרשים**

$$\begin{cases} \dfrac{y_{i+1} - y_i}{\Delta x} = 4y_i + 3x_i, & i = 0, 1, 2, \ldots, n-1 \\ y_0 = -2 \end{cases}$$

שניתן לפתור בקלות באמצעות **נוסחת רקורסיה** פשוטה

$$\begin{cases} y_{i+1} = y_i + (4y_i + 3x_i)\,\Delta x, & i = 0, 1, 2, \ldots, n-1 \\ y_0 = -2 \end{cases}$$

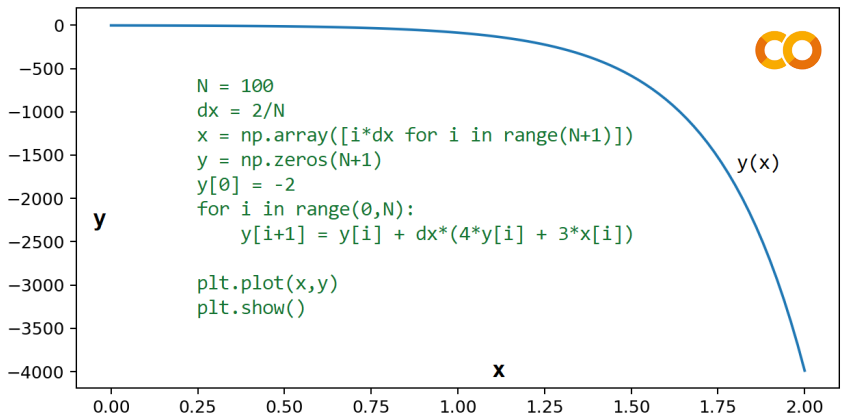
אם לדוגמה נחלק את הקטע $[0, 2]$ ל־$n = 100$ חלקים, אז $\Delta x = \frac{2}{100} = 0.02$, ולכן

$$\begin{cases} y_{i+1} = y_i + 0.02(4y_i + 3x_i), & i = 0, 1, 2, \ldots, 99 \\ y_0 = -2 \end{cases}$$

מה שמאפשר לנו למעשה לחשב את כל ערכי y_i בצורה רקורסיבית!

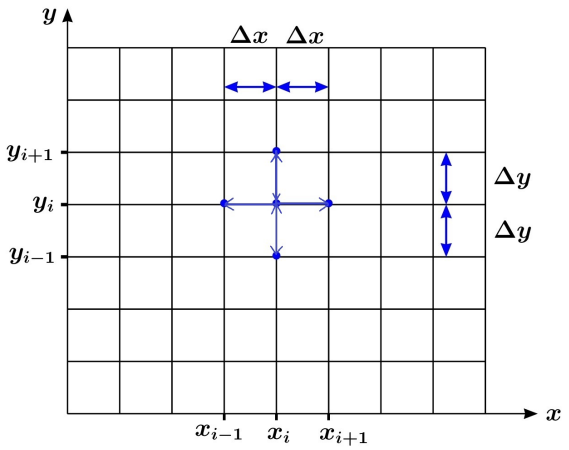
$$\begin{aligned}
y_0 &= -2 \\
y_1 &= y_0 + 0.02(4y_0 + 3x_0) = -2 + 0.02\,[4 \cdot (-2) + 3 \cdot 0.0] = -2.16 \\
y_2 &= y_1 + 0.02(4y_1 + 3x_1) = -2.08 + 0.02\,[4 \cdot (-2.08) + 3 \cdot 0.02] = -2.3316 \\
y_3 &= y_2 + 0.02(4y_2 + 3x_2) = -2.2452 + 0.02\,[4 \cdot (-2.2452) + 3 \cdot 0.04] = -2.515728 \\
&\vdots \quad \vdots \quad \vdots \qquad \vdots \qquad \vdots \\
y_{100} &= y_{99} + 0.02(4y_{99} + 3x_{99}) = -3988.7547771185805
\end{aligned}$$

מדובר ברשימה של 100 חישובים פשוטים שניתן לבצע בקלות באמצעות מחשב ולהציג על ידי גרף כמו זה שרואים באיור 2. גישה לקוד בלחיצה על קישור למחברת קולאב.

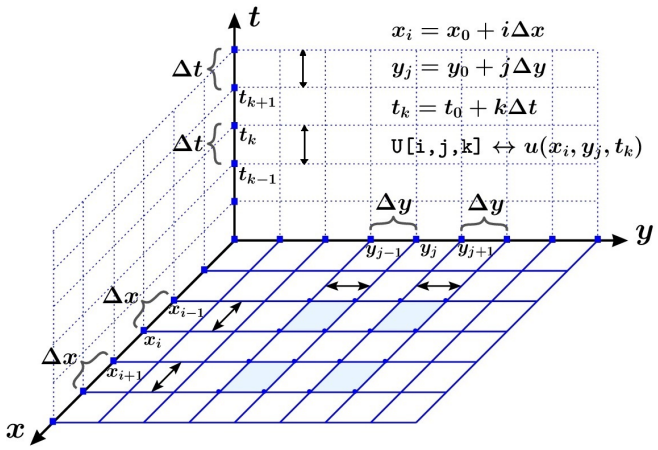


איור 2: פתרון בדיד של המשוואה (1) ברמת חלוקה $n = 100$

קיבלנו אם כן **פתרון בדיד** למשוואה הדיפרנציאלית שלנו! המתמטיקאי השוויצרי לאונרד אוילר (1707-1783) היה ככל הנראה המתמטיקאי הראשון שגילה את שיטת ההפרשים הסופיים, והראה כיצד ניתן לפתור באמצעותה משוואות דיפרנציאליות רגילות. פרטים נוספים בקישור למחברת הקולאב הזו.



איור 3: שריג דו-מימדי טיפוסי עבור שיטת ההפרשים הסופיים



איור 4: שריג תלת-מימדי עבור שיטת ההפרשים הסופיים

באופן דומה, חקירה של משוואות דיפרנציאליות חלקיות של פונקציה בשני משתנים תתבצע על פני שריג דו-מימדי כמו זה שרואים באיור 3, וחקירה של משוואות דיפרנציאליות חלקיות בשלושה משתנים תתבצע על שריג תלת-מימדי כמו זה שרואים באיור 4.

הסיבה העיקרית לפיתוח שיטת ההפרשים הסופיים היתה לאפשר פתרון של בעיות מתמטיות באמצעות אלגוריתמים נומריים וטכנולוגיות חישוב מודרניות שנולדו בסוף המאה הקודמת וממשיכות להתפתח בקצב מואץ בעידן הבינה המלאכותית וכוח המיחשוב ההולך וגדל. האלגוריתמים הנומריים מתבססים על תוצאות מתחומים כגון קומבינטוריקה, מתמטיקה בדידה, ותורת החישוביות, אשר גם הם התפתחו מאוד במאות האחרונות. השאיפה היא להשיג פתרונות קרובים ככל האפשר לפתרונות[1] "האמיתיים". אנו מוותרים על דיוק מוחלט אך מרוויחים יכולת לפתור מגוון רחב יותר של בעיות שלא ניתן לפתור באמצעות השיטות הקלאסיות.

באופן כללי, בכדי לחקור פונקציות של n משתנים נזדקק לשריג הפרשים n-מימדי. למשל בכדי למצוא פתרונות נומריים למשוואת החום $u_t = \alpha(u_{xx} + u_{yy})$ הדו-מימדית, יש לחקור פונקציות של שלושה משתנים $u(x,y,t)$, ולכן נזדקק לשריג הפרשים תלת-מימדי שמוצג בצורה סכימטית באיור 4.

[1] המרכאות הן בגלל שהיום פיזיקאים רבים מאמינים שהמבנה הפיזיקאלי של המרחב, הזמן, מסה, אנרגיה, וכו', הוא בדיד, ושמושגי הרצף האינסופי של המתמטיקה הקלאסית הן למעשה הפשטה תאורטית שאינה משקפת את המציאות הפיזיקאלית! קישור למאמר בנושא.

1.1 הגדרה פורמלית של שריג הפרשים

לכל זוג מספרים שלמים m, n, נגדיר את **קבוצת הטווח (range)** $[m:n]$ על ידי

$$[m:n] = \{\, i \in \mathbb{Z} :\ m \le i < n \,\}$$

כאשר $\mathbb{Z}$ היא קבוצת המספרים השלמים. הסימון $[m:n]$ לקוח מעולם התוכנה, ונועד לסייע לנו לגלוש לקטע התיכנותי בצורה נוחה. נדגיש כי קבוצת הטווח $[m:n]$ אינה כוללת את אינדקס הקצה n. כלומר, מדובר בסדרת המספרים השלמים שמתחילה ב־m ומסתיימת במספר $n-1$. לשם נוחות, נתמקד בשריגים דו-מימדיים ותלת-מימדיים בלבד.

נתחיל עם תאור של שריג תלת-מימדי. יהיו N_x, N_y, N_t **מספרי החלוקה** בכיווני הצירים x, y, t, בהתאמה. מבחינה פורמלית, שריג הפרשים תלת-מימדי מוגדר באמצעות מטריצה תלת-מימדית $\mathbf{U}$

$$\mathbf{U}:\ \Big([0:N_x+1]\times[0:N_y+1]\times[0:N_t+1]\Big) \longrightarrow \mathbb{R}$$

המכפלה הקרטזית $[0:N_x+1]\times[0:N_y+1]\times[0:N_t+1]$ נקראת **תחום השריג**. זהו אוסף סופי של כל **המפתחות** (i,j,k) עבור הצמתים (x_i,y_j,t_k) כפי שרואים באיור 4. לכל **מפתח** (i,j,k) בתחום השריג מתאימה **צומת** (x_i,y_j,t_k) בתחום ההגדרה הבדיד של הפונקציה $u(x,y,t)$, ולכן הערך של השריג $\mathbf{U}$ במפתח (i,j,k) מוגדר להיות

$$\mathbf{U}[i,j,k] = u(x_i,y_j,t_k)$$

כאשר $u(x,y,t)$ היא פונקציית המטרה של הבעייה שצריך לפתור (שלרוב אינה ידועה לנו מלבד על שפת התחום). חשוב להבחין בין **מפתח** (וקטור של אינדקסים) (i,j,k) לבין **הצומת** (x_i,y_j,t_k) המתאימה לו! המפתח (i,j,k) היא שלשה של **מספרים שלמים** שמציינת את המיקום המדויק של הצומת (x_i,y_j,t_k) בשריג. במילים אחרות, המפתח (i,j,k) היא **כתובת** של הצומת (x_i,y_j,t_k) בשריג. בדומה למקובל בסביבות תיכנות מתמטיות, נרחיב את השימוש בביטויי טווח (range) באופן הבא. הביטוי $[m:]$ יסמן את הטווח של כל האינדקסים החל מ־m ועד לאינדקס האחרון באותו ציר. הביטוי $[:n]$ יסמן את הטווח $[0:n]$, והביטוי $[:]$ יסמן את הטווח המלא מ־0 ועד לאינדקס האחרון באותו ציר.

כשנטפל בקוד תוכנה, הסימונים האלה יאפשרו לנו לאתחל את השריג על קבוצות גדולות של אינדקסים. למשל, בשפות התיכנות פייתון או מטלב (Matlab), הביטוי $\mathbf{U}[:,:,0]=3$ מאתחל את ערכי $\mathbf{U}$ על כל המפתחות $(i,j,0)$ ל־3, והביטוי $\mathbf{U}[3,:,0]=7$ יאתחל את השורה השלישית של פרוסה 0 לערך 7. כלומר $\mathbf{U}[3,j,0]=7$ לכל אינדקס j בציר השני $[0:N_y+1]$.

1.2 נגזרות חלקיות בדידות

בהרצאות הקורס מד"ח (עיין גם בחלק IV של חוברת ההרצאות) נלמד כי ניתן לקרב את הנגזרת החלקית באמצעות שלוש נוסחאות שונות

$$u_x(x_i, y_j, t_k) = \frac{u(x_{i+1}, y_j, t_k) - u(x_i, y_j, t_k)}{\Delta x} \tag{2}$$

$$u_x(x_i, y_j, t_k) = \frac{u(x_i, y_j, t_k) - u(x_{i-1}, y_j, t_k)}{\Delta x} \tag{3}$$

$$u_x(x_i, y_j, t_k) = \frac{u(x_{i+1}, y_j, t_k) - u(x_{i-1}, y_j, t_k)}{2\Delta x} \tag{4}$$

הנוסחה הראשונה נקראת **נוסחת הפרש קדמי (forward difference)**, השניה **הפרש אחורי**, והשלישית **הפרש מרכזי (central difference)**. המבנה היותר קומפקטי של שריג ההפרשים מאפשר לנו לרשום את הנוסחאות האלה בצורה מעט יותר קצרה

$$\mathrm{U}_x[i, j, k] = \frac{\mathrm{U}[i+1, j, k] - \mathrm{U}[i, j, k]}{\Delta x} \tag{5}$$

$$\mathrm{U}_x[i, j, k] = \frac{\mathrm{U}[i, j, k] - \mathrm{U}[i-1, j, k]}{\Delta x} \tag{6}$$

$$\mathrm{U}_x[i, j, k] = \frac{\mathrm{U}[i+1, j, k] - \mathrm{U}[i-1, j, k]}{2\Delta x} \tag{7}$$

הביטוי U_x שבאגף שמאל הוא למעשה ייצוג של מושג גזירה חלקית של שריג הפרשים שניתן להגדרה פורמלית על ידי כל אחד מהביטויים שבאגף ימין. לפעמים פעולה מסוג זה נקראת **גזירה בדידה** והיא נלמדת ונחקרת באחד מענפי המתמטיקה הבדידה שנקרא חשבון דיפרנציאלי בדיד (Discrete Calculus). בסביבות תוכנה כגון MATLAB או NumPy קיימות ספריות תוכנה עבור פעולות גזירה מסוג זה.

באותו אופן ניתן לקבל נוסחאות דומות עבור הנגזרות החלקיות U_y, U_t, שלא נטרח לרשום אותן כאן, אך נשתמש בהן בהמשך.

במהלך הקורס (ראה גם חוברת הרצאות בקורס), נוכיח קיום של נוסחאות דומות גם עבור הנגזרות החלקיות מסדר שני.

$$u_{xx}(x_i, y_j, t_k) = \frac{u(x_{i+1}, y_j, t_k) - 2u(x_i, y_j, t_k) + u(x_{i-1}, y_j, t_k)}{\Delta x^2} \tag{8}$$

$$u_{xx}(x_i, y_j, t_k) = \frac{u(x_i, y_j, t_k) - 2u(x_{i-1}, y_j, t_k) + u(x_{i-2}, y_j, t_k)}{\Delta x^2} \tag{9}$$

$$u_{xx}(x_i,y_j,t_k)=\frac{u(x_{i+2},y_j,t_k)-2u(x_{i+1},y_j,t_k)+u(x_i,y_j,t_k)}{\Delta x^2} \tag{10}$$

שניתנות להבעה במונחי שריג הפרשים באופן דומה

$$\mathrm{U}_{xx}[i,j,k]=\frac{\mathrm{U}[i+1,j,k]-2\mathrm{U}[i,j,k]+\mathrm{U}[i-1,j,k]}{\Delta x^2} \tag{11}$$

$$\mathrm{U}_{xx}[i,j,k]=\frac{\mathrm{U}[i,j,k]-2\mathrm{U}[i-1,j,k]+\mathrm{U}[i-2,j,k]}{\Delta x^2} \tag{12}$$

$$\mathrm{U}_{xx}[i,j,k]=\frac{\mathrm{U}[i+2,j,k]-2\mathrm{U}[i+1,j,k]+\mathrm{U}[i,j,k]}{\Delta x^2} \tag{13}$$

נכנה את נוסחה (11) **נוסחת הפרש מרכזי**, נוסחה (12) **נוסחת הפרש אחורי**, נוסחה (13) **נוסחת הפרש קדמי**. ניתן לרשום נוסחאות דומות עבור U_{yy}, U_{tt}, U_{xy}, וכדומה. נציג למשל חישוב סימבולי עבור הנוסחה הקדמית עבור הנגזרת המעורבת u_{xy}

$$\begin{aligned}
u_{xy}(x,y,t)&=\frac{\partial}{\partial y}\left[u_x(x,y,t)\right]=\frac{\partial}{\partial y}\left[\frac{u(x+\Delta x,y,t)-u(x,y,t)}{\Delta x}\right]\\
&=\frac{\frac{\partial}{\partial y}u(x+\Delta x,y,t)-\frac{\partial}{\partial y}u(x,y,t)}{\Delta x}\\
&=\frac{\frac{u(x+\Delta x,y+\Delta y,t)-u(x+\Delta x,y,t)}{\Delta y}-\frac{u(x,y+\Delta y,t)-u(x,y,t)}{\Delta y}}{\Delta x}\\
&=\frac{u(x+\Delta x,y+\Delta y,t)-u(x+\Delta x,y,t)-u(x,y+\Delta y,t)+u(x,y,t)}{\Delta x\Delta y}
\end{aligned}$$

נוכל לרשום נוסחה דומה עבור נקודות על השריג שלנו

$$u_{xy}(x_i,y_j,t_k)=\frac{u(x_{i+1},y_{j+1},t_k)-u(x_{i+1},y_j,t_k)-u(x_i,y_{j+1},t_k)+u(x_i,y_j,t_k)}{\Delta x\Delta y} \tag{14}$$

שניתן להביע במונחי שריג הפרשים באופן הבא

$$\mathrm{U}_{xy}[i,j,k]=\frac{\mathrm{U}[i+1,j+1,k]-\mathrm{U}[i+1,j,k]-\mathrm{U}[i,j+1,k]+\mathrm{U}[i,j,k]}{\Delta x\Delta y} \tag{15}$$

למען הדיוק נדגיש כי הנוסחאות (8)-(15), אינם שוויונים מדויקים במובן הרגיל של הנגזרות החלקיות כפי שהוגדרו בקורסי החדו"א הקלאסיים! אלה הם קירובים בלבד לנגזרות הקלאסיות. אך מאחר וויתרנו מראש על דיוק מוחלט, אנו חופשיים להגדיר אותן מחדש למטרות פיתוח של שיטות נומריות שמטרתן להשיג קירוב טוב בלבד.

בכל הדוגמאות הנ"ל התמקדנו בפונקציה של שלושה משתנים. ניתן כמובן לרשום נוסחאות

דומות עבור פונקציות עם מספר שונה של משתנים. אנו מניחים שהרעיון ברור ולכן לא נטרח לציין זאת במקרים כאלה.

לסיום סעיף זה נדגיש כי בהקשר לפונקציות, שריג הפרשים מתיחס לתחום ההגדרה של הפונקציה בלבד, ולא לערכים שלה. למרות שגם ערכי הפונקציה הם ערכים בדידים, הם אינם חלק מהשריג.

1.3 פתרון מערכות בדידות

בחוברת ההרצאות של הקורס מד"ח (עיין גם בחלק IV, פרק 8) מוצגות מספר דוגמאות לשיטת הפתרון של מד"ח באמצעות שיטת ההפרשים הסופיים. נסכם את תהליך הפתרון בארבעת השלבים הבאים. לשם נוחות, נניח כי השריג שלנו הוא דו-מימדי.

א. בנו שריג הפרשים U מעל תחום ההגדרה של הבעייה הנתונה, באמצעות בחירת מספרי חלוקה בכל מימד של תחום הבעייה.

ב. המירו את המשוואה הדיפרנציאלית למשוואת הפרשים על השריג U, על ידי החלפת הנגזרות החלקיות בנוסחאות הפרשים מתאימות.

ג. המירו את תנאי השפה של הבעייה באמצעות תנאים שקולים על שריג ההפרשים, והוסיפו את התנאים שקיבלתם למערכת המשוואות מהשלב הקודם.

ד. וודאו שקיבלתם מערכת משוואות פתירה. במרבית המקרים, המד"ח שלנו ליניארית, ולכן נקבל מערכת של משוואות ליניאריות בנעלמים $\mathrm{U}[i,j]$ (בהנחה שהשריג שלנו דו-מימדי). יש לוודא שמספר המשוואות גדול או שווה למספר הנעלמים ושהמערכת עקבית ופתירה באופן יחידי.

דוגמה 1: לשם הבהרה נציג דוגמה פשוטה להבנת עקרונות השיטה. נחשב באופן ידני פתרון בדיד מהצורה 4×4 עבור מד"ח ליניארית מסדר ראשון

$$\begin{cases} u_x + u_y = 0, & 0 < x < 3,\ 0 < y < 3 \\ u(x,0) = x, & 0 \le x \le 3 \\ u(0,y) = 2y, & 0 \le y \le 3 \\ u(x,3) = 6 - 2x, & 0 \le x \le 3 \\ u(3,y) = 3 - y, & 0 \le y \le 3 \end{cases} \tag{16}$$

נתאר את התהליך על פי השלבים הנ"ל:

א. בניית שריג: נבחר חלוקה אחידה $N = 3$ בשני הצירים של התחום $[0,3]\times[0,3]$. יחידות השריג שלנו יהיו $\Delta x = \Delta y = \frac{3}{N} = 1$, ולכן במקרה זה נקודות החלוקה יהיו מספרים שלמים $x_i = i$, $y_j = j$. נבחין כי מספר נקודות החלוקה בכל ציר הוא $N + 1 = 4$! לכן

השריג שלנו $\mathbf{U}$ הוא שריג דו-מימדי מהצורה $(N+1)\times(N+1)=4\times4$, אשר יכיל 16 צמתים. נשתמש בנוסחאות הפרשים קדמיות עבור שתי הנגזרות החלקיות

$$\mathbf{U}_x[i,j]=\frac{\mathbf{U}[i+1,j]-\mathbf{U}[i,j]}{\boldsymbol{\Delta x}} \tag{17}$$

$$\mathbf{U}_y[i,j]=\frac{\mathbf{U}[i,j+1]-\mathbf{U}[i,j]}{\boldsymbol{\Delta y}} \tag{18}$$

ב. **המרת המד"ח למשוואת הפרשים:** נחליף את הנגזרות החלקיות u_x , u_y, בנוסחאות ההפרשים

$$\boldsymbol{U}_x[i,j]+\boldsymbol{U}_y[i,j]=\frac{\mathbf{U}[i+1,j]-\mathbf{U}[i,j]}{\boldsymbol{\Delta x}}+\frac{\mathbf{U}[i,j+1]-\mathbf{U}[i,j]}{\boldsymbol{\Delta y}}=0$$

מאחר ובחרנו לעבוד עם שריג אחיד שבו $\boldsymbol{\Delta x}=\boldsymbol{\Delta y}=1$, נקבל

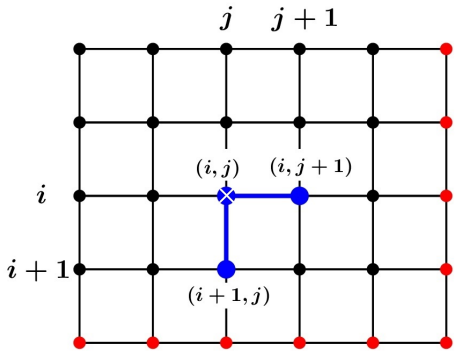
$$\mathbf{U}[i+1,j]-\mathbf{U}[i,j]+\mathbf{U}[i,j+1]-\mathbf{U}[i,j]=0$$

ולאחר פישוט

$$\begin{aligned}&\mathbf{U}[i+1,j]-2\mathbf{U}[i,j]+\mathbf{U}[i,j+1]=0\\&i=1,2\quad j=1,2\end{aligned} \tag{19}$$

חשוב מאוד להבחין בהגבלה של ערכי האינדקסים i, j, ל**צמתים פנימיים**. במקרה התאורטי השכיח, המד"ח מוגבלת לנקודות פנימיות של התחום מאחר שלא תמיד מובטח לנו שהנגזרות החלקיות קיימות על שפת התחום. במקרה הבדיד זה עשוי להתבטא בכך שהצבת צמתי שפה במשוואות ההפרשים אינה תמיד אפשרית. במקרה הנוכחי למשל, כאשר $i=3$, הביטוי $i+1$ **גולש מחוץ לתחום השריג**, ולכן $\mathbf{U}[i+1,j]$ לא מוגדר, ולכן הנגזרת הקדמית $\boldsymbol{U}_x[i,j]$ אינה מוגדרת!

