מבוא לחישוב נומרי

הכנה לקראת המבחן

סמסטר א תש"ע
סיכומי גן-עדן
דינה זליגר

Please read the following important legal information before reading or using these notes. The use of these notes constitutes an agreement to abide by the terms and conditions below, just as if you had signed this agreement.

A. THE SERVICE.

The following notes ("The service") are provided by DinaZil's Notes-Heaven ("Notes-Heaven").

B. DISCLAIMER OF WARRANTIES; LIMITATION OF LIABILITY.

Notes-Heaven does not endorse content, nor warrant the accuracy, completeness, correctness, timeliness or usefulness of any opinions, advice, content, or services provided by the Service.

YOU AGREE THAT USE OF THE SERVICE IS ENTIRELY AT YOUR OWN RISK. THE SERVICE PROVIDED IS PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND. NOTES-HEAVEN EXPRESSLY DISCLAIMS ALL WARRANTIES OF ANY KIND, WHETHER EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION: ANY WARRANTIES CONCERNING THE ACCURACY OR CONTENT OF INFORMATION OR SERVICES. NOTES-HEAVEN MAKES NO WARRANTY THAT THE SERVICE WILL MEET YOUR REQUIREMENTS, OR THAT THE SERVICE WILL BE ERROR FREE; NOR DOES NOTES-HEAVEN MAKE ANY WARRANTY AS TO THE RESULTS THAT MAY BE OBTAINED FROM THE USE OF THE SERVICE OR AS TO THE ACCURACY OR RELIABILITY OF ANY INFORMATION OBTAINED THROUGH THE SERVICE. YOU UNDERSTAND AND AGREE THAT ANY DATA OBTAINED THROUGH THE USE OF THE SERVICE IS DONE AT YOUR OWN DISCRETION AND RISK AND THAT YOU WILL BE SOLELY RESPONSIBLE FOR ANY DAMAGE TO YOUR GPA.

NEITHER NOTES-HEAVEN NOR ANY OF ITS PARTNERS, AGENTS, AFFILIATES OR CONTENT PROVIDERS SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OF THE SERVICE OR INABILITY TO GAIN ACCESS TO OR USE THE SERVICE OR OUT OF ANY BREACH OF ANY WARRANTY.

C. INDEMNIFICATION.

You agree to indemnify and hold Notes-Heaven, its partners, agents, affiliates and content partners harmless from any dispute which may arise from a breach of terms of this Agreement. You agree to hold Notes-Heaven harmless from any claims and expenses, including reasonable attorney's fees and court costs, related to your violation of this Agreement.

D. OWNERSHIP RIGHTS.

The materials provided by the Service may be downloaded or reprinted for personal use only. You acknowledge that the Service contains information that is protected by copyrights, trademarks, trade secrets or other proprietary rights, and that these rights are valid and protected in all forms, media and technologies existing now or hereafter developed. You may not modify, publish, transmit, participate in the transfer or sale, create derivative works, or in any way exploit, any of the Content, in whole or in part. You may not upload, post, reproduce or distribute Content protected by copyright, or other proprietary right, without obtaining permission of the owner of the copyright or other proprietary right.

E. NO COPYING OR DISTRIBUTION.

You may not reproduce, copy or redistribute the design or layout of this service, individual elements of the design, Notes-Heaven logos or other logos appearing on this service, without the express written permission of Notes-Heaven, Inc. Reproduction, copying or redistribution for commercial purposes of the service is strictly prohibited without the express written permission of Notes-Heaven, Inc.

If you have any questions about this statement or the practices of this service you can contact

Dina Zeliger
dinazil@notes-heaven.com

תוכן עניינים

אינטרפולציה ....2..............
שיטות נאיביות ....2..............
שיטת Neville ....3..............
ניתוח איכות ....4..............
אינטרפולציות פולינומיאליות למקוטעין – Splines ....4..............
Natural Splines ....4..............
אינטרפולציה על סריגים אחידים ....5..............
Least Squares Approximate Interpolation ....6..............
אינטרפולציה רב-ממדית ....7..............
Radial Basis Functions ....7..............
אינטגרציה ....9..............
שיטת הטרפז ....9..............
שיטת סימפסון ....9..............
אינטגרציה רב-ממדית ....10..............
אינטגרציית מונטה-קרלו ....10..............
Rejection Sampling ....11..............
מערכות משוואות לא ליניאריות ....12..............
שיטת החצייה ....12..............
שיטת Newton-Raphson ....12..............
מערכות משוואות ליניאריות ....14..............
שיטות ישירות ....14..............
אלימינציה של גאוס ....14..............
פירוק LU ....14..............
פירוק Cholesky ....15..............
שיטות איטרטיביות ....16..............
מטריצות פירוק ....16..............
Steepest Descent ....19..............
Conjugate Gradients ....20..............
Preconditioning ....21..............
Transformed Preconditioned Conjugate Gradients ....22..............
Untransformed Preconditioned Conjugate Gradients ....22..............
ערכים ווקטורים עצמיים ....24..............

שיטת החזקות 24
שיטת החזקות ההפוכה 25
Jacobi Iteration 25
שיטת ארנולדי 26
שיטת Lanczos 27
פירוק QR 27
פירוק QR באמצעות תהליך גראם-שמידט 28
פירוק QR באמצעות Householder Reflection 28
פירוק QR באמצעות מטריצות Givens 28
אופטימיזציה 30
Bracketing 30
Downhill Simplex 30
Line Minimization 31
שימוש בנגזרות 31
Simulated Annealing 32
משוואות דיפרנציאליות רגילות 33
שיטות אוילר 33
שיטת Mid-Step 34
נספח א – פתרונות למבחנים 35
2008 מועד א 35
2008 מועד ב 37
נספח ב – סיכום 40
אינטרפולציה 40
אינטגרציה 42
מערכות משוואות לא ליניאריות 43
מערכות משוואות ליניאריות 44
ערכים ווקטורים עצמיים 49
אופטימיזציה 51
משוואות דיפרנציאליות רגילות 52

אינטרפולציה

__הבעיה__: נתונה קבוצה סופית של ערכים של נקודות על גרף של פונקציה $\{(x_i, y_i)\}_{i=1}^{M}$ כאשר $x_i \in [a,b]$ לכל $1 \le i \le M$. בהינתן $x \in \mathbb{R}$ נרצה לשערך את $f(x)$. אם $x \in [a,b]$ הבעיה נקראת **אינטרפולציה** ואם $x \notin [a,b]$ הבעיה נקראת **אקסטרפולציה**.

אנחנו כמובן מניחים שהפונקציה רציפה, אחרת אין דרך לשערך אותה.

בגדול, לפיתרון יש שני שלבים:

1. **תפירה** (fitting) – מציאת פונקציה $\hat{f}$ כך ש-$\hat{f}(x_i) = y_i$ לכל $1 \le i \le M$. פונקציה זו נקראת **אינטרפולנט**.
2. **הערכה** (evaluation) – חישוב של $\hat{f}(x)$.

ההנחה היא ש-$\hat{f}$ זו היא קירוב טוב של הפונקציה המקורית ולכן הערכה שלה ב-x נותנת קירוב טוב למה שאנחנו רוצים.

בדרך כלל מחפשים את האינטרפולנט מתוך משפחת פונקציות שתלויות בפרמטרים כלשהם $c_1, \dots, c_k$. אז הבעיה היא למצוא פרמטרים כאלה כך ש-$\hat{f}(x_i, c_1, \dots, c_k) = y_i$ לכל $1 \le i \le M$.

__דוגמה__: נתון $f(1) = 1, f(2) = 2$ ואנחנו מחפשים אינטרפולנט $\hat{f}(x, c_1, c_2) = xc_1^2 + x^2 c_2$. כל הצבה של ערך נתון נותנת לנו אילוץ, כלומר משוואה:

$$1 = f(1) = \hat{f}(1, c_1, c_2) = c_1^2 + c_2$$
$$2 = f(2) = \hat{f}(2, c_1, c_2) = 2c_1^2 + 4c_2$$

את המערכת הזו ניתן לפתור ולקבל את $\hat{f}$. לאחר מכן ניתן יהיה לשערך בכל נקודה שנרצה.

__הגדרה__: נתונות $\{(x_i, y_i)\}_{i=1}^{M}$. **אינטרפולציה ליניארית** $\hat{f}$ היא מהצורה $\hat{f}(x, c_1, \dots, c_M) = \sum_{i=1}^{M} g_i(x) c_i$ כאשר g_i פונקציות כלשהן. כלומר, משפחת האינטרפולנטים ליניארית בפרמטרים שלה. **סדר האינטרפולציה** הוא $M-1$. **אינטרפולציה פולינומיאלית** היא אינטרפולציה ליניארית שבה בוחרים $g_i(x) = x^{i-1}$, כלומר האינטרפולנט הוא פולינום ממעלה $M-1$.