בבעיות מסוג זה מומלץ לשרטט **סטנסיל** של המשוואה שמאפיין את הצמתים שניתן להציב בה. באיור 5 רואים שהמשוואה (19) אינה חלה על צמתי השפה האדומים, מאחר והצבת **מרכז הסטנסיל** (i,j) בנקודות אלה תגרום לו לגלוש מחוץ לשריג.



איור 5: סטנסיל עבור משוואת ההפרשים (19)

לכן במקרים רבים שנפגוש בחוברת הזו תירגום של מד"ח לנוסחת הפרשים יחול על צמתים פנימיים בלבד. במקרה הנוכחי למשל, רואים מהסטנסיל שניתן להפעיל את משוואות

ההפרשים על צמתי השפה בהן $i=0$ או $j=0$, אך יש לנו כבר תנאי שפה נתונים עליהם[2], ולכן לא כללנו את $i,j=0$ בתחום משוואה (19). בכדי לפתור את המערכת, נרשום את השוויון (19) בצורה יותר נוחה

$$\mathbf{U}[i,j+1]=2\mathbf{U}[i,j]-\mathbf{U}[i+1,j], \qquad i=1,2 \quad j=1,2 \tag{20}$$

זה נותן לנו 4 משוואות ליניאריות שונות בנעלמים $\mathbf{U}[i,j]$ (הצמתים הפנימיים). נזכיר שוב כי הנעלמים שלנו הם כל ערכי השריג $\mathbf{U}[i,j]$ עבור $i=0,1,2,3$, $j=0,1,2,3$. יש לנו 16 צמתים ולכן יש לנו 16 נעלמים לפתור! 4 משוואות לא יספיקו לכך.

ג. **תנאי שפה:** את 12 המשוואות החסרות אנו אמורים לקבל מתנאי השפה.

$$\begin{cases} \mathbf{U}[i,0]=i, & i=0,1,2,3 \\ \mathbf{U}[i,3]=6-2i, & i=0,1,2,3 \\ \mathbf{U}[0,j]=2j, & j=0,1,2,3 \\ \mathbf{U}[3,j]=3-j, & j=0,1,2,3 \end{cases}$$

מאחר ובחרנו מספר חלוקה קטן $N=3$, נוכל לרשום פירוט בדיד מלא של תנאי השפה

$\mathbf{U}[0,0]=0$	$\mathbf{U}[0,1]=2$	$\mathbf{U}[0,2]=4$	$\mathbf{U}[0,3]=6$
$\mathbf{U}[1,0]=1$			$\mathbf{U}[1,3]=4$
$\mathbf{U}[2,0]=2$			$\mathbf{U}[2,3]=2$
$\mathbf{U}[3,0]=3$	$\mathbf{U}[3,1]=2$	$\mathbf{U}[3,2]=1$	$\mathbf{U}[3,3]=0$

נבחין כי בפינות של התחום קיימת חפיפה בין תנאי השפה, ולכן יש 12 במקום 16. יוצא שתנאי השפה מספקים לנו למעשה פתרונות עבור 12 מתוך 16 הנעלמים של המערכת! אלה כל **צמתי השפה**. זהו כמובן מקרה חריג בגלל גודל החלוקה הקטן שלנו. אם למשל $N=10$, אז תהיה לנו מערכת של 121 משוואות, 40 צמתי שפה, ו־81 צמתים פנימיים. נותר לנו אם כן לפתור 4 נעלמים עבור הצמתים הפנימיים של השריג:

$\mathbf{U}[0,0]=0$	$\mathbf{U}[0,1]=2$	$\mathbf{U}[0,2]=4$	$\mathbf{U}[0,3]=6$
$\mathbf{U}[1,0]=1$	$\mathbf{U}[1,1]=?$	$\mathbf{U}[1,2]=?$	$\mathbf{U}[1,3]=4$
$\mathbf{U}[2,0]=2$	$\mathbf{U}[2,1]=?$	$\mathbf{U}[2,2]=?$	$\mathbf{U}[2,3]=2$
$\mathbf{U}[3,0]=3$	$\mathbf{U}[3,1]=2$	$\mathbf{U}[3,2]=1$	$\mathbf{U}[3,3]=0$

באופן כללי, בכל שריג דו-מימדי עם חלוקה אחידה N, יש $(N-1)^2$ צמתים פנימיים, ו־$4N$ צמתי שפה.

ד. **פתרון מערכת המשוואות:** כאשר מספר החלוקה גדול, למשל $N=999$, נקבל מערכת של

[2] למעשה ישנם מצבים, גם בשימושים בתעשייה, שבהן יש סתירה בין תנאי השפה עצמם (בפינות של השפה למשל). במקרים כאלה, בעייה נפתרת ע"י כך שהתנאים האחרונים גוברים על הקודמים. בשלב התיכנות זה מתבטא בכך שהנתונים האחרונים "דורסים" את הקודמים.

מיליון משוואות במיליון נעלמים! הדרך היחידה לפתור מערכות מסוג זה היא באמצעות אלגוריתמים נומריים שעל חלקם נלמד בפרקים הבאים. בדוגמה הנוכחית בחרנו מספר חלוקה קטן $N=3$ ומד"ח פשוטה מאוד בכדי להעביר לתלמיד רושם מוחשי על איך השיטה עובדת. חלוקה כזו כמובן אינה מניבה רמת דיוק ריאלית, ובפרקים הבאים נפגוש חלוקות יותר גדולות לשם השגת רמת דיוק פרקטית.

את הנעלמים הפנימיים נפתור בעזרת הנוסחה (20) מלמעלה

$$\begin{aligned}
\mathrm{U}[1,1] &= 2\mathrm{U}[1,0] - \mathrm{U}[2,0] = 2\cdot 1 - 2 = 0 \\
\mathrm{U}[2,1] &= 2\mathrm{U}[2,0] - \mathrm{U}[3,0] = 2\cdot 2 - 3 = 1 \\
\mathrm{U}[1,2] &= 2\mathrm{U}[1,1] - \mathrm{U}[2,1] = 2\cdot 0 - 1 = -1 \\
\mathrm{U}[2,2] &= 2\mathrm{U}[2,1] - \mathrm{U}[3,1] = 2\cdot 1 - 2 = 0
\end{aligned}$$

ובכך תמה משימת הפתרון הבדיד מסדר 4×4

$\mathrm{U}[0,0]=0$	$\mathrm{U}[0,1]=2$	$\mathrm{U}[0,2]=4$	$\mathrm{U}[0,3]=6$
$\mathrm{U}[1,0]=1$	$\mathrm{U}[1,1]=0$	$\mathrm{U}[1,2]=-1$	$\mathrm{U}[1,3]=4$
$\mathrm{U}[2,0]=2$	$\mathrm{U}[2,1]=1$	$\mathrm{U}[2,2]=0$	$\mathrm{U}[2,3]=2$
$\mathrm{U}[3,0]=3$	$\mathrm{U}[3,1]=2$	$\mathrm{U}[3,2]=1$	$\mathrm{U}[3,3]=0$

דוגמאות נוספות מסוג זה ניתן למצוא בהמשך החוברת בתרגילים 1, 4, 10, 11. האימון על שריגים קטנים מסייע מאוד בהבנת השיטה ולכן מומלץ לפתור כל אחד מהם.

סיכום: באופן כללי, בשריג דו-מימדי אחיד בגודל חלוקה N, שיטת ההפרשים הסופיים מובילה למערכת של $(N+1)\times(N+1)$ משוואות ליניאריות ב־$(N+1)\times(N+1)$ נעלמים. הנעלמים של המערכת הם ערכי השריג $\mathrm{U}[i,j]$, $i,j=0,1,2,\ldots,N$. המשוואות יתקבלו ממשוואת ההפרשים של המד"ח ותנאי השפה. תנאי השפה של המד"ח יספקו פתרונות מיידיים לחלק או לכל צמתי שפת השריג. נציין כי במד"ח מסדר גדול מ־1, תנאי השפה עשויים להשפיע גם על צמתים פנימיים! במקרים השכיחים מדובר במערכת משוואות דלילה שבה רוב המקדמים שמחוץ לאלכסון מתאפסים, ולכן ניתנת לפתרון באמצעות תימרונים אלגבריים פשוטים. אחת השיטות הנפוצות שנפגוש בהמשך החוברת היא פתרון באמצעות נוסחאות נסיגה.

בפרקים הבאים נפגוש גם שריגים ממימד 3, שבהן החלוקה אינה תמיד אחידה, ושפתרון המערכות שלהן דורש יותר יצירתיות וכישורי תיכנות.

2. משוואת החום (1D/2D)

משוואת החום היא משוואה דיפרנציאלית חלקית שמהווה מודל מתמטי לתופעת התפשטות החום בפיזיקה של חומרים. ניתן לחקור את התופעה על מוט מתכת חד-מימדי (1D-Rod), לוחית מתכת דו-מימדית (2D-plate), או על גופים תלת-מימדיים בעלי נפח (כגון תיבה, כדור, או גליל). שיטות הפתרון שנציג כאן חלות על כל המקרים, אך לשם פשטות נדגים את שיטת ההפרשים הסופיים מעל דוגמה של מוט חד-מימדי, ודוגמה שניה מעל לוחית מלבנית דו-מימדית.

בהמשך נפגוש דוגמאות נוספות מתחום הגלים והתחום הפיננסי.

לפני שניגש לדוגמה של שריג גדול ומעשי, נתאמן שוב על דוגמה פשוטה של שריג 4×4 שניתן לפתור באופן ידני. עיינו שוב בדוגמה 1 מסוף הפרק הקודם ובצעו שוב את ארבעת השלבים המתוארים שם כהכנה להמשך הפרק.

תרגיל 1: (משוואת חום עם צמיחה/דעיכה - 1D heat equation with growth/decay)
מצאו פתרון בדיד מהצורה 4×4 בדומה לזה שהוצג בדוגמה 1.

$$\begin{cases} u_t = 4u_{xx} - u, & 0 < x < 3,\ \ 0 < t < 3 \\ u(0,t) = 0, & 0 \le t \le 3 \\ u(3,t) = 0, & 0 \le t \le 3 \\ u(x,0) = x(3-x), & 0 \le x \le 3 \end{cases}$$

הדרכה: הצורה הכללית משוואת החום הקלאסית היא $u_t = \alpha u_{xx}$. אם נוסיף את הביטוי βu לאגף ימין, נקבל את הצורה הכללית של משוואת חום עם מקדם צמיחה/דעיכה β: $u_t = \alpha u_{xx} + \beta u$. עיינו שוב בדוגמה 1 מסוף הפרק הקודם ובנו מודל בדיד מגודל 4×4 עבור משוואה זו. יש לחלק את ציר-x וציר הזמן t ל־$N=3$ חלקים, ולבנות טבלה 4×4 של ערכי השריג. מומלץ להשתמש בנוסחת נגזרת מרכזית עבור u_{xx}, ונוסחת נגזרת קדמית עבור u_t.

בדוגמה הבאה נבנה מודל בדיד כללי שצורתו $(N+1)\times(N_t+1)$ עבור משוואת החום החד-מימדית הקלאסית $u_t = \alpha u_{xx}$, כאשר N מספר חלוקה עבור אורך המוט $[0,L]$, ו־N_t מספר חלוקה של קטע הזמן $[0,T]$. בחירה מתאימה של הפרמטרים N, N_t, תאפשר לנו לקבל שריג שבאמצעותו נוכל לחשב את הטמפרטורה של המוט בנקודה x, בזמן t, ברמת דיוק מעשית.

דוגמה 2: על ידי שימוש בהפרשים סופיים, נפתור בעיית חום חד-מימדית (1D) טיפוסית מעל מוט באורך $L=1$, מעל קטע זמן באורך $T=0.2$ שניות.

$$\begin{cases} u_t = u_{xx}, & 0 < x < 1,\ \ 0 < t < 0.2 \\ u(0,t) = e^{-\pi^2 t}, & 0 \le t \le 0.2 \\ u(1,t) = -e^{-\pi^2 t}, & 0 \le t \le 0.2 \\ u(x,0) = \cos(\pi x), & 0 \le x \le 1 \end{cases}$$

הפתרון המדויק לבעייה הוא $u(x,t) = e^{-\pi^2 t}\cos(\pi x)$. ניתן כמובן לבדוק זאת ידנית על ידי הצבה. נוכל להשתמש בפתרון זה בכדי לאמת את הפתרון הנומרי המקורב שנקבל על ידי שיטת ההפרשים הסופיים (אנו עושים זאת במחברת הקולאב המצורפת בהמשך).

נבחר מספר חלוקה N עבור הקטע $[0,1]$, ומספר חלוקה N_t עבור קטע הזמן $[0,0.2]$. נגדיר הפרמטרים היסודיים שלנו

$$\Delta x = \frac{1}{N}, \quad \Delta t = \frac{0.2}{N_t}, \quad x_i = i\Delta x, \quad t_k = k\Delta t, \quad \mathrm{U}[i,k] = u(x_i,t_k)$$

נשתמש בנוסחאות גזירה בדידה הבאות

$$\mathbf{U}_{xx}[i,k] = \frac{\mathbf{U}[i+1,k] - 2\mathbf{U}[i,k] + \mathbf{U}[i-1,k]}{\boldsymbol{\Delta x^2}} \tag{21}$$

$$\mathbf{U}_{t}[i,k] = \frac{\mathbf{U}[i,k+1] - \mathbf{U}[i,k]}{\boldsymbol{\Delta t}} \tag{22}$$

בחרנו בנוסחת הפרש מרכזי עבור הנגזרת השנייה $\mathbf{U}_{xx}$ ובנוסחת הפרש קדמי עבור הנגזרת $\mathbf{U}_t$. נתרגם את בעיית החום שלנו לבעייה בדידה על שריג ההפרשים $\mathbf{U}$.

$$\begin{cases} \mathbf{U}_t[i,k] = \mathbf{U}_{xx}[i,k], & 0 < i < N,\ \ 0 < k < N_t \\ \mathbf{U}[0,k] = e^{-\pi^2 t_k}, & 0 \le k \le N_t \\ \mathbf{U}[N,k] = -e^{-\pi^2 t_k}, & 0 \le k \le N_t \\ \mathbf{U}[i,0] = \cos(\pi x_i), & 0 \le i \le N \end{cases}$$

באופן כללי, שיטת ההפרשים הסופיים מובילה למערכת של $(N+1)\times(N_t+1)$ משוואות ליניאריות ב־$(N+1)\times(N_t+1)$ נעלמים. הנעלמים של המערכת הם ערכי השריג $\mathbf{U}[i,k]$, $i = 0,1,2,\ldots,N$, $k = 0,1,2,\ldots,N_t$. ברוב המקרים מדובר במערכת משוואות דלילה שבה רוב המקדמים שמחוץ לאלכסון מתאפסים, ולכן במקרים רבים ניתנת לפתרון באמצעות תימרונים פשוטים כגון נוסחאות נסיגה כפי שנראה בדוגמה זו ובהמשך. מנוסחה (22) נובע

$$\mathbf{U}[i,k+1] = \mathbf{U}[i,k] + \mathbf{U}_t[i,k]\boldsymbol{\Delta t}$$

מהשוויון $\mathbf{U}_t[i,k] = \mathbf{U}_{xx}[i,k]$ נקבל

$$\mathbf{U}[i,k+1] = \mathbf{U}[i,k] + \mathbf{U}_{xx}[i,k]\boldsymbol{\Delta t}$$

זוהי למעשה נוסחת נסיגה[3] פשוטה שמאפשרת לחשב פתרון

$$\boxed{\mathbf{U}[i,k+1] = \mathbf{U}[i,k] + \mathbf{U}_{xx}[i,k]\boldsymbol{\Delta t}}$$

נוסחת הנסיגה מאפשרת לנו לחשב את פיזור החום על המוט בנקודת הזמן $t = t_{k+1}$ מפיזור החום בזמן $t = t_k$. במילים אחרות: אם ידועים לנו הערכים $\mathbf{U}[i,k]$, $0 \le i \le N$, אז נוסחת הנסיגה מאפשרת לנו לגלות את הנעלמים $\mathbf{U}[i,k+1]$ (עבור $0 < i < N$). מאחר והנעלמים $\mathbf{U}[i,0]$ נתונים לנו על יד תנאי השפה, נוכל לחשב את כל ערכי השריג $\mathbf{U}[i,k]$! זהו מהלך טיפוסי להרבה בעיות מסוג זה, ובהמשך נראה עוד כמה דוגמאות שלו.

אבל בנוסחת הנסיגה הסיפור לא נגמר! בכדי להבטיח התכנסות של הפתרון הבדיד לפתרון האמיתי כאשר $N, N_t \to \infty$, יש לוודא שיחידות השריג Δx, Δt, עומדים בתנאי יציבות

[3] יש לוודא שהביטוי $k+1$ אינו מתחבא באגף הימני! אחרת זו לא נוסחת נסיגה! יש להציץ גם בנוסחה (21) ולוודא זאת.

מתאימים. במקרה של משוואת החום החד-מימדית התנאי הוא $\Delta t \leq \frac{\Delta x^2}{2\alpha}$, כאשר α הוא מקדם מוליכות החום (קבוע פיזיקאלי) שמופיע במשוואה $u_t = \alpha u_{xx}$. במקרה הנוכחי $\alpha = 1$, ולכן תנאי היציבות שלנו הוא

$$\Delta t \leq \frac{\Delta x^2}{2}$$

או באופן שקול

$$\gamma = \frac{\Delta t}{\Delta x^2} \leq 0.5$$

מקובל להשתמש בפרמטר γ בבעיות מסוג זה. בהמשך נפגוש תנאי יציבות דומים בבעיות דו-מימדיות של חום וגלים. דיון כללי בנושאי דיוק ויציבות מוצג בפרק 6. הסבר ספציפי לתנאי היציבות עבור משוואת החום ניתן למצוא במקור [8]. קוד פייתון מלא שמממש את הפתרון הזה נמצא במחברת הקולאב המקושרת לאיור 6, שבו מוצג הקטע העיקרי שלו.

```
# Boundary conditions
for i in range(0, N+1):
    U[i,0] = cos(pi * i*dx)          # u(x,0) = cos (\pi x)

for k in range(0, Nt+1):
    U[0,k] = exp(-pi**2 * k * dt)    # u(0,t) = e^{-\pi^2 t}
    U[N,k] = -exp(-pi**2 * k * dt)   # u(1,t) = -e^{-\pi^2 t}

def Solve(U):
  for k in range(0,Nt):
    for i in range(1,N):
      Uxx = (U[i+1,k] - 2*U[i,k] + U[i-1,k]) / (dx**2)
      U[i,k+1] = U[i,k] + Uxx * dt
```

איור 6: קוד פייתון עבור בעיית החום החד-מימדית של דוגמה 2

בתרגיל הבא נחזור שוב למשוואת החום עם צמיחה/דעיכה שפגשנו בתרגיל 1, הפעם נבקש לבנות שריג בגודל כללי $N \times N_t$ שיוכל לשמש למטרה פרקטית.

תרגיל 2: (משוואת חום עם צמיחה/דעיכה - 1D heat equation with growth/decay)

פתרו את הבעייה הבאה בשיטת ההפרשים הסופיים.

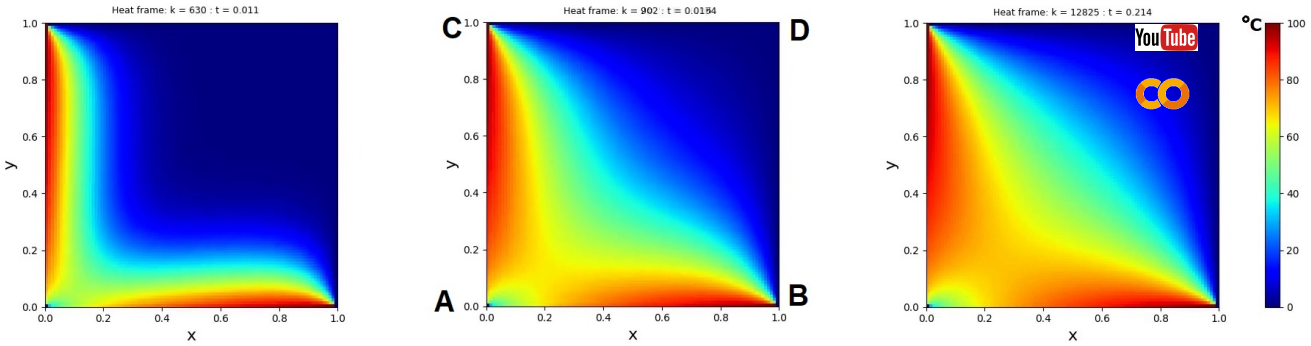
$$\begin{cases} u_t = 4u_{xx} - u, & 0 < x < 2, \;\; 0 < t < 1 \\ u(0,t) = 0, & 0 \leq t \leq 1 \\ u(2,t) = 0, & 0 \leq t \leq 1 \\ u(x,0) = x(2-x)\sin(3\pi x), & 0 \leq x \leq 2 \end{cases}$$

הדרכה: הצורה הכללית של משוואת חום עם מקדם צמיחה/דעיכה β היא: $u_t = \alpha u_{xx} + \beta u$. יש למצוא נוסחת נסיגה עבור מספרי חלוקה N, N_t, כמו בדוגמה הקודמת. פגשנו את המשוואה הזו בתרגיל 1, לכן תוכלו להשתמש בנוסחת הנסיגה שקיבלתם שם. מומלץ להעזר בהעתק של מחברת הקולאב מהדוגמה הקודמת בכדי לבנות קוד פייתון מתאים בכדי לאמת את נוסחת הנסיגה שקיבלתם.