יש משפט שאומר שאם $x_0, \dots, x_M$ נקודות שונות אז לכל $y_0, \dots, y_M$ יש פולינום יחיד p ממעלה M כך ש-$p(x_i) = y_i$ לכל $0 \le i \le M$. ההוכחה של המשפט גם נותנת דרך מפורשת לבנות את הפולינום הזה. אז בעיית האינטרפולציה הפולינומיאלית פתורה.

שיטות נאיביות

התנאים $\hat{f}(x_i) = y_i$ באינטרפולציה ליניארית נותנים את מערכת המשוואות הבאה:

$$\left\{ \sum_{i=1}^{M} g_i(x_j) c_i = y_j \right\}_{j=1}^{M}$$

או בכתיב מטריצות:

$$\begin{pmatrix} g_1(x_1) & & g_M(x_1) \\ & \ddots & \\ g_1(x_M) & & g_M(x_M) \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_M \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_M \end{pmatrix}$$

אם $\begin{pmatrix} g_1(x_1) & & g_M(x_1) \\ & \ddots & \\ g_1(x_M) & & g_M(x_M) \end{pmatrix} \text{im} \ni \begin{pmatrix} y_1 \\ \vdots \\ y_M \end{pmatrix}$ קיים פתרון למשוואה. בפרט, אם האינטרפולציה פולינומיאלית אז נקבל את מערכת המשוואות

$$\begin{pmatrix} 1 & & x_1^{M-1} \\ & \ddots & \\ 1 & & x_M^{M-1} \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_M \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_M \end{pmatrix}$$

זוהי מטריצת ון-דר-מונדה. כמובן, סביר להניח שכל ה-x_i-ים שונות ולכן למערכת המשוואות הזו קיים פיתרון והוא יחיד.

אם נפעל באופן נאיבי לפתרון מערכת המשוואות (למשל ע"י אלימינציה של גאוס) אז שלב התפירה נפתר בסיבוכיות $O(M^3)$ ולאחר שכבר יש לנו את המקדמים של הפולינום, כל הערכה לוקחת זמן $O(M)$ ואין צורך לתפור מחדש.

גישה אחרת לפיתרון הבעיה עוקפת את שלב התפירה ע"י שימוש בפולינום לגרנז' שהוא פשוט הפיתרון הידוע מראש של מערכת המשוואות הזאת:

$$p(x) = \sum_{i=1}^{M} y_i \frac{\prod_{j \neq i}(x - x_j)}{\prod_{j=1}^{M}(x_i - x_j)}$$

כאן עלות ההערכה היא $O(M^2)$ אבל אין את שלב התפירה. כמובן, ניתן היה מהנוסחה הזו לחלץ את המקדמים של הפיתרון אבל זה כרוך בפתיחת סוגריים והעלות תהיה $O(M2^M)$.

שיטת Neville

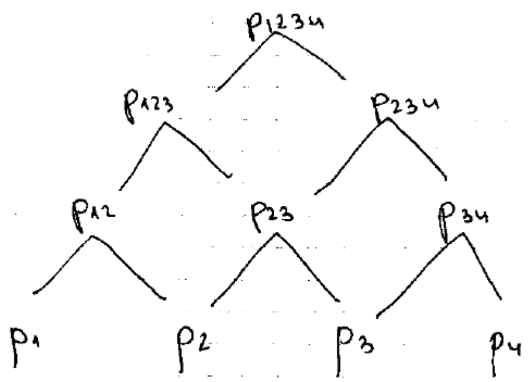
בשתי השיטות שראינו למעלה יש בעיה של יציבות נומרית. בד"כ לא רצוי לחבר מספרים מסדרי גודל שונים כי הדיוק של החישוב נפגע מאוד בגלל אופן הייצוג של המספרים במחשב. בפרט, הבעיה חמורה מאוד כאשר סדר האינטרפולציה גבוה ואז עלול להיות הבדל מאוד גדול בין x ל-x^{M-1}. הרעיון הוא לבנות את הפולינום בשלבים ובכל פעם לעשות חישובים על מספרים בסדרי גודל דומים.