דוגמה 3: תהי D לוחית מלבנית דו-מימדית העשויה מחומר אחיד בצפיפות אחידה. עבור כל נקודה (x,y) על הלוחית D, $u(x,y,t)$ היא הטמפרטורה בנקודה (x,y) בזמן t. הקורס מטפל במקרה החד-מימדי (נא לעיין בחוברת ההרצאות בנוגע להנחות העבודה במקרה הזה). המקרה הדו-מימדי דומה: הפונקציה u גזירה פעמיים ברציפות במשתנים x, y, בנקודות פנימיות של התחום D, גזירה ברציפות על מימד הזמן t, ומקיימת את המשוואה הדיפרנציאלית החלקית הבאה

$$u_t = \alpha(u_{xx} + u_{yy}), \qquad (x,y) \in D, \quad 0 < t < \infty$$

כאשר α הוא קבוע מוליכות ממשי חיובי של החומר ממנו הלוחית עשויה. באיור 7 רואים את מצב החום על פני לוחית מלבנית בשלוש נקודות זמן שונות.



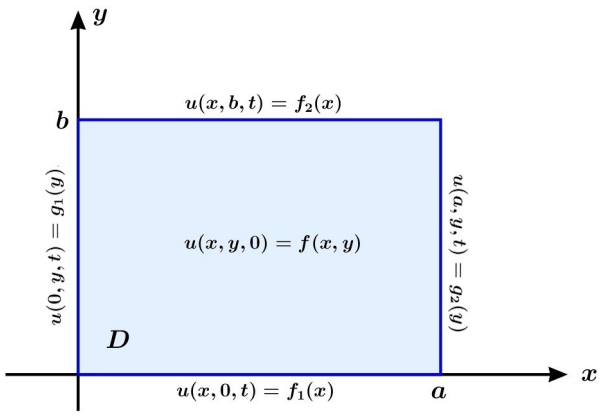
איור 7: מפות חום על לוחית מלבנית בשלוש נקודות זמן (מפת הצבעים מימין היא בין $0°$ ל־ $100°$ צלזיוס)

כמו במקרה החד-מימדי, קיימים שלושה סוגים של תנאי שפה: דיריכלה, נוימן, ותערובת שלהם. בדוגמאות שלנו נצטמצם לתנאי דיריכלה שבהן נקודות השפה מוחזקות בטמפרטורה קבועה (כפונקציה של הנקודה). בדוגמה של איור 7 תנאי השפה נתונים על ידי

$$\text{(23)} \qquad \begin{cases} u(x,0,t) = f_1(x) = 100\sqrt[4]{x} & 0<x<1,\ 0<y<1,\ 0<t<\infty \\ u(x,1,t) = f_2(x) = 0 & 0<x<1,\ 0<y<1,\ 0<t<\infty \\ u(0,y,t) = g_1(y) = 100\sqrt[8]{y} & 0<x<1,\ 0<y<1,\ 0<t<\infty \\ u(0,y,t) = g_2(y) = 0 & 0<x<1,\ 0<y<1,\ 0<t<\infty \\ u(x,y,0) = f(x,y) = 0 & 0<x<1,\ 0<y<1 \end{cases}$$

נציין שבארבעת התנאים הראשונים, הטמפרטורות של נקודות השפה מוחזקות בצורה קבועה לאורך כל הזמן t! שינוי טמפרטורה מתרחש רק על פני נקודות פנימיות של הלוחית. התנאי החמישי מציין את הטמפרטורה ההתחלתית ($t=0$) של הנקודות הפנימיות של הלוחית. נציין שבזמן $t=0$ יש חוסר תאימות בין הטמפרטורות שעל שפת הלוחית (שהן חיוביות על הצלעות AB, AC) לטמפרטורות בנקודות הפנימיות (שהן אפס). תופעה זו מכונה לפעמים "אמבט חום". ניתן להבחין כיצד החום מתפשט משתי הצלעות AB, AC, לכיוון הפינה D.

באיור 8 מוצג תאור כללי של משוואת החום מעל לוחית מלבנית $D=[0,a]\times[0,b]$. תנאי השפה הנלווים הם תנאי דיריכלה. כאמור זה הסוג היחיד שבו נטפל בפרוייקט הנוכחי. דוגמאות לתנאי נוימן וסוג מעורב ניתן למצוא בספרים כגון [18], [17], [20].

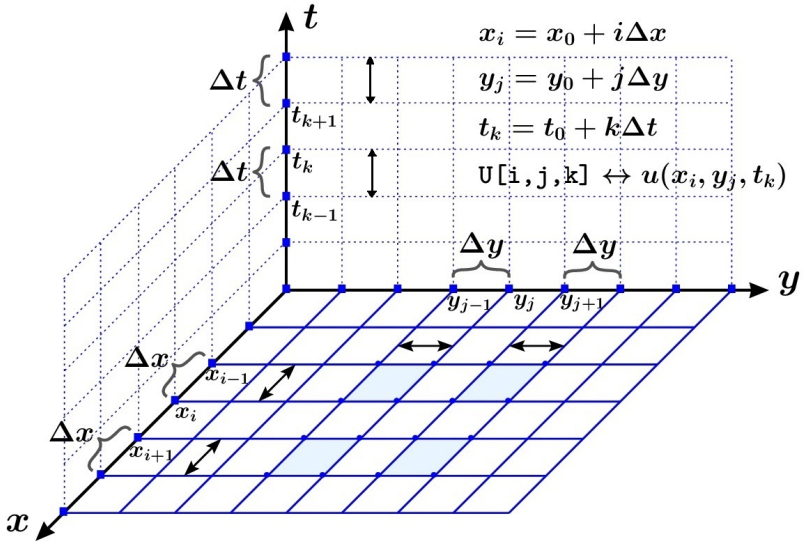


$$\begin{cases} u_t = \alpha(u_{xx}+u_{yy}), & 0<x<a,\ 0<y<b,\ 0<t<\infty \\ u(x,y,0)=f(x,y), & 0<x<a,\ 0<y<b \\ u(x,0,t)=f_1(x), & 0\le x\le a,\ 0\le t<\infty \\ u(x,b,t)=f_2(x), & 0\le x\le a,\ 0\le t<\infty \\ u(0,y,t)=g_1(y), & 0\le y\le b,\ 0\le t<\infty \\ u(a,y,t)=g_2(y), & 0\le y\le b,\ 0\le t<\infty \end{cases}$$

איור 8: משוואת החום עם תנאי שפה מסוג דיריכלה מעל לוחית דו-מימדית

קיימים תנאים על הפונקציות f, f_1, f_2, g_1, g_2, הדרושים לשם הבטחת קיום ויחידות של פתרון למשוואת החום שלנו, שלא נפרט כאן. ניתן למצוא אותם במקורות הנ"ל.

בכדי לטפל במד"ח בשלושה משתנים יש לעבוד מעל **שריג הפרשים תלת-מימדי**. ראה איור 9.



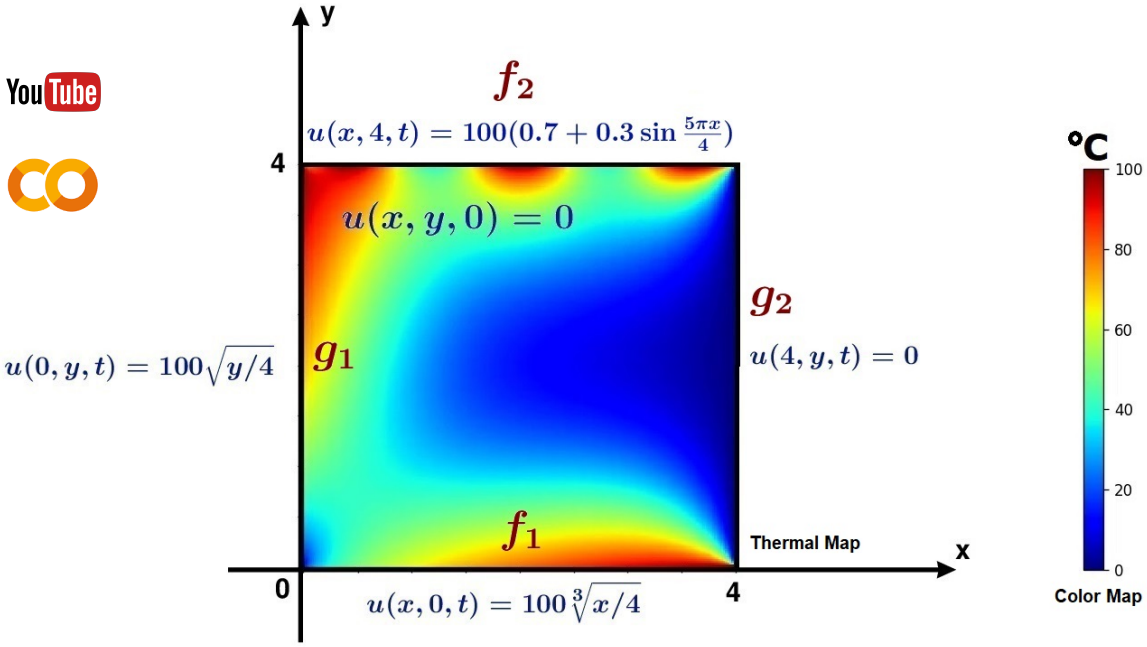
איור 9: שריג תלת-מימדי עבור משוואת החום במשתנים x, y, t.

דוגמה 4: לשם המחשת השיטה, נפתור בעזרתה את משוואת החום הדו-מימדית מעל הריבוע $D=[0,4]\times[0,4]$ (כלומר $a=b=4$), בקטע הזמן $[0,T]$.

$$\begin{cases} u_t = 0.1(u_{xx}+u_{yy}), & 0<x<4,\ 0<y<4,\ 0<t<T \\ u(x,y,0)=0, & 0<x<4,\ 0<y<4 \\ u(x,0,t)=100\sqrt[3]{x/4}, & 0\le x\le 4,\ 0\le t\le T \\ u(x,4,t)=100(0.7+0.3\sin\frac{5\pi x}{4}), & 0\le x\le 4,\ 0\le t\le T \\ u(0,y,t)=100\sqrt{y/4}, & 0\le y\le 4,\ 0\le t\le T \\ u(4,y,t)=0, & 0\le y\le 4,\ 0\le t\le T \end{cases} \tag{24}$$

תנאי השפה מתוארים באיור 10. הטמפרטורה על הצלע הימנית היא אפס כפי שזה בולט על פי

הגוון הכהה של הכחול בקרבתה. האיור כולל גם שני קישורים לסרטון יוטיוב ומחברת קולאב עבור דוגמה זו (בצד ימין למטה)[4].



איור 10: תנאי שפה עבור בעייה (24). מפת חום בזמן $t = 5.910$.

פתרון הבעייה זו בשיטה הרגילה אפשרי אך מסובך למדי עקב הצורך בפיתוח טורי פוריה של $\sqrt{x}$ או $\sqrt[3]{y}$ שהיא משימה די מייגעת. בשיטת ההפרשים הסופיים, לעומת זאת, פונקציות מהסוג הזה אינן מהוות מכשול.

לשם פשטות, נשתמש בגודל חלוקה N זהה עבור הקטע $[0,4]$ בציר-x וגם בציר-y. לכן $\Delta x = \Delta y = \frac{4}{N}$. מספר החלוקה של ציר הזמן $[0,T]$ יקבע בדרך כלל על פי מספר התמונות הדרושות בכדי לייצר סרטון סימולציה[5]. בינתיים נסמן אותו על ידי N_t, ולכן $\Delta t = \frac{T}{N_t}$. בנוסף לכך, קיומו של פתרון בדיד ברמת קירוב רצויה תלוי גם בבחירה מתאימה של מספרי החלוקה N, N_t, שנדונים ביתר הרחבה בפרק 6, ובמאמרים [16], [19], אשר נשתמש בהן בדוגמאות שלנו, אך לא נדון בהן.

בהנחה שבחירת גודלי החלוקה עומדת בתנאים הדרושים, נוכל להניח כי שריג ההפרשים $\mathbf{U}$ מתכנס לפתרון מקורב u במובן מסוים: $\mathbf{U}[i,j,k] \sim u(i\Delta x, j\Delta y, k\Delta t)$. סוג הקירוב עשוי להיות במובן של התכנסות נקודתית של $\mathbf{U}[i,j,k]$ לערך $u(i\Delta x, j\Delta y, k\Delta t)$, או התכנסות במידה שווה, ולפעמים התכנסות במובן אחר.

לשם כך יש לתרגם את המשוואה ואת תנאי השפה שלנו לתנאים שקולים על שריג ההפרשים

[4] יש חוסר תאימות של תנאי השפה בנקודות $(0,4,t)$, $(4,4,t)$ אשר עשוי להוות בעייה עבור קיום פתרון מדויק קלאסי, אך אינו מהווה בעייה עבור קיום פתרון מקורב בשיטת ההפרשים הסופיים! מבחינת האלגוריתם, הסדר שבו תנאי השפה נקלטים לחישוב קובע את הערך הסופי של u על נקודות האלה. ראה סעיף 6.2.

[5] אבל לפעמים יש חובה לבחור מספר חלוקה גדול לקבלת דיוק טוב, ואז סרטון הסימולציה ישתמש רק בדגימה של התמונות.

$\mathbf{U}$. נתחיל עם נוסחאות הגזירה שפגשנו בסעיף 1.2

$$\mathbf{U}_t[i,j,k] = \frac{\mathbf{U}[i,j,k+1] - \mathbf{U}[i,j,k]}{\boldsymbol{\Delta t}}$$

$$\mathbf{U}_{xx}[i,j,k] = \frac{\mathbf{U}[i+1,j,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i-1,j,k]}{\boldsymbol{\Delta x^2}}$$

$$\mathbf{U}_{yy}[i,j,k] = \frac{\mathbf{U}[i,j+1,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i,j-1,k]}{\boldsymbol{\Delta y^2}}$$

הבחירה בין שלושת האפשרויות (נוסחה אחורית, מרכזית, או קדמית) היא קריטית עבור פתרון הבעייה. בחרנו בנוסחה קדמית עבור $\mathbf{U}_t$, ונוסחה מרכזית עבור $\mathbf{U}_{xx}$, $\mathbf{U}_{yy}$, מסיבה שתתברר בהמשך. נציג את משוואת החום כתלות בפונקציה הבדידה $\mathbf{U}$ (ונזכיר שוב כי $\boldsymbol{\Delta y = \Delta x}$)

$$\mathbf{U}_t = \boldsymbol{\alpha}\left(\mathbf{U}_{xx} + \mathbf{U}_{yy}\right) \tag{25}$$

נבטא את $\mathbf{U}_{xx} + \mathbf{U}_{yy}$ על ידי נוסחת הפרשים

$$\mathbf{U}_{xx}[i,j,k] + \mathbf{U}_{yy}[i,j,k] =$$

$$= \frac{\mathbf{U}[i+1,j,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i-1,j,k]}{\boldsymbol{\Delta x^2}} + \frac{\mathbf{U}[i,j+1,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i,j-1,k]}{\boldsymbol{\Delta x^2}}$$

$$= \frac{1}{\boldsymbol{\Delta x^2}}\left(\mathbf{U}[i-1,j,k] + \mathbf{U}[i+1,j,k] + \mathbf{U}[i,j-1,k] + \mathbf{U}[i,j+1,k] - 4\mathbf{U}[i,j,k]\right)$$

הביטוי שבתוך הסוגריים בשורה האחרונה

$$\boxed{\boldsymbol{\Delta}\mathbf{U} = \mathbf{U}[i-1,j,k] + \mathbf{U}[i+1,j,k] + \mathbf{U}[i,j-1,k] + \mathbf{U}[i,j+1,k] - 4\mathbf{U}[i,j,k]} \tag{26}$$

נקרא אופרטור לפלס בדיד (Discrete Laplace Operator) ובהקשרים אחרים הוא גם מכונה סטנסיל (Stencil). זוהי תבנית גאומטרית שנהוג להציג בתרשים כמו זה שרואים באיור 11 שמבטא דפוס גרפי של האופרטור. קיבלנו את הנוסחה הבאה

$$\boxed{\mathbf{U}_{xx}[i,j,k] + \mathbf{U}_{yy}[i,j,k] = \frac{\boldsymbol{\Delta}\mathbf{U}}{\boldsymbol{\Delta x^2}}} \tag{27}$$

נוסחה זו שימושית גם עבור משוואת לפלס ומשוואת הגלים, ולכן ראויה לציון. נציב זאת במשוואת החום הבדידה $\mathbf{U}_t = \alpha(\mathbf{U}_{xx} + \mathbf{U}_{yy})$ (25) ונקבל

$$\mathbf{U}_t[i,j,k] = \frac{\mathbf{U}[i,j,k+1] - \mathbf{U}[i,j,k]}{\boldsymbol{\Delta t}} = \frac{\boldsymbol{\alpha}\boldsymbol{\Delta}\mathbf{U}}{\boldsymbol{\Delta x^2}}$$

לאחר חילוץ פשוט נקבל **נוסחת נסיגה** עבור $\mathbf{U}$ במימד הזמן k

$$\mathbf{U}[i,j,k+1] = \mathbf{U}[i,j,k] + \frac{\alpha \Delta t}{\Delta x^2}\,\mathbf{\Delta U}$$

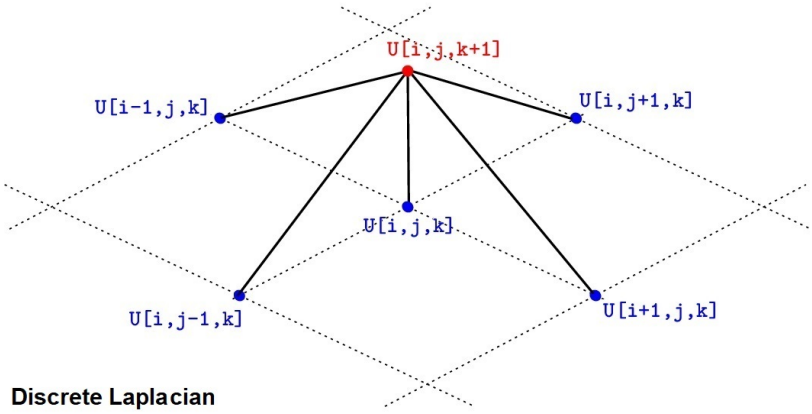
מקובל לסמן

$$\gamma = \frac{\alpha \Delta t}{\Delta x^2}$$

לכן נוסחת הנסיגה שלנו הופכת ליותר קצרה וקריאה

$$\mathbf{U}[i,j,k+1] = \mathbf{U}[i,j,k] + \gamma \mathbf{\Delta U} \tag{28}$$

המשמעות של נוסחה זו היא שבהינתן מפת חום בנקודת זמן $t = t_k$, נוכל לחשב את מפת החום בנקודת הזמן הבאה $t = t_{k+1}$. מאחר ומפת החום בזמן $t_0 = 0$ נתונה לנו כחלק מתנאי השפה, נוכל לחשב את כל מפות החום שבאות לאחר מכן ברקורסיה!



איור 11: סטנסיל (Stencil) עבור נוסחת הנסיגה לפתרון משוואת החום (קוד פייתון)

אך יש לדייק יותר: הנוסחה (28) תקפה רק עבור **מפתחות פנימיים** (i,j): $0 < i,j < N$, מאחר והביטויים $i \pm 1$, $j \pm 1$ מופיעים ב־$\mathbf{\Delta U}$. לכן בכדי לקבל פתרון יחיד נזדקק לאילוצים הנובעים מתנאי השפה. באיור 11 רואים את התלות של נקודה על מפת חום $k+1$ בנקודות ממפת חום הקודמת k. מאיור זה רואים כי רק נקודות פנימיות של המפה $k+1$ ניתן לקבל מהמפה k. נקודות השפה יטופלו על ידי תנאי השפה.

נעבור עכשיו לתרגום תנאי השפה לתנאים שקולים על שריג ההפרשים $\mathbf{U}$. למשל, את התנאי $u(x,y,0) = 0$ יש לרשום בצורה הבאה

$$\mathbf{U}[i,j,0] = 0, \quad 0 < i < N, \;\; 0 < j < N$$

נציין שוב כי אינדקסי הקצה $i,j = 0$ או $i,j = N$ אינם כלולים באילוץ הזה. בסביבות תוכנה

כגון MATLAB או NumPy תנאי זה נכתב על ידי שורת קוד פשוטה וקצרה

```
U[1:N,1:N,0] = 0
```

באופן דומה, תנאי השפה $u(x,0,t)=f_1(x)$,$u(x,L,t)=f_2(x)$ יתורגמו לתנאי שריג

$$\begin{aligned} \mathbf{U}[i,0,k] &= f_1(x_i), \quad 0 \le i \le N, \;\; 0 \le k \le N_t \\ \mathbf{U}[i,N,k] &= f_2(x_i), \quad 0 \le i \le N, \;\; 0 \le k \le N_t \end{aligned}$$

תנאי השפה $u(0,y,t)=g_1(x)$,$u(L,y,t)=g_2(x)$ יתורגמו לתנאי שריג

$$\begin{aligned} \mathbf{U}[0,j,k] &= g_1(y_j), \quad 0 \le j \le N, \;\; 0 \le k \le N_t \\ \mathbf{U}[N,j,k] &= g_2(y_j), \quad 0 \le j \le N, \;\; 0 \le k \le N_t \end{aligned}$$

כאשר $x_i = i\Delta x$,$y_j = j\Delta y$.

לאחר איתחול השריג $\mathbf{U}$ בתנאי השפה, יש להפעיל את נוסחת הנסיגה (28) עבור כל k בתחום $k = 1, 2, \ldots, N_t - 1$. באיור 12 ניתן לראות כיצד זה מתבצע באמצעות 5 שורות קוד Python (חבילת NumPy). בכתיבת הקוד התבססנו על הבלוג המצוין [3] של Nervadof בנושא זה, ואנו מודים לו על כך. קוד פייתון מלא לפתרון הבדיד של הבעייה שלנו ניתן לראות במחברת הקולאב heat.ipynb המקושרת לאיור 12.

```
def Solve(U):
  for k in range(0, num_frames-1):
    for i in range(1, N-1):
      for j in range(1, N-1):
        DU = U[i+1,j,k] + U[i-1,j,k] + U[i,j+1,k] + U[i,j-1,k] - 4*U[i,j,k]
        U[i,j,k+1] = U[i,j,k] + gamma * DU
```

איור 12: סימולציה של משוואת החום באמצעות קוד NumPy

הביטוי **DU** שמופיע בתחתית איור 11, ואיור 12, הוא הקוד עבור הלפלסיאן הבדיד שלנו, $\mathbf{\Delta U}$ שמופיע בכמה הקשרים דומים כגירסה בדידה של הלפלסיאן הרציף $u_{xx}+u_{yy}$. נפגוש אותו שוב במשוואת הגלים.

איתחול המערך התלת-מימדי U מצריך היכרות ומיומנות בשימוש במערכים רב-מימדיים בסביבות תיכנות מתמטיות כמו MATLAB או NumPy. מאחר ומדובר בקורס מתמטי, יצרנו מימשק יותר נוח שבאמצעותו ניתן להגדיר את תנאי השפה ללא נסיון תיכנותי קודם בסביבות אלה. תלמידים בעלי נסיון כזה יוכלו כמובן להוריד את חבילת הקוד שלנו ולהתעמק בפרטים. באיור 13 רואים כיצד מתבצע האיתחול של תנאי השפה בדוגמה שלנו.

בשורה הראשונה מגדירים את הפונקציה $f(x,y)$ המשמשת לאיתחול $u(x,y,0)$. בארבעת השורות הבאות מגדירים את הפונקציות **f1**, **f2**, **g1**, **g2** עבור תנאי השפה. האיתחול עצמו מתבצע בשורה האחרונה באמצעות הפונקציה **boundary_conditions** שמטפלת

```
f  = lambda x,y: 0
f1 = lambda x: 100 * (x/L) ** (1/3)
f2 = lambda x: 100 * (0.7 + 0.3*sin(5*pi*x/L))
g1 = lambda y: 100 * (y/L)**(1/2)
g2 = lambda y: 0
boundary_conditions(U, f, f1, f2, g1, g2, dx, dy)
```

איור 13: תנאי שפה עבור משוואת החום באמצעות קוד NumPy

בכל הפרטים הטכניים המפותלים הדרושים לאיתחול המערך U. הפקודה **lambda** היא פקודת Python להגדרת פונקציה מיידית בשורה אחת. פונקציות יותר מסובכות ניתן לבנות באמצעות המנגנון הרגיל של **def**.

3. דוגמאות נוספות

לאחר שלמדנו את השיטה באמצעות שתי דוגמאות של משוואת חום, ננסה להפעיל אותה על דוגמאות נוספות של משוואות דיפרנציאליות חלקיות. נפתח עם דוגמה פשוטה של משוואה ליניארית מסדר ראשון, ונשאיר לתלמיד שני תרגילים נוספים לשם לימוד עצמי.

דוגמה 5: מצאו פתרון בדיד $u(x,y)$ בצורת נוסחת נסיגה מעל שריג הפרשים מתאים, עבור המשוואה הדיפרנציאלית החלקית הבאה (סדר ראשון) מעל ריבוע היחידה $[0,1]\times[0,1]$.

$$\begin{cases} xu_x + (1+y)u_y = 0, & 0 < x < 1,\ 0 < y < 1 \\ u(x,0) = 2x, & 0 \le x \le 1 \\ u(0,y) = 0, & 0 \le y \le 1 \\ u(x,1) = x, & 0 \le x \le 1 \\ u(1,y) = \frac{2}{1+y}, & 0 \le y \le 1 \end{cases} \tag{29}$$

פתרון: זוהי מד"ח ליניארית מסדר ראשון אשר ניתן לפתור גם באמצעות השיטות הרגילות שנלמדות בתחילת הקורס. הפתרון הסגור הוא $u(x,y) = \frac{2x}{1+y}$, ולכן ניתן להשתמש בו בכדי לאמת את הפתרון הנומרי שנשיג באמצעות שיטת ההפרשים הסופיים. יהי N מספר חלוקה משותף לשני הצירים של התחום הריבועי שלנו $D = [0,1]\times[0,1]$. נסמן $\Delta x = \Delta y = \frac{1}{N}$. ננסה להשתמש בנוסחה אחורית עבור הנגזרת לפי y, ובנוסחה קדמית עבור הנגזרת לפי y:

$$\mathrm{U}_x[i,j] = \frac{\mathrm{U}[i,j] - \mathrm{U}[i-1,j]}{\Delta x}$$

$$\mathrm{U}_y[i,j] = \frac{\mathrm{U}[i,j+1] - \mathrm{U}[i,j]}{\Delta y}$$

נזכיר כי $x = i\Delta x$, $y = j\Delta y$, ונציב במשוואה $xu_x + (1+y)u_y = 0$

$$x\mathrm{U}_x + (1+y)\mathrm{U}_y = i\Delta x\mathrm{U}_x[i,j] + (1+j\Delta y)\mathrm{U}_y[i,j] = 0$$

קיבלנו משוואת הפרשים

$$i\Delta x\mathrm{U}_x[i,j]+(1+j\Delta y)\mathrm{U}_y[i,j]=0$$

במשוואה מעורבים הנגזרות הבדידות U_x , U_y, ובמקרה זה יש צורך להשתמש בנוסחאות המתאימות בכדי לחלץ ממנה נוסחת נסיגה:

$$\begin{aligned}
&i\Delta x\mathrm{U}_x[i,j]+(1+j\Delta y)\mathrm{U}_y[i,j]=0\\
\Rightarrow\quad &i\Delta x\cdot\frac{\mathrm{U}[i,j]-\mathrm{U}[i-1,j]}{\Delta x}+(1+j\Delta y)\frac{\mathrm{U}[i,j+1]-\mathrm{U}[i,j]}{\Delta y}=0\\
\Rightarrow\quad &i\cdot(\mathrm{U}[i,j]-\mathrm{U}[i-1,j])+(1+j\Delta y)\frac{\mathrm{U}[i,j+1]-\mathrm{U}[i,j]}{\Delta y}=0\\
\Rightarrow\quad &i\Delta y\cdot(\mathrm{U}[i,j]-\mathrm{U}[i-1,j])+(1+j\Delta y)\left(\mathrm{U}[i,j+1]-\mathrm{U}[i,j]\right)=0\\
\Rightarrow\quad &(1+j\Delta y)\left(\mathrm{U}[i,j+1]-\mathrm{U}[i,j]\right)=i\Delta y\cdot\mathrm{U}[i-1,j]-i\Delta y\cdot\mathrm{U}[i,j]\\
\Rightarrow\quad &(1+j\Delta y)\mathrm{U}[i,j+1]=(1+j\Delta y)\mathrm{U}[i,j]+i\Delta y\cdot\mathrm{U}[i-1,j]-i\Delta y\cdot\mathrm{U}[i,j]\\
\Rightarrow\quad &(1+j\Delta y)\mathrm{U}[i,j+1]=[1+(j-i)\Delta y]\cdot\mathrm{U}[i,j]+i\Delta y\cdot\mathrm{U}[i-1,j]\\
\Rightarrow\quad &\mathrm{U}[i,j+1]=\frac{[1+(j-i)\Delta y]\cdot\mathrm{U}[i,j]+i\Delta y\cdot\mathrm{U}[i-1,j]}{(1+j\Delta y)}
\end{aligned}$$

קיבלנו נוסחת נסיגה

$$\boxed{\mathrm{U}[i,j+1]=\frac{[1+(j-i)\Delta y]\cdot\mathrm{U}[i,j]+i\Delta y\cdot\mathrm{U}[i-1,j]}{(1+j\Delta y)}}$$

הנוסחה תקפה עבור $i=1,2,\ldots,N-1$, $j=0,1,\ldots,N-2$. על ידי שימוש בתנאי השפה על $\mathrm{U}[i,0]$, $\mathrm{U}[0,j]$, ניתן לחשב את U על כל המפתחות האחרים. הקוד פייתון שרואים באיור 14 עבור הפתרון הוא קצר ואלגנטי.

תרגיל 3: מצאו פתרון בדיד $u(x,y)$ בצורת נוסחת נסיגה מעל שריג הפרשים מתאים, עבור המשוואה הדיפרנציאלית החלקית הבאה (סדר ראשון) מעל ריבוע היחידה $[0,1]\times[0,1]$.

$$\begin{cases}
xu_x-yu_y=u, & 0<x<1,\ 0<y<1\\
u(x,0)=x, & 0\le x\le 1\\
u(0,y)=0, & 0\le y\le 1\\
u(x,1)=\frac{x}{1+x}, & 0\le x\le 1\\
u(1,y)=\frac{1}{1+y}, & 0\le y\le 1
\end{cases}\tag{30}$$

```python
dx = dy = 1/N
U = np.zeros((N+1, N+1))

# Boundary conditions
for i in range(0, N+1):
    U[i,0] = 2*i*dx             # u(x,0) = 2x
    U[i,N] = i*dx               # u(x,1) = x

for j in range(0, N+1):
    U[0,j] = 0                  # u(0,y) = 0
    U[N,j] = 2 / (1+j*dy)       # u(1,y) = 2/(1+y)

def Solve(U):
  for i in range(1,N):
    for j in range(0,N-1):
      U[i,j+1] = ((1 + (j-i)*dx)*U[i,j] + i*dx*U[i-1,j]) / (1 + j*dx)
```

איור 14: קוד פייתון עבור דוגמא 5

מצאו פתרון סגור לבעייה באמצעות השיטות שנלמדו בתחילת הקורס, ובצעו השוואה בינו לבין הפתרון הבדיד מעל נקודות השריג.

הדרכה: מומלץ קודם למצוא פתרון סגור לבעייה באמצעות השיטות הרגילות שנלמדו בקורס. זה יעזור לאמת את הפתרון הבדיד. יש לבחור מספרי חלוקה שונים N_x , N_y , עבור שני הצירים של התחום $[0,1]\times[0,1]$. ישנם שלוש נוסחאות נגזרת בדידה עבור $\mathbf{U}_x$: אחורית, קדמית, ומרכזית, ושלוש עבור $\mathbf{U}_y$. בסך הכל יש לבדוק 9 מצבים. ברוב המקרים מתקבלים נוסחאות נסיגה שאינן עובדות מסיבות שונות (כגון חילוק באפס). הדרך המומלצת להתגבר על בעיות אלה היא לבנות נוסחת נסיגה "הפוכה": במקום התקדמות מ־ 1 ל־ N , יש להתקדם מ־ N ל־ 1 ! כלומר, יש להתקדם משלב i לשלב $i-1$! לשם כתיבת קוד פייתון, מומלץ להעתיק את המחברת קולאב של התרגיל הקודם ולבנות עליה את הקוד המתאים לתרגיל זה. על סמך הנסיונות שלנו יש בעיית יציבות מינורית סביב הנקודה $(0,1)$. בכדי למזער אותה, יש לבחור מספר חלוקה N_x שהוא משמעותית יותר גדול מ־ N_y (למשל פי 1000).

נציג תרגילים נוספים בנושא משוואת הגלים החד-מימדית לסוגיה השונים (חלק מהם). כמו קודם נתאמן שוב על דוגמה פשוטה של שריג 4×4 שניתן לפתור באופן ידני. עיינו שוב בדוגמה 1 מסוף הפרק הקודם ובצעו שוב את ארבעת השלבים המתוארים שם כהכנה להמשך הפרק. הפעם צפויה הפתעה קטנה עם תנאי השפה על הנגזרת הראשונה, ולכן מומלץ מאוד שלא לדלג על האימון הזה.

תרגיל 4: מצאו פתרון בדיד מהצורה 4×4 בדומה לזה שהוצג בדוגמה 1. עבור משוואת הגלים החד-מימדית הבאה

$$\begin{cases} u_{tt} = 4u_{xx}, & 0 < x < 3,\ \ 0 < t < 3 \\ u(0,t) = 0, & 0 \le t \le 3 \\ u(3,t) = 1, & 0 \le t \le 3 \\ u(x,0) = x^2(3-x), & 0 \le x \le 3 \\ u_t(x,0) = 1, & 0 \le x \le 3 \end{cases}$$

הדרכה: יש לעיין שוב בדוגמה 1 מסוף הפרק הקודם ולבנות מודל בדיד מגודל 4×4 עבור המשוואה שלנו. מומלץ להשתמש

בנוסחאות נגזרת מרכזית עבור u_{xx} , u_{tt} , ובנוסחת נגזרת קדמית עבור u_t . יש לשים לב לכך שהתנאי $u_t(x,0)=1$ דורש טיפול מיוחד!

תרגיל 5: עשו שימוש בנוסחת ההפרשים שקיבלתם בפתרון תרגיל 4 בכדי לפתור בשיטת ההפרשים הסופיים את משוואת הגלים החד-מימדית הקלאסית הבאה

$$\begin{aligned}
&u_{tt} = 4u_{xx}, && 0<x<1,\ \ 0<t<8\\
&u(0,t)=0, && 0\le t\le 8\\
&u(1,t)=0, && 0\le t\le 8\\
&u(x,0)=\sin(\pi x), && 0\le x\le 1\\
&u_t(x,0)=0, && 0\le x\le 1
\end{aligned}$$

הדרכה: יש לשים לב שאורך המיתר וקטע הזמן שונים מאלה שבתרגיל 4, ולכן נדרשות התאמות. גם תנאי השפה השתנו. לחובבי תיכנות: מומלץ לבנות מחברת קולאב ולבדוק אם הפתרון נראה הגיוני על ידי שרטוטים גרפיים של מצב המיתר בזמנים שונים.

תרגיל 6: מצאו פתרון בשיטת הפרשים סופיים עבור משוואת הגלים החד-מימדית עם מקדם **שיכוך (Damping)**

$$\begin{aligned}
&u_{tt}+2u_t = 4u_{xx}, && 0<x<1,\ \ 0<t<8\\
&u(0,t)=0, && 0\le t\le 8\\
&u(1,t)=0, && 0\le t\le 8\\
&u(x,0)=\sin(\pi x), && 0\le x\le 1\\
&u_t(x,0)=0, && 0\le x\le 1
\end{aligned}$$

קבלו פתרון בצורת נוסחת נסיגה דומה לזו שקיבלנו עבור משוואת החום (או תרגיל 5).

תרגיל 7: מצאו נוסחת הפרשים סופיים עבור **משוואת Klein-Gordon**

$$\begin{aligned}
&u_{tt}-c^2u_{xx}+m^2u=0, && 0<x<1,\ \ 0<t<1\\
&u(0,t)=0, && 0\le t\le 1\\
&u(1,t)=0, && 0\le t\le 1\\
&u(x,0)=\sin(4\pi x), && 0\le x\le 1\\
&u_t(x,0)=0, && 0\le x\le 1
\end{aligned}$$

זוהי גירסה חד-מימדית של משוואת גלים מתחום הפיזיקה של שדות קוואנטיים. קבלו פתרון בצורת נוסחת נסיגה דומה לזו שקיבלנו עבור משוואת החום (או תרגיל 5).

הדרכה: בספרי פיזיקה שונים המשוואה נקראת גם משוואת Klein-Fock-Gordon או משוואת Klein-Gordon-Fock. זוהי משוואת גלים מתחום תורת היחסות הקוואנטית, והיא קשורה גם למשוואת שרדינגר, ולשדה שרדינגר.

תרגיל 8: (Schrödinger Equation)

פתרו את **משוואת שרדינגר** החד-מימדית עם תנאי השפה הבאים ע"י שיטת ההפרשים הסופיים

$$
\begin{aligned}
&i\hbar\psi_t = -\frac{\hbar^2}{2m}\psi_{xx} + V(x,t)\psi, && a < x < b,\ \ 0 < t < T \\
&\psi(a,t) = 0, && 0 \le t \le T \\
&\psi(b,t) = 0, && 0 \le t \le T \\
&\psi(x,0) = f(x), && a \le x \le b
\end{aligned}
$$

זוהי גירסה חד-מימדית של משוואת הגלים של שרדינגר הידועה מתחום הפיזיקה הקוואנטית. יש למצוא פתרון בצורת נוסחת נסיגה דומה לזו שקיבלנו עבור משוואת החום (או תרגיל 5).

הדרכה: סוג נוסף של משוואת גלים חד-מימדית מתחום הפיזיקה הקוואנטית. הפונקציה $\psi(x,t)$ נקראת **פונקציית הגל** של שרדינגר עבור **חלקיק** בעל מסה m. הפתעה! הפונקציה $\psi(x,t)$ מקבלת ערכים מרוכבים! למרות שהארגומנטים שלה (x,t) ממשיים! מסתבר שזה לא מהווה מכשול לשיטת ההפרשים הסופיים, ויש לפעול כרגיל. יש לשים לב למספר המדומה $i = \sqrt{-1}$ שמופיע בתחילת המשוואה, ולהזהר שלא לבלבל אותו עם האינדקס i (לכן מומלץ להשתמש באינדקסים j , k , עבור שריג ההפרשים). הפרמטר m מציין את המסה של החלקיק. הקבוע $\hbar$ קשור לקבוע פלנק (Max Plank). מאחר והאספקט הפיזיקאלי של הבעייה לא חשוב לנו פה, אפשר להניח כי $\hbar = 1$. הפונקציה $V(x,t)$ היא פונקציית האנרגיה הפוטנציאלית של החלקיק, הנתונה לנו מראש, ולכן נוכל להשתמש בערכים בדידים שלה $V(x_j,t_k)$ בנוסחת הנסיגה. כמו כן, הפונקציה $f(x)$ נתונה כתנאי שפה $\psi(x,0) = f(x)$ בזמן $t = 0$, ולכן נוכל להשתמש בערכים בדידים שלה $f(x_j)$ בנוסחת הנסיגה. כמו בדוגמאות הקודמות, יש חשיבות רבה לגודל $\gamma = \frac{\Delta t}{2m\Delta x^2}$ שעשוי לצוץ במהלך הפתרון. תנאי היציבות (**תנאי CFL**) להתכנסות הפתרון הבדיד הוא: $\gamma \le 0.5$. יש למצוא נוסחת נסיגה מהצורה $\mathbf{U}[j,k+1] = \Psi\big(U[i,k], V(x_i,t_k), f, \Delta x, \Delta t\big)$, כאשר $\mathbf{U}$ הוא שריג ההפרשים של $\psi(x,y)$, והביטוי Ψ שבצד ימין הוא ביטוי אלגברי שבו מעורבים הפרמטרים שחושבו בשלב הקודם k.

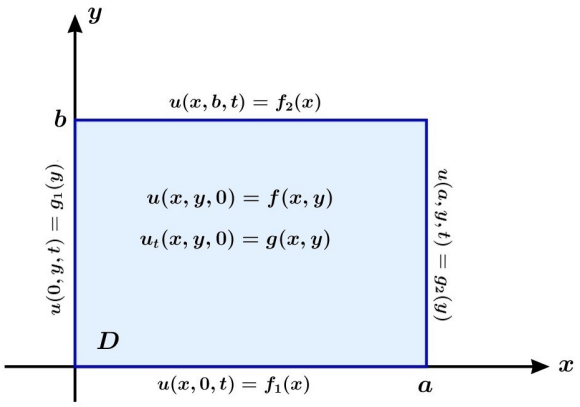
4. משוואת הגלים במישור (2D)

חוברת ההרצאות של הקורס (חלק III, פרק 6) מטפלת במשוואת הגלים החד-מימדית מעל מיתר חד-מימדי (1D). הטיפול האנליטי עבור המקרה הדו-מימדי (2D-Membrane) דומה אך מוגבל לתחומים ותנאי התחלה/שפה מאוד ספציפיים. בסעיף הנוכחי נראה כיצד להשתמש בשיטת ההפרשים הסופיים (ראה חלק IV של חוברת ההרצאות) בכדי לקבל פתרון נומרי עבור המשוואה ללא הגבלות מסוג זה. לשם פשטות נסתפק בדוגמה של מלבן דו-מימדי $D = [0,a]\times[0,b]$ שעשוי מחומר אחיד בצפיפות אחידה (ראה איור 15).