נבנה סדרת פולינומים $p_{i,\ldots,i+m}$ כך שלכל $j = 1, \ldots, m$, $p_{i,\ldots,i+m}(x_j) = y_i$. נגדיר באינדוקציה:

$$p_i(x) = y_i$$

$$p_{i,\ldots,i+m}(x) = \frac{(x - x_i)p_{i+1,\ldots,i+m}(x) + (x_{i+m} - x)p_{i,\ldots,i+m-1}(x)}{x_{i+m} - x_i}$$

ככה נוצר עץ חישוב:



לבסוף, הפולינום $p_{1,\dots,M}(x)$ הוא הפולינום שמעניין אותנו.

גם בשיטה הזו מוותרים על שלב התפירה וניגשים מייד להערכה. עלות ההערכה היא $O(M^2)$, כמו בשיטת הפולינום של לגרנז', אלא שהיתרון כאן הוא הדיוק הנומרי.

ניתוח איכות

משפט: נניח שנתונות נקודות $\{(x_i,y_i)\}_{i=0}^{M}$ על גרף של פונקציה f כך ש-$x_i \in [a,b]$ לכל $0 \le i \le M$ ו-p מתלכדת עם f על נקודות אלה. אם ל-f יש $M+1$ נגזרות בקטע $[a,b]$ אז לכל $x \in [a,b]$ קיים $\xi \in [a,b]$ כך ש-

$$R(x) = f(x) - p(x) = \frac{(x-x_0)(x-x_1)\dots(x-x_M)}{(M+1)!} f^{(M+1)}(\xi) = \frac{w(x)}{(M+1)!} f^{(M+1)}(\xi)$$

אם ניקח את כל הנקודות להיות במרחקים שווים זו מזו, נניח δ אז $w(x) \le M!\,\delta^{M+1}$ ולכן

$$|R(x)| \le \frac{1}{M}\delta^{M+1}\left|f^{(M+1)}(\xi)\right|$$

ולכן נקבל שבקטע $[a,b]$ מתקיים

$$|R(x)| \le K \cdot O(\delta^{M+1})$$

כאשר $K = \max_{\xi\in[a,b]}\left|f^{(M+1)}(\xi)\right|$.

המסקנה היא שאם עניינו באינטרפולציה, כלומר $a = x_0, b = x_M$ ומעניין אותנו לשערך נקודות בתוך הקטע אז ככל שנוסיף נקודות דגימה, המרחק δ יקטן והשגיאה תקטן. אבל זה בהנחה ש-K נשאר קבוע עם הוספת הנקודות. בפועל, בהרבה מקרים נגזרות גבוהות של פונקציות נוטות "להשתגע" ואז החסם שלנו לא מאוד מוצלח. זה לא אומר שהשגיאה תהיה גדולה בפועל אבל הניתוח התיאורטי שלנו לא מאפשר לראות זאת ואין הבטחה שהקירוב יהיה טוב יותר.

אינטרפולציות פולינומיאליות למקוטעין – Splines

ראינו שכאשר מתאימים הרבה נקודות בבת אחת לא הצלחנו להוכיח שבהכרח השערוך יהיה טוב. ניתן לעשות את התפירה לקבוצות קטנות יותר של נקודות. יש שתי אפשרויות:

1. חלוקת הקטע הגדול לתת קטעים זרים התחומים ע"י N נקודות בכל אחד מהם ושימוש באינטרפולציה ממעלה $N-1$ בכל קטע. הפונקציה המתקבלת היא כמובן רציפה אבל היא לא בטוח גזירה בנקודות החפיפה.
2. בחירה שרירותית של N נקודות וחישוב האינטרפולציה רק עליהן (למשל, ניתן לבחור את N הנקודות הכי קרובות לנקודה שרוצים לשערך). אם אנחנו צופים שנרצה לשערך רק בנקודה אחת זו יכולה להיות אפשרות טובה.