עבור כל נקודה (x,y) על המשטח D, $u(x,y,t)$ היא נקודת הגובה z שבה הנקודה נמצאת בזמן t. הפונקציה $u(x,y,t)$ גזירה פעמיים ברציפות במשתנים x, y, בנקודות פנימיות של התחום D, גזירה פעמיים ברציפות במימד הזמן t, ומקיימת את המשוואה הדיפרנציאלית החלקית הבאה

$$u_{tt} = c^2(u_{xx} + u_{yy}), \quad (x,y) \in D, \quad 0 < t < \infty$$

כאשר c הוא קבוע פיזיקאלי שנקבע על פי סוג החומר וצפיפותו.



$$\begin{cases} u_{tt}=c^2(u_{xx}+u_{yy}), & 0<x<a,\ 0<y<b,\ 0<t<\infty \\ u(x,y,0)=f(x,y), & 0<x<a,\ 0<y<b \\ u_t(x,y,0)=g(x,y), & 0<x<a,\ 0<y<b \\ u(x,0,t)=f_1(x), & 0\le x\le a,\ 0\le t<\infty \\ u(x,b,t)=f_2(x), & 0\le x\le a,\ 0\le t<\infty \\ u(0,y,t)=g_1(y), & 0\le y\le b,\ 0\le t<\infty \\ u(a,y,t)=g_2(y), & 0\le y\le b,\ 0\le t<\infty \end{cases}$$

איור 15: משוואת גלים עם תנאי שפה מסוג דיריכלה מעל משטח דו-מימדי

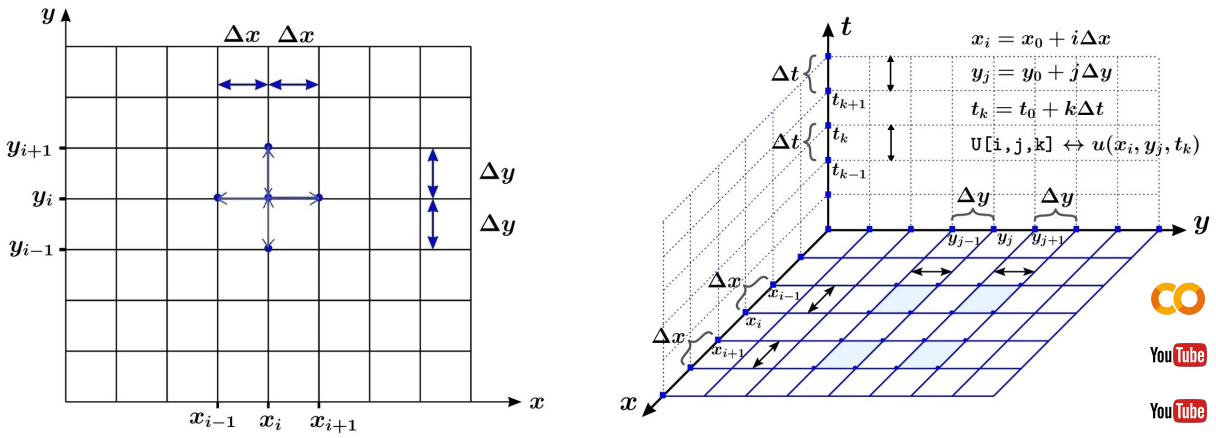
בדומה למשוואת החום, קיימים סוגים שונים של תנאי שפה/התחלה עבור משוואת הגלים. אנו נדון בסוג השכיח שבו נתון לנו הגובה והמהירות של כל נקודה פנימית של המשטח בזמן $t=0$, באמצעות שתי פונקציות $f(x,y)$, $g(x,y)$. נקודות השפה של D נמצאות במצב מנוחה קבוע ואינן נעות לאורך ציר הזמן. למשל אם המשטח D נתון על ידי מלבן $[0,a]\times[0,b]$, אז תנאי השפה נראים כמו בצד ימין של איור 15

4.1 פתרון באמצעות שיטת ההפרשים הסופיים

כמו בדוגמה של משוואת החום, לשם פשטות נסתפק בתחום ריבועי $a=b=L$ מעל קטע זמן סופי $[0,T]$. נבחר מספר חלוקה N עבור הקטע $[0,L]$, ומספר חלוקה N_t עבור קטע הזמן $[0,T]$. כמו בסעיף הקודם נבנה שריג תלת-מימדי $\mathbf{U}$ מהצורה

$$(N+1)\times(N+1)\times(Nt+1)$$

נזכיר שוב כי חלוקת ציר כלשהו ל־N תתי-קטעים מייצרת $N+1$ צמתים!



איור 16: שריג הפרשים רב-מימדי

על סמך נוסחאות הגזירה שפיתחנו בסעיף הראשון, נוכל להשתמש בנוסחאות הגזירה הבאות

$$\mathbf{U}_{tt}[i,j,k]=\frac{\mathbf{U}[i,j,k]-2\mathbf{U}[i,j,k-1]+\mathbf{U}[i,j,k-2]}{\Delta t^2} \tag{31}$$

$$\mathbf{U}_{xx}[i,j,k] = \frac{\mathbf{U}[i+1,j,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i-1,j,k]}{\boldsymbol{\Delta x^2}} \tag{32}$$

$$\mathbf{U}_{yy}[i,j,k] = \frac{\mathbf{U}[i,j+1,k] - 2\mathbf{U}[i,j,k] + \mathbf{U}[i,j-1,k]}{\boldsymbol{\Delta x^2}} \tag{33}$$

יש לשים לב לכך שבחרנו נוסחת הפרשים אחורית עבור $\mathbf{U}_{tt}$, ונוסחת הפרשים מרכזית עבור $\mathbf{U}_{xx}$, $\mathbf{U}_{yy}$. מנוסחה (31) קל לחלץ את $\mathbf{U}[i,j,k]$

$$\mathbf{U}[i,j,k] = 2\mathbf{U}[i,j,k-1] - \mathbf{U}[i,j,k-2] + \mathbf{U}_{tt}[i,j,k]\boldsymbol{\Delta t^2}$$

נתון לנו כי $\mathbf{U}_{tt} = \boldsymbol{c^2}\left(\mathbf{U}_{xx} + \mathbf{U}_{yy}\right)$ לכן

$$\mathbf{U}[i,j,k] = 2\mathbf{U}[i,j,k-1] - \mathbf{U}[i,j,k-2] + \boldsymbol{c^2}\Big(\mathbf{U}_{xx}[i,j,k] + \mathbf{U}_{yy}[i,j,k]\Big)\boldsymbol{\Delta t^2}$$

על סמך נוסחת הלפלסיאן הבדיד (27) שפגשנו בעמוד 18,

$$\boldsymbol{c^2}\Big(\mathbf{U}_{xx}[i,j,k] + \mathbf{U}_{yy}[i,j,k]\Big) = \frac{\boldsymbol{c^2}\boldsymbol{\Delta}\mathbf{U}}{\boldsymbol{\Delta x^2}}$$

קבלנו נוסחת נסיגה בציר הזמן k

$$\mathbf{U}[i,j,k] = 2\mathbf{U}[i,j,k-1] - \mathbf{U}[i,j,k-2] + \frac{\boldsymbol{c^2\Delta t^2}}{\boldsymbol{\Delta x^2}}\boldsymbol{\Delta}\mathbf{U}$$

מקובל לסמן

$$\gamma = \frac{\boldsymbol{c\Delta t}}{\boldsymbol{\Delta x}}$$

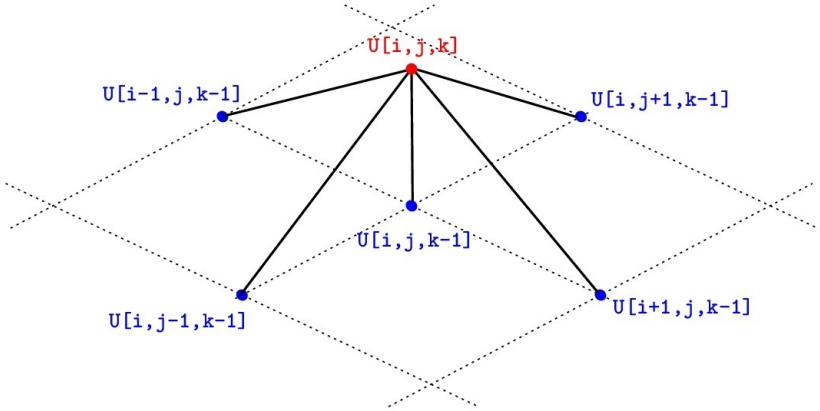
ואנו מקבלים נוסחת נסיגה קצרה וברורה יותר

$$\boxed{\mathbf{U}[i,j,k] = 2\mathbf{U}[i,j,k-1] - \mathbf{U}[i,j,k-2] + \gamma^2\boldsymbol{\Delta}\mathbf{U}} \tag{34}$$

הביטוי $\boldsymbol{\Delta}\mathbf{U}$ הוא האופרטור לפלס הבדיד (Discrete Laplace Operator) אותו פגשנו כבר בדיון על משוואת החום! הוא מופיע שוב ושוב במשוואות מסוג זה. בסביבות תוכנה, זהו מבנה נתונים שנקרא **Stencil** ונהוג לייצג אותו על ידי סכימה גאומטרית כמו באיור 17.

הפעם נוסחת הנסיגה מתבססת על שני מצבים קודמים של הגל בנקודות זמן $k-1$, $k-2$. כלומר, בכדי שנוכל להפעיל את הנוסחה לשם קבלת פתרון מלא, דרושים לנו מצב הגל בנקודות הזמן $k=0$, $k=1$. מצב הגל בנקודת הזמן $k=0$ נתון לנו באמצעות תנאי השפה $u(x,y,0) = f(x,y)$. בכדי לקבל את מצב הגל בנקודת הזמן $k=1$ נשתמש בתנאי השפה השני $u_t(x,y,0) = g(x,y)$.

$$\mathbf{U}_t[i,j,k] = \frac{\mathbf{U}[i,j,k+1] - \mathbf{U}[i,j,k]}{\boldsymbol{\Delta t}}$$




```
DU = U[i-1,j,k-1] + U[i+1,j,k-1] + U[i,j-1,k-1] + U[i,j+1,k-1] - 4*U[i,j,k-1]
U[i,j,k] = 2*U[i,j,k-1] - U[i,j,k-2] + gamma*DU
```

איור 17: סטנסיל (Stencil) עבור נוסחת הנסיגה לפתרון משוואת הגלים (קוד פייתון)

נציב $k=0$ ונקבל

$$\mathbf{U}_t[i,j,0] = \frac{\mathbf{U}[i,j,1] - \mathbf{U}[i,j,0]}{\boldsymbol{\Delta t}} = g(x_i, y_j)$$

לכן מצב הגל בנקודת הזמן $k=1$ נתון על ידי

$$\mathbf{U}[i,j,1] = \mathbf{U}[i,j,0] + \mathbf{U}_t[i,j,0]\boldsymbol{\Delta t} = f(x_i, y_j) + g(x_i, y_j)\boldsymbol{\Delta t}$$

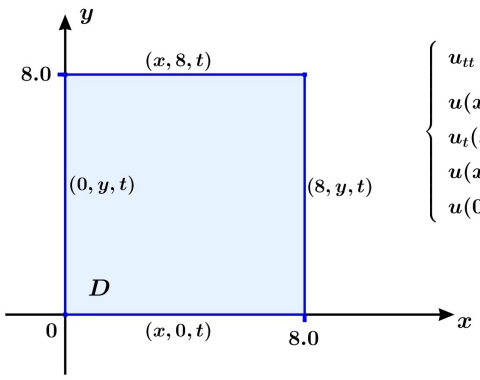
לכן בשלב זה יש בידינו את כל מה שדרוש בכדי לרשום אלגוריתם לביצוע סימולציה של מצבי הגל בכל נקודת זמן $k = 0, 1, 2, \ldots, N_t - 1$. באיור 18 מוצג קוד פייתון בן 5 שורות בלבד עבור נוסחת הנסיגה.

```
def Solve(U):
  Nrows, Ncols, Nt = U.shape
  for k in range(2, Nt):
    for i in range(1, Nrows-1):
      for j in range(1, Ncols-1):
        DU = U[i-1,j,k-1] + U[i+1,j,k-1] + U[i,j-1,k-1] + U[i,j+1,k-1] - 4*U[i,j,k-1]
        U[i,j,k] = 2*U[i,j,k-1] - U[i,j,k-2] + gamma**2 * DU
```

איור 18: קוד NumPy עבור פתרון משוואת הגלים

יש לשים לב לכך שהקוד הזה אינו מחשב את הערכים של $\mathbf{U}[i,j,k]$ עבור מפתחות הקצה $i = 0, N$ או $j = 0, N$! כי הטווח `range(1, N)` אינו כולל את הערכים 0, N! הטיפול בנקודות הקצה מתבצע על ידי תנאי השפה בשלב הקודם להפעלת נוסחת הנסיגה. קוד NumPy מלא לפתרון דוגמה טיפוסית של משוואת הגלים עם הסברים עבור כל הפרמטרים המעורבים, נראה בדוגמה הבאה.

דוגמה 6: נפתור את משוואת הגלים הבאה באמצעות שיטת ההפרשים הסופיים



$$\begin{cases} u_{tt} = u_{xx} + u_{yy}, & 0 < x < 8,\ 0 < y < 8,\ 0 < t < \infty \\ u(x,y,0) = 8e^{-8\left((x-4)^2+(y-4)^2\right)} \sin\frac{\pi x}{4} \sin\frac{\pi y}{4}, & 0 < x < 8,\ 0 < y < 8 \\ u_t(x,y,0) = 0, & 0 < x < 8,\ 0 < y < 8 \\ u(x,0,t) = u(x,8,t) = 0, & 0 \le x \le 8,\ 0 \le t < \infty \\ u(0,y,t) = u(8,y,t) = 0, & 0 \le y \le 8,\ 0 \le t < \infty \end{cases}$$

הפעם התחום הריבועי שלנו הוא $D = [0,8]\times[0,8]$. לכן בכדי לקבל רמת דיוק גבוהה נאלץ לבצע חלוקה עדינה מזו שעשינו במקרה של משוואת החום.

תנאי שפה: כל ארבעת הפונקציות $f_1(x)$, $f_2(x)$, $g_1(x)$, $g_2(x)$ מתאפסות על שפת הריבוע. פונקציית ההתחלה $f(x,y)$ היא מכפלה מעניינת בין פונקציית גאוסיאן ושתי פונקציות טריגונומטריות (בהשראת דוגמת היוטיוב [23]).

$$u(x,y,0) = 8e^{-8\left((x-4)^2+(y-4)^2\right)} \sin\frac{\pi x}{4} \sin\frac{\pi y}{4}$$

כפי שניתן לראות בסרטון הסימולציה המקושר, הפונקציה הזו מייצרת שתי "קפיצות" (blips) עיליות ושתי קפיצות תחתיות, שלאחר מכן משכפלים את עצמם בכל מיני צורות. מהירות הגל $u_t(x,y,0)$ בזמן $t=0$ היא אפס בכל נקודות התחום.

בסימולציה שלנו בחרנו מספר חלוקה $N = 160$, ולכן $\Delta x = \Delta y = 0.05$. כל שאר פרטי הסימולציה מפורטים ומוסברים במחברת הקולאב המקושרת לאיורים 18, 19.

```
L = 8.0                       # Length of square membrane (LxL)
N = 50                        # Grid division size
c = 1.0                       # Phisical wave speed constant
fps = 30                      # Frames per second (for video simulation)
seconds = 20                  # Simulation time (video duration)
num_frames = int(seconds*fps) # Total number of time frames (defined in wave2sim.py module)
dx = dy = L/N                 # Step sizes = x,y grid units
gamma = 0.4                   # CFL (Courant/Friedrich/Lewy) condition is met: gamma = c*dt/dx <= sqrt(2)/2
#gamma = c*dt/dx              # This is the standard definition of gamma. See class notes or project booklet
# We prefer to define gamma explicitly to ensure CFL condition, and then use it to define dt
dt = gamma * dx/c             # Automatic choice for dt (See link below)
T = seconds * fps * dt        # Physical time (the real time)
U = np.zeros((N+1, N+1, num_frames))  # Our FDM array container (grid)
```

איור 19: הפרמטרים של הסימולציה של משוואת הגלים

נסביר חלק מהפרמטרים:

א. CFL - Courant/Friedrich/Lewy condition זהו תנאי יציבות שמבטיח כי הפתרון הבדיד

מתכנס לפתרון אמיתי או חריג. במקרה של משוואת הגלים, תנאי היציבות[6] הוא

$$\left(\frac{c\Delta t}{\Delta x}\right)^2+\left(\frac{c\Delta t}{\Delta y}\right)^2\le 1$$

במקרה שלנו, השריג הוא אחיד (כלומר $\Delta x=\Delta y$), ולכן תנאי היציבות הוא

$$\gamma=\frac{c\Delta t}{\Delta x}\le\frac{\sqrt{2}}{2}$$

התנאי מאפשר לנו לבחור ערך "מיטבי" של יחידת זמן Δt כפונקציה של Δx, Δy, שמבטיח פתרון יציב ובנוסף לכך הערך המתקבל של Δt לרוב מניב סימולציה במהירות נעימה לעיין אנושית. פרטים נוספים על התנאי החשוב הזה ניתן ללמוד מהבלוג [19].

ב. בכדי להבטיח את קיום תנאי CFL, החלטנו להתחיל עם הגדרה קונקרטית של $\gamma=0.4$, שנמצא בסביבות מרכז הקטע $(0,\frac{\sqrt{2}}{2})$ של הערכים שתנאי CFL מאפשר, ולאחר מכן להגדיר את Δt באמצעות הנוסחה $\Delta t=\frac{\gamma\Delta x}{c}$.

ג. הפרמטר **fps** (frames per second) מציין את מספר התמונות לשניה בסרטון הסימולציה. ברוב המקרים המספר המומלץ לעיין אנושית הוא $\text{fps}=30$.

ד. כמו שראינו כבר במשוואת החום, יש להבחין בין **זמן הסימולציה seconds** שהוא גם זמן סרטון הוידאו, לבין **הזמן הפיזיקאלי T** שהוא הזמן האמיתי שבו תופעת הגל מתרחשת, אשר עשוי להיות קצר מאוד (מיקרו שנייה למשל) או ארוך מאוד (מיליון שניות למשל)[7]. ברוב המקרים לא נרצה לצפות בסרטון כזה יותר מ־30 עד 60 שניות ולכן יש להאיץ או להאט את הסימולציה בהתאם. על ההאצה וההאטה ניתן לשלוט באמצעות בחירת גודלי חלוקה מתאימים ו/או סינון חלק מהתמונות (דגימה).

ה. שריג ההפרשים שלנו U הוא מערך NumPy תלת-מימדי שצורתו היא

$$(N+1)\times(N+1)\times\text{num\_frames}$$

מספר התמונות בסימולציה **num_frames** שווה למעשה למכפלה **seconds*fps** (זמן הסימולציה כפול מספר התמונות לשניה). הזמן הפיזיקאלי **T** הוא **seconds*fps*dt** ולא בהכרח **num_frames*dt**! מאחר ובסימולציות יותר מורכבות אין חובה להשתמש בכל התמונות ומסתפקים במדגם קטן מהם (כמו תמונה אחת לכל עשר). לשם פשטות, במקרה הנוכחי אנו משתמשים בכל התמונות.

להשלמת התמונה יש להבין כיצד מזינים את תנאי השפה לשריג U. ברוב הדוגמאות שמופיעות באינטרנט או בספרי לימוד רבים זה מתבצע באמצעות מגע ישיר עם המפתחות של המערך U. אולם סוג עבודה כזה מתאים רק עבור תכנתים מיומנים שמתמצאים בנבכי המערכים

[6] התבססנו על שאילתא של ChatGPT אשר אומתה באמצעות קוד. ראה מסמך: https://samyzaf.com/fdm/gpt_wave2d_cfl.pdf
[7] יחידת הזמן בכל המקרים היא שנייה

הרב-מימדיים של NumPy. בכדי להתגבר על המכשול הזה, חבילת הפייתון שלנו fdmtools שאותה יצרנו עבור הקורס, כוללת מימשק יותר נוח שחוסך מאיתנו את ההתעסקות עם מנגנוני תיכנות מורכבים מדי.

```
f  = lambda x,y: 8*exp(-8*((x-4)**2 + (y-4)**2)) * sin(pi*x/4) * sin(pi*y/4)
g  = lambda x,y: 0
f1 = lambda x: 0
f2 = lambda y: 0
g1 = lambda y: 0
g2 = lambda y: 0
```

איור 20: הגדרת פונקציות השפה עבור השריג

מלבד הפקודה **lambda** שמציינת פונקציה קצרה, הכל נראה כמו הגדרת פונקציות מתמטיות רגילות. הפקודה האחרונה **RunSimulation** בקוד שבמחברת הקולאב, מבצעת גם את הזנת התנאים האלה לשריג, מריצה את הסימולציה ובונה את השריג $\mathbf{U}$. עובדה זו משחררת אותנו מהתעסקות עם פרטים טכניים רבים, ואמורה להקל עלינו בהרצה של דוגמאות נוספות שנפגוש בתרגילים הבאים.

תרגיל 9: פתרו את משוואת הגלים הבאה באמצעות שיטת ההפרשים הסופיים ובצעו סימולציה עבור זמן פיזיקאלי $T = 30$ שניות. וודאו שהסימולציה מציגה לפחות 6 מחזורים של הגל.

$$\begin{cases} u_{tt} = u_{xx} + u_{yy}, & 0 < x < 2,\ \ 0 < y < 2,\ \ 0 < t < \infty \\ u(x,y,0) = xy(2-x)(2-y), & 0 < x < 2,\ \ 0 < y < 2 \\ u_t(x,y,0) = xy, & 0 < x < 2,\ \ 0 < y < 2 \\ u(x,0,t) = u(x,2,t) = 0, & 0 \le x \le 2,\ \ 0 < t < \infty \\ u(0,y,t) = u(2,y,t) = 0, & 0 \le y \le 2,\ \ 0 < t < \infty \end{cases}$$

יש לשים לב לכך ש־$c = 1.0$.

הדרכה: העתק את מחברת הקולאב wave.ipynb למחברת חדשה wave_ex9.ipynb ובצע את השינויים הנדרשים לביצוע הסימולציה. יש לשים לב לכך שהפעם תנאי השפה השני $u_t(x,y,0)$ אינו אפס! לכן מדוגמה זו ניתן ללמוד על ההשפעה שיש למהירות ההתחלתית של הגל בזמן $t = 0$ על ההתנהגות שלו בהמשך. ניתן לצפות בדוגמה של סימולציה כזו על ידי לחיצה על כפתור היוטיוב למעלה.

מדובר בתחום ריבועי $D = [0,2]\times[0,2]$ עם תנאי שפה פשוטים: כל ארבעת הפונקציות $f_1(x)$, $f_2(x)$, $g_1(x)$, $g_2(x)$ מתאפסות על שפת הריבוע. מצב היריעה בזמן $t = 0$ הוא $xy(2-x)(2-y)$, ומהירות התחלתית בזמן $t = 0$ היא $u_t(x,y,0) = xy$.

פתרון בעייה זו באמצעות השיטות הקלאסיות מצריך פיתוח של טור פורייה דו-מימדי של הפונקציה $g(x,y) = xy(2-x)(2-y)$, ולא נציג אותו כאן מפאת קוצר הזמן והיגיעה. התלמיד המעוניין מוזמן להציץ בספרים [17], [18], [20], [22].

נסיים את הפרק במשוואת פואסון על שמו של המתמטיקאי הצרפתי Siméon Denis Poisson. זוהי

למעשה משוואת לפלס לא הומוגנית. בניגוד לדוגמאות שפגשנו עד כה, הפתרון שלה באמצעות שיטת ההפרשים הסופיים מצריך תושיה ברמה יותר גבוהה. מערכת המשוואות המתקבלת אינה קלה לפתרון ומצריכה שיטות ליכסון מתקדמות מתחום האלגברה הליניארית, ומפאת קוצר היריעה לא נכנס אליו בחוברת הזו. נסתפק בזוג תרגילים לבנייה ידנית של שריג הפרשים קטן מהצורה 4×4 ו־ 5×5, מהסוג שפגשנו בדוגמה 1 בפרק המבוא.

תרגיל 10: משוואת פואסון (S. D. Poisson) דו-מימדית. עיינו שוב בדוגמה 1 שבפרק המבוא, ומצאו פתרון בדיד מהצורה 4×4 עבור משוואת פואסון עם תנאי השפה המצורפים.

$$\begin{cases} u_{xx}+u_{yy}=10e^{x+\frac{y}{3}}, & 0<x<3,\ \ 0<y<3 \\ u(x,0)=9e^{x}, & 0\le x\le 3 \\ u(x,3)=9e^{x+1}, & 0\le x\le 3 \\ u(0,y)=9e^{\frac{y}{3}}, & 0\le y\le 3 \\ u(3,y)=9e^{3+\frac{y}{3}}, & 0\le y\le 3 \end{cases}$$

תרגיל 11: משוואת פואסון (S. D. Poisson) דו-מימדית. עיינו שוב בדוגמה 1 שבפרק המבוא, ומצאו פתרון בדיד מהצורה 5×5 עבור משוואת פואסון עם תנאי השפה המצורפים.

$$\begin{cases} u_{xx}+u_{yy}=17e^{x+\frac{y}{4}}, & 0<x<4,\ \ 0<y<4 \\ u(x,0)=16e^{x}, & 0\le x\le 4 \\ u(x,4)=16e^{x+1}, & 0\le x\le 4 \\ u(0,y)=16e^{\frac{y}{4}}, & 0\le y\le 4 \\ u(4,y)=16e^{4+\frac{y}{4}}, & 0\le y\le 4 \end{cases}$$

יש לשים לב שהפעם גודל החלוקה הוא $N=4$, ולכן צורת השריג 5×5. בעייה לא פשוטה. הפעילות המומלצת לגביה מפורטת בהדרכה.

הדרכה: בעייה קשה. מדובר במערכת של 25 משוואות ב־ 25 נעלמים. תשעה מהם נפתרים על ידי תנאי השפה ואז נותרים עם מערכת מוקטנת של 16 משוואות ב־ 16 נעלמים. סדר הנעלמים במערכת יש חשיבות. הסדר המקובל נקרא **לקסיקוגרפי**:

$$\mathrm{U}[1,1],\ \mathrm{U}[1,2],\ \mathrm{U}[1,3],\ \mathrm{U}[1,4],\ \mathrm{U}[2,1],\ \mathrm{U}[2,2],\ \mathrm{U}[2,3],\ \ldots\ \mathrm{U}[4,1],\ \mathrm{U}[4,2],\ \mathrm{U}[4,3],\ \mathrm{U}[4,4],$$

מומלץ רק לחשב את מטריצת המקדמים של המערכת המוקטנת. למטריצה יש צורה מבנה סמטרי מובחן מסוג ידוע, וניתן למצוא שיטות ליכסון שלה במאמץ נוסף.

5. משוואת Black-Scholes-Merton

משוואת Black-Scholes-Merton, או בקיצור **BSM**, היא משוואה דיפרנציאלית חלקית לחישוב מחירי נגזרות פיננסיות (כגון אופציות PUT או CALL בשוק המניות) כפונקציה של מחיר הנכס הבסיסי s וזמן ההבשלה t. המשוואה מתבססת על מודל כלכלי בשם Black–Scholes model אשר פותח על ידי הכלכלנים Fischer Black, Myron Scholes, בשנת 1997. מיירון שולס ורוברט מרטון זכו בפרס נובל לכלכלה עבור תרומתם בנושא זה (פישר בלק נפטר בשנת 1995).

השווי v (value) של אופציה במחיר מימוש K (strike price) הוא פונקציה $v = v(s,t)$ של מחיר הנכס s (בדרך כלל מנייה) וזמן ההבשלה t. במודל של בלק-שולס הפונקציה $v(s,t)$ צריכה לקיים את המשוואה הדיפרנציאלית החלקית הבאה, שנקראת משוואת Black-Scholes-Merton:

$$v_t + rsv_s + \tfrac{1}{2}\sigma^2 s^2 v_{ss} = rv \tag{35}$$

הקבוע r הוא ערך הריבית הממוצעת לתקופת הבשלות, σ הוא קבוע סטטיסטי שמודד את **התנודתיות (volatility)** של מחיר הנכס. מומלץ לצפות בסרטון היוטיוב הנ"ל בכדי לקבל הסבר קצר על הנגזרות פיננסיות שבהן המשוואה מטפלת. המשוואה הזו נובעת משיקולים פיזיקאליים של פיזור חום, תנועה בראונית, ותהליכים סטוכסטיים. הסבר מפורט מובא בפרק 7.

בדומה לטיפול במשוואות החום והגלים, בהינתן אופציה במחיר מימוש K, נבנה מודל של שריג הפרשים $\mathrm{V}[i,k]$ עבור $v(s,t)$ באופן הבא

- השריג $\mathrm{V}[i,k]$ בנוי מעל מלבן $[0,S]\times[0,T]$.
- $[0,S]$ הוא טווח מחירי הנכס.
- $S = S_{max}$ הוא הגבול העליון של מחיר הנכס: בדרך כלל $S = 2K$ או $S = 3K$.
- $[0,T]$ טווח נקודות הזמן שעבורן נרצה לחשב את ערך האופציה.

לשם חידוד, נציין שוב: **הפונקציה $\mathrm{v(s,t)}$ מחשבת את ערך האופציה על נכס שמחירו כרגע הוא s, במחיר מימוש K, בזמן הבשלה t.**

לשם בניית המודל הבדיד V יש לבחור מספר חלוקה N_s עבור קטע המחירים $[0,S]$, ומספר חלוקה N_t עבור קטע הזמן $[0,T]$. נקבל שתי רשימות צמתים

$$s_i = i\Delta s, \quad i = 0,1,2,\ldots,N_s, \quad \Delta s = \frac{S}{N_s}$$

$$t_k = k\Delta t, \quad k = 0,1,2,\ldots,N_t, \quad \Delta t = \frac{T}{N_t}$$

וכמובן נגדיר $\mathrm{V}[i,k] = v(s_i,t_k)$, כאשר $v(s,t)$ הוא פתרון אפשרי למשוואה (תחת תנאי שפה מתאימים). נגדיר את הנגזרות הבדידות V_s, V_{ss}, V_t כמו קודם. לשם פשטות נגביל את הדיון לאופציות רכש במודל האירופאי בלבד (CALL Option). זוהי אופציה המקנה לרוכש את הזכות

לקנות את הנכס במחיר קבוע מראש $\mathbf{K}$ (Strike Price או Exercise Price) בזמן קבוע מראש t (שנמדד מרגע הרכישה $t=0$). העסקה תוכל להתבצעת רק בזמן ההבשלה t ולא לפני כן כמו שהמודל האמריקאי מאפשר.

5.1 פתרון באמצעות שיטת ההפרשים סופיים

המשוואה הדיפרנציאלית עבור אופציית רכש (Call Option) היא

$$-v_t + rsv_s + \tfrac{1}{2}\sigma^2 s^2 v_{ss} = rv \tag{36}$$

נבחין כי במקרה של אופציות רכש המשוואה מתחילה בביטוי $-v_t$ ולא v_t כפי שהיא בדרך כלל מוצגת (ראו [6]). הפתרון שנציג מתבסס על הבלוג [4] של אנטוני סמולסקי, אשר בנוסף הואיל לרשום קוד פייתון מלא במחברת יופיטר, אשר עליה ביססנו את מחברת הקולאב שלנו. מקור חשוב נוסף בשפת התיכנות C++ עם פירוט מצוין של התאוריה והקוד נמצא במקור [6].

נשתמש בנוסחת הפרשים מרכזית עבור V_t ,V_{ss} ,V_s ונוסחת הפרשים קדמית

$$\mathrm{V}_s[i,k] = \frac{\mathrm{V}[i+1,k] - \mathrm{V}[i-1,k]}{2\Delta s} \tag{37}$$

$$\mathrm{V}_{ss}[i,k] = \frac{\mathrm{V}[i+1,k] - 2\mathrm{V}[i,k] + \mathrm{V}[i-1,k]}{\Delta s^2} \tag{38}$$

$$\mathrm{V}_t[i,k] = \frac{\mathrm{V}[i,k+1] - \mathrm{V}[i,k]}{\Delta t} \tag{39}$$

בשפת הפרשים, המשוואה הדיפרנציאלית (36) היא

$$-\mathrm{V}_t[i,k] + r(i\Delta s)V_s[i,k] + \frac{1}{2}\sigma^2(i\Delta s)^2\mathrm{V}_{ss}[i,k] = r\mathrm{V}[i,k]$$

לכן

$$-\mathrm{V}_t[i,k] = r\mathrm{V}[i,k] - r(i\Delta s)V_s[i,k] - \frac{1}{2}\sigma^2(i\Delta s)^2\mathrm{V}_{ss}[i,k]$$

לכן

$$-\frac{\mathrm{V}[i,k+1] - \mathrm{V}[i,k]}{\Delta t} = r\mathrm{V}[i,k] - r(i\Delta s)V_s[i,k] - 0.5\sigma^2(i\Delta s)^2\mathrm{V}_{ss}[i,k]$$

ולאחר תימרון קטן נקבל

$$\mathrm{V}[i,k+1] = \mathrm{V}[i,k] - \Delta t\left(r\mathrm{V}[i,k] - r(i\Delta s)V_s[i,k] - 0.5\sigma^2(i\Delta s)^2\mathrm{V}_{ss}[i,k]\right)$$

נוודא כי האינדקס $k+1$ אינו מופיע באגף ימין (יש להביט שוב בנוסחאות (37), (38)). לכן קבלנו נוסחת נסיגה לחישוב שווי האופציה בזמן t_{k+1} על סמך השווי בזמן t_k!

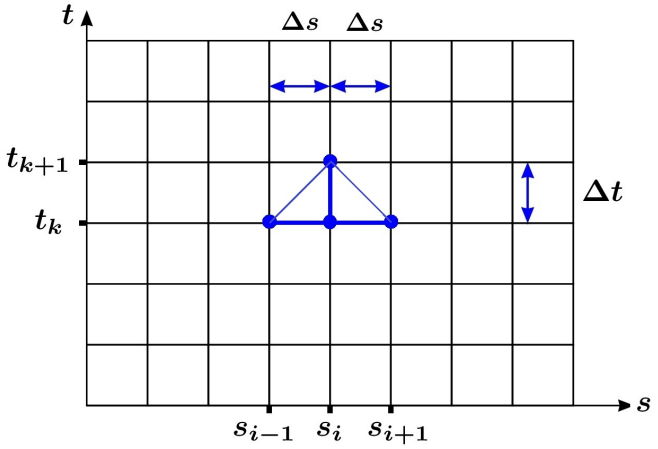
מקובל לסמן

$$\gamma = r\mathrm{V}[i,k] - r(i\Delta s)V_s[i,k] - 0.5(i\sigma\Delta s)^2\mathrm{V}_{ss}[i,k]$$

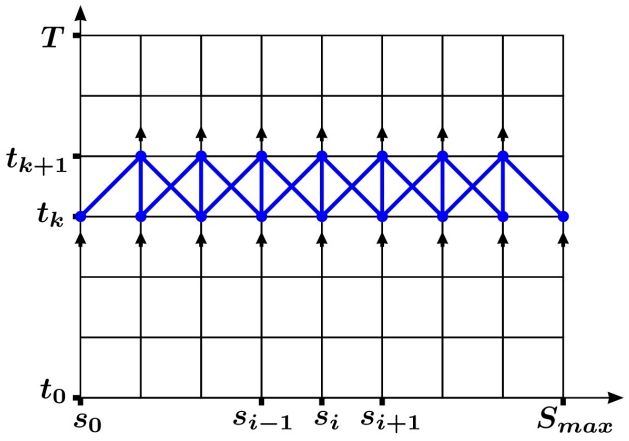
לכן נוסחת הנסיגה שלנו מתקצרת לנוסחה הבאה

$$\boxed{\mathrm{V}[i,k+1] = \mathrm{V}[i,k] - \gamma\Delta t} \tag{40}$$

מנוסחה זו אנו לומדים כיצד שווי אופציה של נכס שמחירו s_i בזמן t_{k+1}, תלוי בשלושת הערכים הקודמים s_{i-1}, s_i, s_{i+1} בזמן t_k. בכל רגע נתון t_k, ומחיר נכס s_i, קיימת הסתברות שווה לכך שברגע הבא t_{k+1}, מחיר הנכס ירד ל־s_{i-1}, יעלה ל־s_{i+1}, או יישאר כפי שהוא s_i. נוסחת הנסיגה משקללת את שלושת האפשרויות האלה (ראה סטנסיל מתאים באיור 21).



איור 21: סטנסיל (Stencil) עבור נוסחת הנסיגה למשוואת BSM



איור 22: מעבר מקומה k לקומה $k+1$ בנוסחת הנסיגה

אחד היתרונות בשיטת ההפרשים היא שניתן ללמוד מהן תנאי השפה הדרושים מצורת הסטנסיל. מנוסחאות הגזירה (38), (37), ברור כי הן אינן ישימות עבור אינדקס קצה $i = 0$ או $i = N_s$.[8] לכן נוסחת הנסיגה תופסת רק עבור $0 < i < N_s$ (ראה איור 22). לכן הערכים $\mathrm{V}(0,k)$, $\mathrm{V}(N_s,k)$, חייבים להיות נתונים כתנאי שפה. כמו כן בכדי שנוכל להפעיל את הנוסחה, הקומה $k = 0$ של השריג צריכה להיות נתונה מראש. קביעת תנאים אלה באופן מדויק היא קריטית לקבלת רמת דיוק גבוהה. נסביר כיצד תנאים אלה נקבעים בהמשך. לסיכום: בהנתן תנאי שפה עבור $t_0 = 0$, $s = 0$, $s = S$, הנוסחה הזו תניב לנו את הפתרון המלא לכל נקודת זמן!

לא קשה לרשום קוד פייתון מתאים עבור נוסחה זו, כפי שלמדנו כבר מהדוגמאות של משוואות החום והגלים. כל הסיפור של בלק-שולס-מרטון מסתכם בששת שורות הקוד הבאות:

ניתן למצוא את הקוד פייתון המלא עם הסברים במחברת הקולאב שלנו, שמתבססת על המחברת של אנטוני סמולסקי.

[8] ברוב המקרים נוסחאות הנסיגה ישימות רק עבור מפתחות פנימיים של השריג

```
for k in range(0, Nt):      # Time loop
    # And now ... Nobel prize winning diff equation ;)
    for i in range(1, Ns):  # Asset loop
        Vs = (V[i+1,k] - V[i-1,k]) / (2*ds)              # Vs[i,k] as central difference
        Vss = (V[i+1,k] - 2*V[i,k] + V[i-1,k]) / (ds**2)  # Vss[i,k] as central difference
        gamma = r*V[i,k] - r*(i*ds) * Vs - 0.5 * sigma**2 * (i*ds)**2 * Vss
        V[i,k+1] = V[i,k] - dt * gamma
```

איור 23: קוד NumPy עבור פתרון משוואת Black-Scholes-Merton

5.2 תנאי שפה

בנוסף לבלוג המעולה של אנטוני סמולסקי, נעזרנו גם במקור [5] בכדי לטפל בתנאי הקצה של המודל הבדיד שלנו. כמקובל ברוב המקרים נעבוד על קטע מחירים $[0, S]$ כאשר $S = 3K$ הוא המחיר המקסימלי של הנכס[9], K הוא מחיר המימוש המבוקש. יש לטפל בשלושה תנאים

א. $t = 0$: רכישת אופציה לזמן קצר מאוד כגון $t = 0$ (שבו רכישת האופציה והמימוש מתבצעים בו-זמנית) אינה מקרה ריאלי אך ההגיון הפשוט יאמר שבעסקה כזו לא צריך לקרות כלום. כלומר אם $s \geq K$ אז עלות האופציה היא $s - K$, ובאותו הרגע הרוכש ישלם K בכדי לרכוש את המנייה והעסקה תסתיים באפס רווח לשני הצדדים. אם $s < K$ אז עלות האופציה הוא 0, והרוכש כמובן יוותר על רכישת המנייה מאחר וההצעה שלו K גבוהה ממחירה. לכן תנאי הקצה במקרה זה הוא שלכל s בקטע המחירים $[0, S]$

$$v(s, 0) = \begin{cases} 0, & s < K \\ s - K, & s \geq K \end{cases}$$

או בצורה יותר קצרה

$$v(s, 0) = \max\{s - K, 0\}$$

כפי שתיכף נראה בגירסת הקוד.

ב. $s = 0$: כאשר מחיר הנכס הוא $s = 0$, הצעת מחיר מימוש $K > 0$ אינה ריאלית (החברה פשטה רגל), ולכן מבחינה פורמלית עלות האופציה היא 0 גם כן. כלומר $v(0, t) = 0$ לכל $0 \leq t \leq T$.

ג. $s = S$: כאמור $S = 3K$ הוא מחיר הנכס המקסימלי שנכלל במודל הבדיד שלנו V. אם מחיר הנכס גבוה מאוד ביחס למחיר המימוש המבוקש K, ההגיון הפשוט יאמר שעלות האופציה $v(S, t)$ צריכה להיות גבוהה בהתאם. ללא שיקולי ריבית, הערך צריך להיות $v(S, t) = S - K$. ההנחה היא שמחיר הנכס יישאר בסביבות המחיר הזה, ולכן על רוכש האופציה לשלם מראש את ההפרש $S - K = 2K$. אבל מאחר ומחיר המימוש K מתבצע בזמן t, יש לקחת בחשבון גם את הריבית

$$v(S, t) = S - Ke^{-rt}$$

[9] בשימושים יותר פרקטיים מקובל להגדיר את S בתור ערך המנייה שנמצא במרחק של 8 סטיות תקן מהמחיר הממוצע.

ליבת הקוד פייתון לפתרון הבעייה בשיטת ההפרשים הסופיים (כולל תנאי השפה) מוצגת באיור 24.

```
for i in range(Ns+1):
    V[i,0] = max(i*ds - K, 0)       # Initial condition at time t=0

for k in range(0, Nt+1):            # Time loop
    V[0,k] = 0                      # Boundary condition at s=0
    V[Ns,k] = S - K*exp(-r*k*dt)    # Boundary condition at s=S (Smax)

for k in range(0, Nt):      # Time loop
    # And now ... Nobel prize winning diff equation ;)
    for i in range(1, Ns):  # Asset loop
        Vs = (V[i+1,k] - V[i-1,k]) / (2*ds)              # Vs[i,k] as central difference
        Vss = (V[i+1,k] - 2*V[i,k] + V[i-1,k]) / (ds**2)  # Vss[i,k] as central difference
        gamma = r*V[i,k] - r*(i*ds) * Vs - 0.5 * sigma**2 * (i*ds)**2 * Vss
        V[i,k+1] = V[i,k] - dt * gamma
```

איור 24: קוד NumPy עבור פתרון משוואת Black-Scholes-Merton (כולל תנאי שפה)

הקוד המלא של הפונקציה **`Solve`** עבור משוואת BSM מוצג במחברת הקולאב bsm.ipynb. יש לשים לב לאובייקט **`data`** המוחזר על ידי פונקציה זו שנקרא DataFrame. ניתן להפיק ממנו מידע רב לגבי המודל הבדיד V באמצעות שאילתות פשוטות. זהו הרכיב היסודי בחבילת עיבוד וניתוח מידע pandas של פייתון. ניתן גם לייצא אותו לטבלת Excel ולנתח אותו באמצעות הכלים של תוכנת Excel.

```
data
```

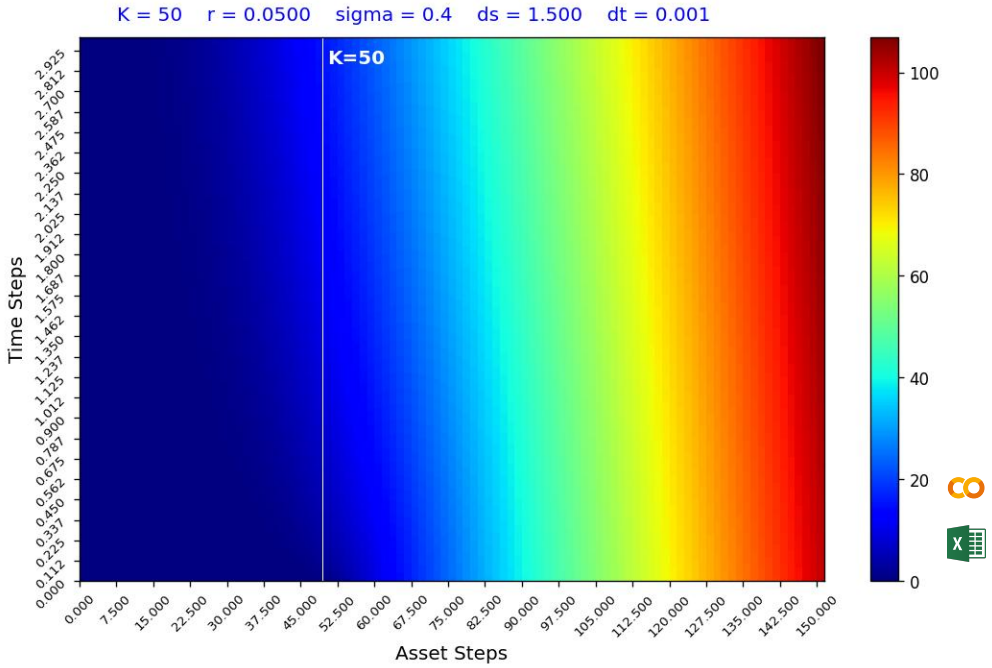
	0.00	3.75	7.50	11.25	15.00	18.75	22.50	26.25	30.00	33.75	...	116.25	120.00	123.75	127.50	131.25	135.00	138.75	142.50	146.25	150.00
0.000	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	66.250	70.000	73.750	77.500	81.250	85.000	88.750	92.500	96.250	100.000
0.004	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	66.259	70.009	73.759	77.509	81.259	85.009	88.759	92.509	96.259	100.009
0.007	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	66.268	70.018	73.768	77.518	81.268	85.018	88.768	92.518	96.268	100.018
0.011	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	66.276	70.026	73.776	77.526	81.276	85.026	88.776	92.526	96.276	100.026
0.014	0.0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	...	66.285	70.035	73.785	77.535	81.285	85.035	88.785	92.535	96.285	100.035
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2.986	0.0	0.001	0.025	0.147	0.465	1.050	1.933	3.119	4.592	6.333	...	74.485	78.063	81.651	85.247	88.850	92.459	96.072	99.690	103.311	106.934
2.989	0.0	0.001	0.025	0.147	0.467	1.053	1.937	3.124	4.599	6.340	...	74.495	78.072	81.660	85.256	88.858	92.467	96.081	99.698	103.319	106.942
2.993	0.0	0.001	0.025	0.148	0.468	1.055	1.941	3.129	4.605	6.347	...	74.504	78.082	81.669	85.265	88.867	92.475	96.089	99.706	103.326	106.949
2.996	0.0	0.001	0.025	0.149	0.470	1.058	1.945	3.134	4.611	6.355	...	74.513	78.091	81.678	85.273	88.876	92.484	96.097	99.714	103.334	106.957
3.000	0.0	0.001	0.025	0.149	0.471	1.060	1.949	3.139	4.617	6.362	...	74.523	78.100	81.687	85.282	88.884	92.492	96.105	99.722	103.342	106.965

855 rows × 41 columns

איור 25: טבלת pandas של המודל V

בנוסף לשלל הפעולות שניתן לבצע על טבלת Pandas ניתן לצייר מפת חום שבה ערך האופציה $v(s,t)$ מיוצג על ידי טמפרטורה מתאימה בתחום המלבני של השריג (ראה איור 26). ויזואליזציה נוספת ניתן לקבל על ידי משטח תלת-מימדי שניתן לייצר בקלות מהטבלה שלנו (ראה מחברת קולאב). ראה איור 28.