Natural Splines

על שתי האלטרנטיבות למעלה, ובעיקר על הראשונה, אולי נרצה לכפות גם אילוצים שקשורים לנגזרת. אם יש לנו מידע גם על הנגזרות של הפונקציה בנקודות הדגימה אפשר לשלב אותן במשוואות. במקרה זה נצטרך להשתמש בפולינומים ממעלה N. למשל, אם נתונות לנו שתי נקודות דגימה:

$$f(x_1) = y_1, f(x_2) = y_2$$
$$f'(x_2) = y_2'$$

ניתן לפתור את הבעיה ע"י פולינום ממעלה שנייה $p(x) = ax^2 + bx + c$. אז נקבל מערכת משוואות לכל אחד מהאילוצים שלנו ונקבל שלוש משוואות בשלושה נעלמים.

אם אין לנו מידע מוקדם על הנגזרות נשמש בתוצאות הביניים כדי להסיק מידע על הנגזרות וכך נוכל לכפות על הפתרון שלנו להיות גזיר גם בנקודות הדגימה. כלומר, נפתור את המשוואות עבור הקטע הראשון. זה ייתן לנו פולינום f_1 ממעלה $N-1$ שמתלכד עם N נקודות הדגימה הראשונות. ואז כדי לחשב את f_2 נוסיף לאילוצים שלנו גם את המשוואה $p_1'(x_{N-1}) = p_2'(x_{N-1})$. כך נוכל להמשיך הלאה ולתפור פונקציה לכל הקטע.

אינטרפולציה על סריגים אחידים

נניח כעת שנתונות לנו נקודות על גרף הפונקציה במרווחים אחידים ויתר על כן, הנקודות שאותן נרצה לשערך תמיד יחתכו את הקטעים הקטנים באותו מיקום יחסי. כלומר אם נתונות לנו הנקודות $x_0, \dots, x_M$ אז לכל שתי נקודות y, z שנרצה לשערך מתקיים $\frac{x_0-z}{x_1-z} = \frac{x_i-z}{x_{i+1}-z} = \frac{x_i-y}{x_{i+1}-y}$ לכל $i = 0, \dots, M$.

אינטרפולציה פולינומיאלית מייצרת ערכים חדשים כפונקציה של הקלט $\{(x_i, y_i)\}_{i=0}^{M}$. כלומר, מתקבלים $c_0, \dots, c_M$ כך ש-$p(x) = \sum_{i=0}^{M-1} c_i x^i$. בכתיב מטריצות זה למעשה

$$p(x) = \left\langle \begin{pmatrix} 1 \\ \vdots \\ x^{M-1} \end{pmatrix}, \begin{pmatrix} c_0 \\ \vdots \\ c_{M-1} \end{pmatrix} \right\rangle = \langle \vec{x}, \vec{c} \rangle$$

אבל הרי $\vec{c}$ התקבל כפתרון מערכת משוואות $A\vec{c} = \vec{y}$. לכן $p(x) = \langle \vec{x}, A^{-1}\vec{y} \rangle$ פונקציה ליניארית ב-$\vec{y}$. כמו כן, מאחר שהנקודות החדשות נמצאות במרווחים אחידים בקטעים שלנו, הבעיה שלנו אינווריאנטית תחת הזזה. כלומר, אם נזיז את הקלט y_i ב-t נקבל את אותה התוצאה מוזזת ב-t.

ידוע שההעתקות הליניאריות היחידות אשר אינווריאנטיות תחת הזזה הן קונבולוציות.[1] ולכן ניתן לקבל את השערוך כקונבולוציה עם הקלט שלנו.

<u>דוגמה</u>: Up sampling

אינטרפולציה מסדר ראשון: נמצא את וקטור הקונבולוציה שמייצר דגימה מהירה פי שלושה של הקלט שלנו. נניח שהאינטרפולציה היא מסדר ראשון, כלומר מחברים כל זוג נקודות ע"י קו ישר. במקרה זה ברור שמתקיים

$$\begin{aligned} z_{1+3k} &= y_{k+1} \\ z_{2+3k} &= \frac{2}{3} y_{k+1} + \frac{1}{3} y_{k+2} \\ z_{3+3k} &= \frac{1}{3} y_{k+1} + \frac{2}{3} y_{k+2} \end{aligned}$$

כאשר y_i הנקודות המקוריות ו-z_i הנקודות שחדשות שאנחנו רוצים לשערך. את המשוואות האלה אנחנו רוצים להפוך לצורה של קונבולוציה. כלומר, אנחנו רוצים למצוא את הפילטר f שיוצר את הערכים האלה. קודם כל, אנחנו צריכים לרווח את הווקטור y כך שיהיה מקום להכניס את הערכים החדשים שאנחנו מחשבים. כלומר, את הפילטר אנחנו מפעילים על $(y_1, 0,0, y_2, 0,0, y_3, 0,0, \dots)$. התוצאה המבוקשת היא

$$\left(y_1, \frac{2}{3} y_1 + \frac{1}{3} y_2, \frac{1}{3} y_1 + \frac{2}{3} y_2, y_2, \dots\right)$$

ולכן הפילטר הוא $\left(\frac{1}{3}, \frac{2}{3}, 1, \frac{2}{3}, \frac{1}{3}\right)$.