מפת חום ומשטח תלת-מימדי מאפשרים לקבל רושם כללי על ההתנהגות של ערך האופציה על פני מנעד רחב של מחירי נכס וזמנים, אך אינם פרקטיים לחישוב $v(s,t)$ עבור ערכים ספציפיים



איור 26: מפת חום של הייצוג הטבלאי של $\mathbf{V}$

של מחיר וזמן הבשלה. לשם כך יש לנו את קובץ האקסל הנ"ל, אך יהיה יותר אפקטיבי להעביר שאילתות ממוקדות לאובייקט **data** כפי שניתן לראות במחברת הקולאב הנלווית.

תרגיל 12: בתאריך 20.07.2020 המחיר של מניית AAPL היה 100 דולר. הריבית באותה שנה בארה"ב היתה 4.25%, ורמת הרגישות של המנייה באותה שנה היתה 0.2.

א. בנה מודל בדיד V של מניית AAPL לתקופה של 4 שנים עבור מחיר מימוש של 100 דולר (Call Option).

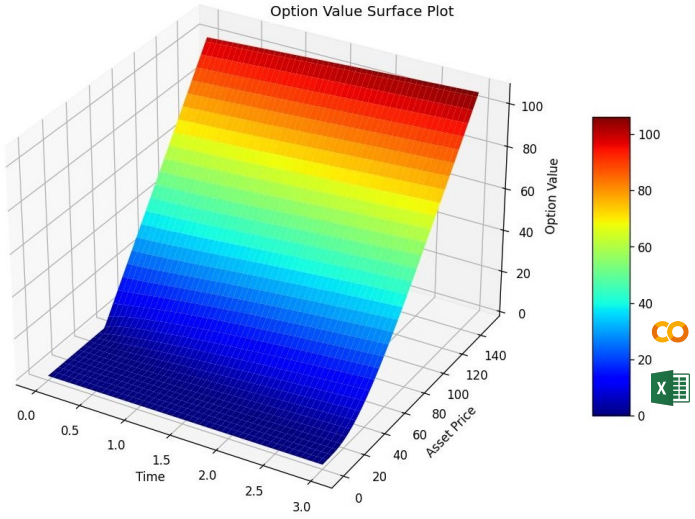
ב. העזר במודל בכדי לייצר טבלת מחירי אופציות לארבעת השנים מהתאריך הנ"ל. לאחר בניית הטבלה בדוק מהן האופציות שמומשו ברווח, ומה היה הרווח בכל מקרה? נתונים על המנייה ניתן לקבל על ידי לחיצה על איור 27.

ג. בנה מודל בדיד V של מניית AAPL לתקופה של 4 שנים עבור מחיר מימוש של 110.

ד. על סמך נתוני המנייה לשנים 2020-2023, ערוך השוואה בין הרווחים ממכירת האופציה בשני המקרים. יש להציג טבלת רווחים לשלושת השנים 2020-2023 (לאחר ניכוי עלויות וריביות כמובן).

הדרכה: יש לקחת $r = 0.0425$, $\sigma = 0.2$. לקבלת רמת דיוק סבירה יש לקחת $N_s = 100$ לכל הפחות. אפשר לעבוד עם יחידת זמן של שנה אחת, ולכן קטע הזמן שלנו יהיה $[0, 4]$ (כלומר $T = 4$). קטע מחירי המנייה יהיה $[0, 300]$ ($S = 300$). הטבלה צריכה להתייחס רק לפרקי הזמן $t = 1.0, 2.0, 3.0, 4.0$. עבור סעיפים ג-ד יש לבנות מודל חדש לגמרי שבו מחיר המימוש הוא $K = 110$. הרווח על האופציה בזמן המימוש שווה למחיר המנייה בזמן המימוש פחות מחיר המימוש והעלות של האופציה.

איור 27: אתר המידע של מניית AAPL



איור 28: ויזואליזציה של $\mathbf{V}$ באמצעות משטח תלת-מימדי

6. נספח: דיוק, יציבות, והתכנסות הפתרון הבדיד לפתרון הרציף

השאלה המרכזית לגבי הפתרון הבדיד שמתקבל ע"י שיטת ההפרשים הסופיים היא מהם התנאים שמבטיחים לנו רמת דיוק רצויה ביחס לפתרון האמיתי?

לשם פשטות, הדיון בפרוייקט זה מוגבל למשוואות דיפרנציאליות חלקיות ליניאריות מסדר 2 בלבד. מטרתנו היא בעיקר להציג את הרעיון וללמוד את השיטה בעזרת דוגמאות פשוטות. רשימת המקורות שבסוף החוברת כוללת ספרים ומאמרים הדנים במשוואות יותר כלליות. בחלק השלישי של חוברת הקורס נראה כי הצורה הכללית של מד"ח ליניאריות מסדר 2 היא

$$a(x,y)u_{xx} + 2b(x,y)u_{xy} + c(x,y)u_{yy} + d(x,y)u_x + e(x,y)u_y + f(x,y)u = g(x,y) \tag{41}$$

כאשר $a(x,y)$,$b(x,y)$,$c(x,y)$,$d(x,y)$,$e(x,y)$,$f(x,y)$,$g(x,y)$ פונקציות גזירות פעמיים ברציפות מעל תחום מלבני נתון D. אם (x_i, y_j) הם צמתים של שריג מעל המלבן D, על פי חלוקה $i = 0, 1, 2, \ldots, N_x$,$j = 0, 1, 2, \ldots, N_y$ נוכל להגדיר כמו קודם

$$\mathrm{U}[i,j] = u(x_i, y_j), \quad \mathrm{A}[i,j] = a(x_i, y_j), \quad \mathrm{B}[i,j] = b(x_i, y_j), \quad \mathrm{C}[i,j] = c(x_i, y_j),$$
$$\mathrm{D}[i,j] = d(x_i, y_j), \quad \mathrm{E}[i,j] = e(x_i, y_j), \quad \mathrm{F}[i,j] = f(x_i, y_j), \quad \mathrm{G}[i,j] = g(x_i, y_j)$$

ולאחר מעבר למשתנים הבדידים במשוואה (41), נקבל מערכת של $(N_x - 1) \times (N_y - 1)$ משוואות (מפתחות פנימיים בלבד)

$$\mathrm{A}[i,j]\,\mathrm{U}_{xx}[i,j] + 2\mathrm{B}[i,j]\,\mathrm{U}_{xy}[i,j] + \mathrm{C}[i,j]\,\mathrm{U}_{yy}[i,j]$$
$$+ \mathrm{D}[i,j]\,\mathrm{U}_x[i,j] + \mathrm{E}[i,j]\,\mathrm{U}_y[i,j] + \mathrm{F}[i,j]\,\mathrm{U}[i,j] = \mathrm{G}[i,j]$$

על ידי שימוש בנוסחאות הנגזרת מהסוג (8)-(15) שפגשנו בפרק הראשון נמיר את כל הנגזרות החלקיות לביטויים בנעלמים $\mathrm{U}[i,j]$ בלבד, ובכך נקבל מערכת משוואות בנעלמים $\mathrm{U}[i,j]$. נציין כי מספר הנעלמים הוא $(N_x + 1) \times (N_y + 1)$ כמספר הצמתים בשריג. לכן למערכת הזו יש

לצרף את המשוואות שנגזרות מתנאי השפה על המלבן D (מסוג דיריכלה, נוימן, או מעורב) בכדי לקבל מערכת משוואות מלאה שתבטיח פתרון יחיד.

6.1 רמת דיוק ויציבות

בהנחה שמתקיימים תנאי משפט הקיום והיחידות עבור משוואה (41) (עם תנאי שפה), מובטח לנו קיום פתרון רציף יחיד $u(x,y)$ עבור המשוואה מעל המלבן D. נחזור שוב על השאלה שבה פתחנו: האם קיימים תנאים המבטיחים קיום של פתרון בדיד $\mathrm{U}[i,j]$ בכל רמת דיוק נדרשת? ניתן למשל לדרוש שהמרחק בין $u(x_i,y_j)$ ו־$\mathrm{U}[x_i,y_j]$ יהיה חסום על ידי ε נתון:

$$\left|\mathrm{U}[x_i,y_j]-u(x_i,y_j)\right|<\varepsilon,\quad i=0,1,2,\ldots,N_x,\quad j=0,1,2,\ldots,N_y$$

נחדד יותר: האם קיימת התכנסות נקודתית $\mathrm{U}[x_i,y_j]\to u(x_i,y_j)$ כאשר גודלי החלוקה N_x, N_y שואפים לאינסוף? דרישת **היציבות (stability)** מציבה אתגר יותר קשיח: האם קיימים תנאים שמבטיחים שככל שהחלוקה יותר מעודנת (ערכים גדולים יותר של N_x, N_y) כך גם המרחק בין U ל־u מתקצר? התשובה לשאלות אלה תלויה בסוג המשוואה ובתנאים הנלווים לה. ברוב המקרים המוכרים לנו התשובה חיובית, אך לא נציג כאן ניסוחים או הוכחות לכך, מלבד כמה דוגמאות וקישורים למקורות.

במשוואת החום והגלים הדו-מימדיות, פגשנו את התנאי של Courant/Friedrich/Lewy (CFL) שמבטיח יציבות אם הפרמטרים של החלוקה של השריג מקיימים אותו. נזכיר למשל כי בפרק 2 ראינו שתנאי יציבות מספיק עבור משוואת החום מעל מוט באורך L, למשך זמן T, הוא

$$\gamma=\frac{\alpha\Delta t}{\Delta x^2}\le 0.5$$

כאשר $\Delta x=\frac{L}{N}$, $\Delta t=\frac{T}{N_t}$. ההסבר לכך כלול במקור [8].

תנאי יציבות עבור משוואת החום הדו-מימדית הוא

$$\gamma=\frac{\alpha\Delta t}{\Delta x^2}+\frac{\alpha\Delta t}{\Delta y^2}\le 0.5$$

כאשר $\Delta x=\Delta y$ (כפי שזה לרוב), התנאי הופך להיות $\gamma=\frac{\alpha\Delta t}{\Delta x^2}\le 0.25$.

תנאי יציבות עבור משוואת הגלים החד-מימדית היא

$$\gamma=\frac{c\Delta t}{\Delta x}\le 1$$

כאשר c הוא הקבוע הפיזיקאלי של המשוואה. תנאי יציבות עבור משוואת הגלים הדו-מימדית הוא

$$\left(\frac{c\Delta t}{\Delta x}\right)^2+\left(\frac{c\Delta t}{\Delta y}\right)^2\le 1$$

עבור שריג אחיד ($\Delta x=\Delta y$) תנאי היציבות הוא: $\gamma=\frac{c\Delta t}{\Delta x}\le\frac{\sqrt{2}}{2}$.

הוכחת יציבות עבור משוואה דיפרנציאלית רגילה כללית מסדר ראשון ניתן למצוא בפרק 8 (משפט 8.2) של [21]. מומלץ להציץ במקרה הזה מאחר והוא כולל רעיון אופייני להוכחות יציבות גם עבור סוגי מד"ח.

תנאי יציבות דומים קיימים עבור משוואות דיפרנציאלית חלקיות לסוגיהן. נסתפק בדוגמאות שמנינו לעיל. תלמידים המעוניינים להעמיק יותר בנושא ההתכנסות והיציבות עבור מד"ח מסדר גבוה יותר, מוזמנים לעיין בסיכומי הרצאות [16], [8], [11] (סעיף 1.4.7), ובמאמר [19] הדן בתנאי CFL.

6.2 פתרונות בדידים חריגים

בנוסף לקירובים לפתרונות אמיתיים מדויקים, קיימים גם פתרונות בדידים חריגים שאינם קירובים לפתרון אמיתי כלשהו. זה קורה במקרים של משוואות ותנאי שפה שאינם בהכרח פתירים מבחינה מתמטית טהורה, אך למרות זאת ניתן לייצר עבורן משהו קרוב לפתרון במובן נומרי. דוגמה כזו פגשנו בתחילת פרק 2 במשוואת החום (23). משוואה זו כוללת תנאי שפה סותרים בנקודות קצה[10], ולכן יש קפיצות (אי-רציפות) בנקודות אלה שמונעות קיום של פתרון מתמטי מדויק על בסיס משפטי הקיום והיחידות הידועים. למרות זאת, שיטת ההפרשים הסופיים תעבוד גם במקרים אלה ותניב פתרונות נומריים בדידים די טובים במובנים מעשיים, ואפילו בעלי חשיבות פרקטית בתעשיה הטכנולוגית.

קיום פתרונות בדידים חריגים עשוי להשמע כבעייתי, אך מדובר למעשה ביתרון נוסף של שיטת ההפרשים הסופיים, מאחר ואי-תאימות בין תנאי שפה עשוי להתרחש דוקא במודלים של תרחישים מעשיים, והשיטה הנומרית היא היחידה שתניב לנו פתרונות מעשיים במקרים כאלה.

7. נספח: הבסיס התאורטי למשוואת BSM

7.1 תהליכים סטוכסטיים

הבסיס התאורטי העיקרי למשוואת בלק-שולס-מרטון נמצא בלב ענף מיוחד בתורת ההסתברות שנקרא חקר תהליכים סטוכסטיים. הנחת המוצא במודל של בלק-שולס-מרטון היא שפונקציית המחיר של נכס פיננסי היא למעשה תהליך סטוכסטי.

תהליכים סטוכסטיים (Stochastic processes) הוא ענף חשוב בתורת ההסתברות החוקר את החוקים ודפוסי ההתנהגות של תהליכים אקראיים כגון תנועת מולקולה של גז (Brownian Motion), דיפוזיה, פיזור חום, גידול של אוכלוסיות חיידקים, תנועת מחיר נכס פיננסי, וכדומה. זהו נושא רחב יריעה שמוסדות אקדמיים רבים מקדישים לו קורס שלם ולכן מומלץ לקבל התרשמות בלבד מהקישורים הנ"ל.

[10] הסתירה נפתרת על ידי כך שהתנאים מוזנים לאלגוריתם על פי סדר הופעתם, ולכן הערך הסופי של הפונקציה נקבע על ידי תנאי השפה האחרון שמוזן לאלגוריתם שנוגע לנקודה זו.

במסגרת המצומצמת שלנו נסתפק באיפיון הקצר הבא: תהליך סטוכסטי $S = S(t)$ הוא פונקציה תלוית זמן t המתאפיינת על ידי המשוואה הדיפרנציאלית הסטוכסטית הבאה

$$dS = \mu S\, dt + \sigma S\, dW \tag{42}$$

כאשר $W = W(t)$ הוא תהליך וינר (Wiener process), שהוא הרכיב האקראי לחלוטין של התהליך הסטוכסטי, והערך הצפוי $\mu S\, dt$ של נדידת (drift) הערך S. במקרה של מחיר נכס מדובר בריבית או בקצב עלייה (או ירידה) צפוי של ערך הנכס בסביבת השוק שבו הנכס נסחר. כלומר מדובר בשילוב בין תהליך בעל חוקיות צפויה לתהליך אקראי לחלוטין.

ננסה לפרט יותר: תהליך סטוכסטי מורכב משני רכיבים

א. רכיב רציונלי: $\mu S\, dt$

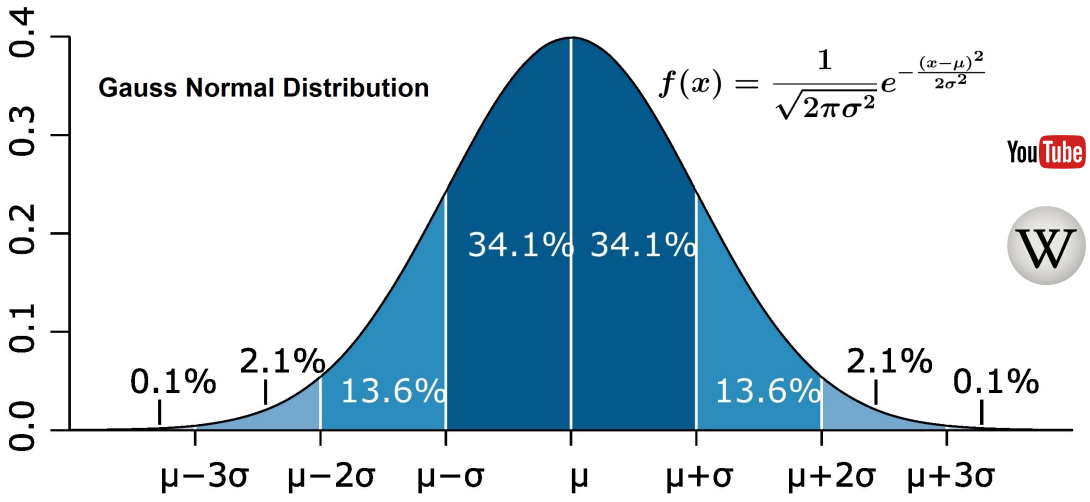
תהליך השינוי (או הנדידה) הצפוי שלו. הערך μ מציין קבוע גידול צפוי של המשתנה $S = S(t)$. במקרה של נכס פיננסי מדובר בקצב גידול צפוי של השוק שבו הנכס נסחר (שנצפה לאורך תקופה היסטורית משמעותית). לכן הביטוי $\mu S\, dt$ מחשב את השינוי הצפוי של S בקטע הזמן dt (ללא המרכיב האקראי).

ב. רכיב אקראי: $\sigma S\, dW$

לרכיב זה שלושה חלקים: S הוא המחיר של הנכס בנקודת זמן t, σ הוא רמת התנודתיות או הרגישות (volatility) של הנכס, ו־$W(t)$ הוא תהליך שינוי אקראי (noise) שבדרך כלל נקרא תהליך וינר (Wiener process). שם נוסף לתהליך זה הוא תנועה בראונית (Brownian motion) (במובן היותר רחב מזה של תנועה מולקולרית אקראית). הרכיב $\sigma S\, dW$ הוא אם כן הרכיב האקראי (noise) שמתווסף לרכיב הקודם. למשל, מלבד קצב גידול ידוע מראש, אשר השפעתו ברורה וצפויה בסבירות גבוהה, מחירו של נכס פיננסי (כגון מנייה, מוצר חשמלי או מזון) עשוי להיות מושפע מעוד אלפי גורמים כגון: מצב חברתי/תרבותי, מצב פוליטי, מצב גאופוליטי, התנהגות של חברות מתחרות, מצבי רוח ויחסים בין דירקטורים ומנהלי ייצור, אירועי מזג אוויר, סיכסוכים אישיים, משברים נפשיים, וכדומה. אין שום דרך מעשית לכמת ולחזות את ההשפעה של גורמים אלה על מחיר הנכס, והדרך הכי פשוטה למדל (modeling) את הרכיב הזה היא באמצעות פונקציה אקראית $W(t)$ שנקראת תהליך וינר. זוהי פונקציה אקראית לחלוטין של הזמן t (מתקדמת בצעדים אקראיים לחלוטין). לכל יחידת זמן Δt, התוספת ΔW היא מספר ממשי אקראי לחלוטין בקטע $[-\Delta t, \Delta t]$, שאינה תלויה בהיסטוריה הקודמת של $W(t)$. לכל ערך ממשי של ΔW יש הסתברות שההתפלגות שלה היא התפלגות נורמלית עם שונות (Variance) Δt, ולכן סטיית התקן (Standard Deviation) היא $\sqrt{\Delta t}$.

באיור 29 רואים כיצד נראה גרף הצפיפות של ההתפלגות הנורמלית של גאוס בעלת ערך מרכזי (כלומר ממוצע) μ וסטיית תקן σ. מאחר וזהו קורס משוואות דיפרנציאליות, לא נוכל להרחיב מעבר לכך, ואנו ממליצים ללמוד יותר על נושא זה על ידי לחיצה על כפתור הויקיפדיה W

שבצד ימין של האיור, שמספק תמצית טובה שלו.



איור 29: התפלגות נורמלית (פעמון גאוס) עם ממוצע μ וסטיית תקן σ

בנוסף לקישור לדף הויקיפדיה, יש גם קישור לסרטון יוטיוב מעולה מערוץ 3Blue1Brown אשר מסביר את הנושא מנקודת המבט של עזר לימודי שנקרא לוח גלטון (Galton Board). קורסי הסתברות וסטטיסטיקה רבים משתמשים בעזר הלימודי הזה בכדי להמחיש את עקרונות היסוד של תורת ההסברות וההתפלגות הנורמלית. מומלץ מאוד לצפות בסרטונים אלה בכדי להבין לעומק את מושג האקראיות שבלב תהליך וינר.

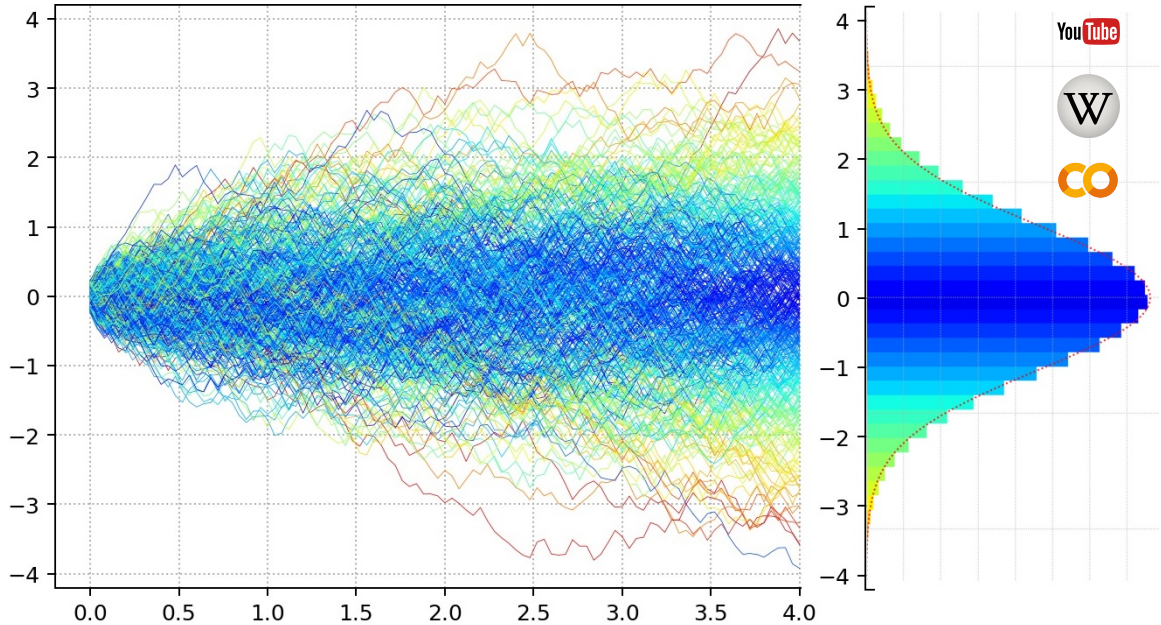
7.2 רקע סטטיסטי

בצד שמאל של איור 30 מוצגים 400 גרפים של פונקציות אקראיות מהקטע $[0,4]$ לטווח $[-4,4]$. הצבע של כל גרף נקבע על פי הערך הסופי שלו בנקודה האחרונה $t=4$ (על פי מפת צבעים רגילה: כחול קרוב לאפס, אדום קרוב ל־4 או -4). לא קשה להבחין כי ההתפלגות של הגרפים מזכירה את הפעמון של גאוס (התפלגות נורמלית). לאחר שצפינו בסרטונים על לוח גלטון, נוכל להבחין כי הגרפים הם למעשה מסלולים אפשריים של כדור על לוח גלטון אופקי ורציף.

הערכים של כל פונקציה אקראית הם בטווח $[-4,4]$. בכל נקודת זמן t בקטע $[0,4]$, הערך של $f(t+\Delta t)$ נקבע על ידי הערך $f(t)$ בתוספת של ערך אקראי מתוך הקטע הסגור $[-\Delta t,\Delta t]$. כל אחת מהפונקציות התקבלה ברקורסיה באמצעות הקוד הבא:

$$f(t+\Delta t) = f(t) + \mathrm{Random}(-\Delta t,\Delta t)$$

הפונקציה $\mathrm{Random}(-\Delta t,\Delta t)$ מחזירה ערך אקראי בקטע $[-\Delta t,\Delta t]$, ולכן בכל פעם שנבנה f כזו נקבל פונקציה חדשה (בסבירות מאוד גבוהה). מיון הפונקציות האקראיות מתבצע על פי הערך הסופי של הגרף בנקודה $t=4$ ("זמן הבשלות של האופציה"). נציין כי ההגדרה הפורמלית של פונקציית וינר היא פונקציה אקראית עבור כל $\Delta t>0$! מההגדרה לא ברור איך פונקציות כאלה נראות ואיזה תכונות יש להן? למשל, ניתן להוכיח שהן רציפות "ברוב"



איור 30: 400 גרפים אקראיים $f : [0,4] \to [-4,4]$

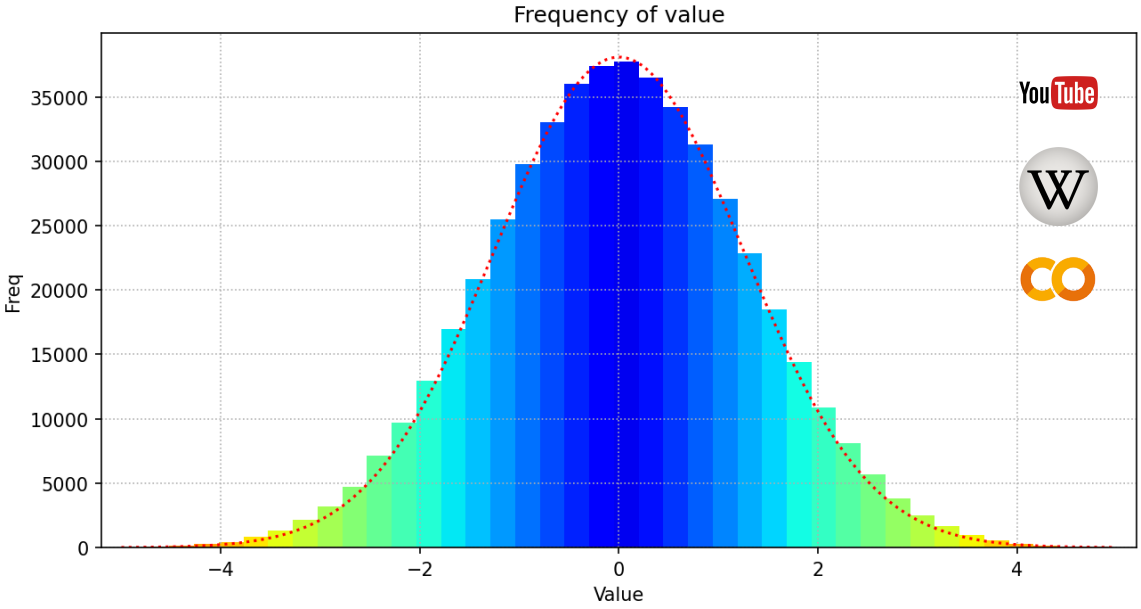
הנקודות (מלבד קבוצה ממידה אפס), אך אינן גזירות "ברוב" הנקודות. לא נוכל לפרט יותר מזה במסגרת המוגבלת שלנו.

פונקציה $W(t)$ מהסוג הנ"ל היא הגרעין של הרכיב האקראי של תהליך סטוכסטי: החלק האקראי של תהליך סטוכסטי מתקבל על ידי הכפלת $W(t)$ במקדם רגישות (volatility) σ ומחיר נכס $S(t)$. תהליך וינר מתאפיין בכך שההתפלגות של הפונקציה Random היא **נורמלית**. בצד ימין של איור 30 וגם באיור 31 מוצגת דיאגרמת העמודות שמפלחת את הגרפים האקראיים ל־50 סלים על פי ערך המטרה שלהם $f(4)$. דיאגרמת העמודות משמשת גם כמפת צבעים.

הדיאגרמה באיור 31 מציגה התפלגות של $500,000$ פונקציות אקראיות בקטע $[0,4]$. בחרנו בכמות הגדולה הזו בכדי לוודא שאכן דיאגרמת העמודות הולכת ונעשית קרובה לפעמון של גאוס. הגרפים סווגו ל־50 סלים שונים על פי ערך מטרה שלהם $f(4)$. רואים למשל שרוב הפונקציות משתייכות לסלים הכחולים בסביבת אפס.

7.3 הלמה של איטו (Kiyoshi Itô 1915-2008)

שם נוסף, אך יותר כללי לתהליך הסטוכסטי שתארנו לעיל הוא תהליך איטו הקרוי על שמו של המתמטיקאי היפני Kiyoshi Itô ("The most famous Japanese in Wall Street"). המתמטיקאי היפני **קיושי איטו** הקדיש חלק ניכר מעבודתו המחקרית לתהליכים סטוכסטיים בשוק הפיננסי ופיתח ענף מתמטי שלם העוסק בחשבון דיפרנציאלי ואינטגרלי של פונקציות סטוכסטיות. הפונקציות הסטוכסטיות מתאפיינות על ידי גרף שמתנהג בצורה אקראית ולכן אינן גזירות במובן הרגיל, ולכן לא ניתנות להבנה באמצעות הכלים הרגילים של החדו"א הקלאסי (למעשה מושג הרציפות אינו חל עליהם במובן המוכר לנו). אחת התוצאות הכי חשובות של המחקר שלו עבור המודל של בלק-שולס-מרטון היא "**כלל שרשרת סטוכסטי**" שנקרא הלמה של איטו שנשתמש בה עבור בניית המשוואה הדיפרנציאלית של BSM. כזכור, כלל שרשרת מאפשר לנו לחשב נגזרת של



איור 31: ההתפלגות הגרפים האקראיים על פי הערך הסופי היא נורמלית

פונקציה מורכבת. במקרה שלנו, אנו מעוניינים לגזור את פונקציית ערך האופציה $V(S,t)$ כפונקציה מורכבת $V(S(t),t)$, כאשר $S(t)$ הוא מחיר המנייה בזמן t.

משפט 1: (הלמה של איטו) יהי $x = x(t)$ תהליך איטו נתון. כלומר קיימות פונקציות $a(x,t)$, $b(x,t)$ כך ש־

$$dx = a(x,t)dt + b(x,t)dW \tag{43}$$

תהי $y = f(x,t)$ פונקציה גזירה ברציפות בשני המשתנים. אזי

$$dy = \left[\frac{\partial f}{\partial x}a(x,t) + \frac{\partial f}{\partial t} + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}b(x,y)^2\right]dt + \frac{\partial f}{\partial x}b(x,t)dW \tag{44}$$

תהליך איטו $x(t)$ מתאפיין על ידי המשוואה (43) שבה $a(x,t)$, $b(x,t)$ הן פונקציות רציפות של המשתנה x ומשתנה זמן t, ואילו $W(t)$ הוא תהליך וינר. כמו בתהליך סטוכסטי, הרכיב הראשון $a(x,t)$ הוא הרכיב הרציונלי הצפוי, בעוד שהרכיב $b(x,t)dW$ הוא הרכיב האקראי. החשיבות של הלמה של איטו, שהיא כאמור סוג של כלל שרשרת, שהיא שומרת על ההפרדה הברורה של נגזרת התהליך הסטוכסטי המורכב $f(x(t),t)$ לשני רכיבים: רכיב אחד בעל סדר וחוקיות הניתנת לטיפול באמצעות החשבון הניוטוני הקלאסי, ורכיב שני אקראי לחלוטין שיטופל באמצעות היגיון סטטיסטי/הסתברותי (Itô calculus).

לא נוכל להוכיח את הלמה הזו במסגרת המוגבלת של הקורס. נציין רק שההוכחה מתבססת על הפיתוח לטור טיילור של פונקציה $f(x,t)$ בשני משתנים. הנוסחה העיקרית (44) מורכבת משני חלקים: החלק הרציונלי (זה שבתוך הסוגריים המרובעות) הוא למעשה שלושת האיברים הראשונים של הטור טיילור של $f(x,t)$, והחלק האקראי $\frac{\partial f}{\partial x}b(x,t)dW$. הוכחה מלאה של הלמה של איטו ניתן למצוא בסעיף 5.4 של הספר [15].

אם נפעיל את הלמה של איטו על הפונקציה $V(S,t)$ במשוואה (42) נקבל

$$dV = \left[\frac{\partial V}{\partial S}\,\mu S + \frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right] dt + \frac{\partial V}{\partial S}\,\sigma S\,dW \tag{45}$$

המשוואה שקיבלנו נקראת משוואה דיפרנציאלית סטוכסטית, והיא די מסובכת לפתרון בגלל הרכיב האקראי dW שאין לנו שום כלים לטפל בו במסגרת שלנו.

7.4 התימרון של בלק/שולס/מרטון

הרעיון המבריק של בלק/שולס/מרטון (אשר עליו זכו בפרס נובל לכלכלה) היה לנטרל את הרכיב האקראי באמצעות בניית תיק השקעות תאורטי שמשלב בתוכו נגזרות (אופציות) ומניות רגילות באופן כזה שהתיק מאוזן לחלוטין בכל נקודת זמן (ללא הפסד או רווח מלבד הצמדה לריבית צפויה r). הרעיון הוא לרכוש אופציה על מנייה נתונה וכמות מאזנת של יחידות רגילות של אותה מנייה באופן כזה שהפסד ברכישה אחת מתקזז על ידי רווח ברכישה השנייה. ליתר דיוק, הרכיב האקראי של רכישה אחת יתבטל על ידי הרכיב האקראי של הרכישה השנייה, וכתוצאה מכך הרכיב האקראי של התיק בכללותו יעלם מהמשוואה! לכן המשוואה הסטוכסטית שלנו תהפוך למשוואה רגילה שאולי נוכל לפתור באמצעות השיטות המתמטיות הרגילות.

נציג תאור סכימטי בלבד של הרעיון מבלי להיכנס לפרטים הטכניים הקטנים. יהי $S(t)$ מחיר המנייה בזמן t, $V(S,t)$ מחיר אופציית מכר (PUT) של מנייה במחיר מטרה S בזמן t. התיק הספקולטיבי שלנו יורכב משתי רכישות בלבד:

א. רכישה ארוכת טווח של אופציית מכר (PUT) אחת במחיר מטרה $S(0)$ (כלומר המחיר הנוכחי של המנייה בזמן ההתחלה $t=0$),

ב. אחזקה קבועה של $\alpha = \frac{\partial V}{\partial S}$ מניות רגילות בכל רגע נתון t.

ההנחה התאורטית היא שניתן לרכוש או למכור חלקים לא שלמים של מניה בכל רגע נתון, שכן הערך של α עשוי להיות כל מספר ממשי, כולל לא-רציונלי! הנחה תאורטית נוספת היא שאין עמלות על ביצוע פעולות קניה ומכירה, אחרת אין שום היתכנות לסימולציה שלנו.

הערך Π של תיק כזה בכל רגע נתון הוא

$$\Pi = V - \alpha S \tag{46}$$

השינוי הרגעי של Π הוא[11]

$$d\Pi = dV - \alpha\,dS \tag{47}$$

[11] הגזירה בכל המקרים היא על פי המשתנה t. אבל השוויון הזה לא לגמרי מנומק משום שהתעלמנו מכך שגם $\alpha = \frac{\partial V}{\partial S}$ היא פונקציה של t! לכן היה צריך לגזור את המכפלה αS על פי כלל המכפלה. ההנמקה המלאה לכך שניתן להתעלם מהגזירה של α מוסברת באריכות בסעיף 5.6 של הספר [15].

השילוב של (42), (45) ו־(47) נותן

$$\begin{aligned} d\Pi &= -\alpha\, dS + \left(\alpha\,\mu S + \frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt + \alpha\,\sigma S\, dW \\ &= -\alpha\,(\mu S\, dt + \sigma S\, dW) + \left(\alpha\,\mu S + \frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt + \alpha\,\sigma S\, dW \\ &= \left(\frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt \end{aligned}$$

קיבלנו לכן

$$d\Pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt \tag{48}$$

כפי שקיווינו, הרכיב האקראי dW נעלם מהמשוואה! לכן היא כבר לא סטוכסטית! כלומר השינוי בשווי התיק אינו תלוי ברכיב האקראי dW[12]. נציין שיש לעדכן את התיק בכל יחידת זמן Δt מאחר והנגזרת $\alpha = \frac{\partial V}{\partial S}$ משתנה בכל רגע נתון. כלומר, בכל יחידת זמן Δt יש לוודא שכמות המניות בתיק שווה $\alpha = \frac{\partial V}{\partial S}$ באותו הרגע (על ידי רכישה או מכירת חלקי מניות). כמובן שמדובר בניסוי מחשבתי בלבד, מאחר ובשוק האמיתי לא ניתן לרכוש חלקים לא שלמים של אופציה בכל רגע נתון.

בהנחה שהתנהגות השוק נורמלית לחלוטין (אין מפולות, הזדמנויות ארביטראז', משבר כלכלי וכדומה), השינוי הצפוי בשווי השוק צריך להיות תואם לגובה הריבית r הצפוי בסביבת השוק. כלומר השינוי הצפוי הוא $d\Pi = r\Pi dt$, ולכן על סמך שוויון (48) נוכל לרשום

$$r\Pi dt = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt$$

על סמך שוויון (46)

$$r\Pi dt = r\left(V - \frac{\partial V}{\partial S}\,S\right) dt = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\,\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2\right) dt$$

ומכאן נקבל את משוואת בלק-שולס-מרטון

$$\frac{\partial V}{\partial t} + \frac{1}{2}\,\frac{\partial^2 V}{\partial S^2}\,\sigma^2 S^2 + \frac{\partial V}{\partial S}\,rS - rV = 0$$

זוהי משוואה הרבה יותר פשוטה ממשוואה (45), שניתנת לטיפול באמצעות כלים רגילים מתחום המשוואות הדיפרנציאלית החלקיות. מסתבר שעבור אופציות אירופיות קיים פתרון אנליטי שמתקבל על ידי הצבה שממירה אותה למשוואת החום. ראה עמוד 7 של המאמר [13], סעיף 4 של המאמר [14], וסעיף 5.8 של הספר [15]. עבור אופציות אמריקאיות לא קיים פתרון כזה, והפתרון היחיד שידוע הוא פתרון נומרי שמתקבל באמצעות שיטת ההפרשים הסופיים כפי שהצגנו קודם.

[12] ליתר דיוק הוא נבלע בתוכה כתוצאה מכך שהתיק שלנו מורכב משני נכסים אשר הרכיבים האקראיים שלהם מבטלים אחד את השני.

אנו מודים מאוד לד"ר דניאל הקמן (Daniel Hackmann) אשר מאמרו הבהיר [13] מסביר את הנושא בצורה הסכימטית הכי טובה ביחס לכל המקורות שבדקנו (לאחר חיפושים רבים).

8. ביבליוגרפיה

1. **Numerical Partial Differential Equations: Finite Difference Methods.**
   James W. Thomas. Springer; 1995th edition (September 11, 1995).
   https://link.springer.com/book/10.1007/978-1-4899-7278-1
   https://www.amazon.com/-/he/J-W-Thomas/dp/0387979999
2. **Introductory Finite Difference Methods for PDEs.**
   D. M. Causon, C. G. Mingham. University of Manchester, England, 2010).
   https://www.cs.man.ac.uk/~fumie/tmp/introductory-finite-difference-methods-for-pdes.pdf
   https://www.bookboon.com
3. **Solving 2D Heat Equation Numerically using Python.** *G. Nervadof.* Level Up Coding.
   https://levelup.gitconnected.com/solving-2d-heat-equation-numerically-using-python-3334004aa01a
4. **Explicit Finite Difference with Black-Scholes Formula (with code).**
   Antoni Smolski. Medium.com Blog.
   https://antonismolski.medium.com/implementation-of-black-scholes-formula-using-finite-difference-method-with-code-965fd0539808
5. **Option pricing using the Black-Scholes model, without the formula.**
   R. Diter. Medium.com Blog.
   https://diter.medium.com/option-pricing-using-the-black-scholes-model-without-the-formula-e09235f75fb7
6. **C++ Explicit Euler Finite Difference Method for Black Scholes.**
   QuantStart Team. QuantStart Newsletter.
   https://www.quantstart.com/articles/C-Explicit-Euler-Finite-Difference-Method-for-Black-Scholes
7. **Solution of the Black-Scholes Equation by Finite Difference Schemes.**
   Ganga Ram, Kedar Nath Uprety, Harihar Khanal. Nepal Journals Online.
   Google Indexed Nepal Journal Paper
8. **Finite-Difference Method for the 1D Heat Equation.** *Padmanabhan Seshaiyer.*
   George Mason University. Dept. of Mathematical Sciences.
   https://math.gmu.edu/~pseshaiy/F12/m679/m679_F12_notes_hw1.pdf
9. **The 2D wave equation.** *Ingo Berg.* Recreational Programming.
   https://beltoforion.de/en/recreational_mathematics/2d-wave-equation.php
10. **Finite difference methods for 2D and 3D wave equations.** *Hans Petter Langtangen.*
    https://hplgit.github.io.

https://hplgit.github.io/num-methods-for-PDEs/doc/pub/wave/sphinx/._main_wave005.html

https://hplgit.github.io/num-methods-for-PDEs/doc/pub/wave/sphinx

11. **Finite Difference Computing with PDEs - A Modern Software Approach.** *Hans Petter Langtangen, Svein Linge.* Springer Link.

https://link.springer.com/book/10.1007/978-3-319-55456-3

https://hplgit.github.io/fdm-book/doc/pub/book/pdf/fdm-book-4print.pdf

12. **Programming for Computations - A Gentle Introduction to Numerical Simulations with Python.** *Svein Linge, Hans Petter Langtangen.* Springer Link.

https://link.springer.com/book/10.1007/978-3-319-32428-9

https://hplgit.github.io/prog4comp/doc/pub/p4c_Python.pdf

13. **Solving the Black Scholes Equation using a Finite Difference Method.** *Daniel Hackmann.* Personal Webpage.

http://www.danhackmann.com/BlackScholes7.pdf

http://samyzaf.com/fdm/BlackScholes7.pdf

14. **Stochastic Calculus and the Black–Scholes–Merton Model.** *Noah Fischer.* Univ. of Chicago.

https://math.uchicago.edu/~may/REU2021/REUPapers/Fischer.pdf

15. **The Concepts of Mathematical Finance.** *M. S. Joshi.* Cambridge University Press.

https://assets.cambridge.org/97805215/14088/frontmatter/9780521514088_frontmatter.pdf

16. **Finite Difference Methods for Differential Equations.** *Randall J. LeVeque.* Dept. of Applied Math. University of Washington.

https://edisciplinas.usp.br/pluginfile.php/41896/mod_resource/content/1/LeVeque%20Finite%20Diff.pdf

17. **An Introduction to Partial Differential Equations.** *Y. Pinchover, J. Rubinstein.* Cambridge University Press, 4th printing, 2008.

18. **Partial Differential Equations (4th Edition).** *Fritz John.* Springer 1971, ISBN 10: 0387900217, ISBN 13: 9780387900216.

19. **Understanding the Importance of the CFL Condition in CFD Simulations.** *Anthony Massobrio.* NEURAL CONCEPT (BLOG).

https://www.neuralconcept.com/post/understanding-the-importance-of-the-cfl-condition-in-cfd-simulations

20. **DIFFERENTIAL EQUATIONS.** *Paul Dawkins.* Lamar University.

https://tutorial.math.lamar.edu/Classes/DE/DE.aspx

21. **Ordinary Differential Equations (Hebrew textbook).**

Samy Zafrany. Personal Webpage.

https://samyzaf.com

22. Fourier Series and Integral Transforms (Hebrew textbook).

Samy Zafrany. Personal Webpage.

https://samyzaf.com

23. 2D Wave Equation Simulation using Python. *Abel Flores Prieto.* YouTube.

https://www.youtube.com/watch?v=iS_msvfhB1I

https://github.com/abelfp/wave_equation_simulation

Dedicated to the memory of James Harris Simons
April 25, 1938 – May 10, 2024
A mathematician, and philanthropist.
"The mathematician that cracked Wall Street"