אינטרפולציה ריבועית: נגדיל את מספר הדגימות פי שניים ע"י שימוש באינטרפולציה מסדר שני. כלומר, וקטור הקלט שלנו הופך ל-$(y_1, 0, y_2, 0, \dots)$. האינטרפולנט הוא פולינום מהצורה

[1] תזכורת: קונבולוציה של שני וקטורים u, v היא הווקטור המוגדר ע"י $(u * v)_i = \sum_{n=0}^{N-1} u_k v_{i-k}$

כלומר מערכת המשוואות שמתקבלת היא

אם נפתור את מערכת המשוואות נקבל

$$a = \frac{y_2 - 2y_1 + y_0}{2}$$
$$b = -\frac{y_2 - 4y_1 + 3y_0}{2}$$
$$c = y_0$$

כעת נוכל למצוא את $p\left(\frac{1}{2}\right)$ פשוט ע"י הצבה:

$$p\left(\frac{1}{2}\right) = \frac{y_2 - 2y_1 + y_0}{2} \cdot \frac{1}{4} - \frac{y_2 - 4y_1 + 3y_0}{2} \cdot \frac{1}{2} + y_0 =$$
$$= \frac{3}{8}y_0 + \frac{3}{4}y_1 - \frac{1}{8}y_2 = \frac{3}{8}y_0 + 1 \cdot 0 + \frac{3}{4}y_1 + 0 \cdot 0 - \frac{1}{8}y_2$$

מכאן ניתן להסיק שפילטר האינטרפולציה הוא $\left(-\frac{1}{8}, 0, \frac{3}{4}, 1, \frac{3}{8}\right)$. לא מפתיע שהתוצאה אינה סימטרית, שהרי מאחר שהשתמשנו בשלוש נקודות כדי לייצר את האינטרפולנט, הנקודה המשוערכת אינה נמצאת במרכז הקטע שמוגדר ע"י הנקודות ולכן הנקודות השונות תורמות תרומה שונה לשיערוך.

אגב, אחרי חישובים כאלה, בתור בדיקת שפיות אפשר לבדוק אם הפילטר שמצאנו עושה אינטרפולציה נכונה לפונקציה קבועה. כלומר, נניח שווקטור הקלט שלנו הוא $(a, 0, a, 0, a, 0, \dots)$. אם נחשב את הקונבולוציה אכן נקבל $(a, a, a, a, a, a, \dots)$. זה לא אומר שהפילטר שמצאנו הוא נכון, אבל אילו לא היינו מקבלים $(a, a, a, a, a, a, \dots)$, אז בוודאי הייתה לנו טעות איפשהו.

Least Squares Approximate Interpolation

יש לנו אוסף נקודות $\{(x_i, y_i)\}_{i=1}^{N}$ ואנחנו רוצים להתאים להן פולינום ממעלה שקטנה מ-N. בוודאי לא בהכרח קיים פולינום שמתאים באופן מושלם לכל הנקודות ולכן נרצה למצוא את הפולינום שמתאים בצורה הכי טובה שאפשר. מדד האיכות שנהוג להשתמש בו הוא **השגיאה הריבועית**. אם הפולינום המקרב p, השגיאה הריבועית מוגדרת ע"י $E = \sum_{i=1}^{N}(p(x_i) - y_i)^2$. המטרה היא למצוא פולינום שממזער את השגיאה הריבועית. ניתן לכתוב את הבעיה בצורה מטריציונית: נניח ש-M היא מעלת הפולינום שמחפשים והמקדמים שלו הם $c_0, \dots, c_M$. אזי מחפשים את המינימום לפי $c_0, \dots, c_M$ של

$$\|p(x_i) - y_i\| = \left(\begin{pmatrix} 1 & x_1 & \dots & x_1^M \\ \vdots & & \ddots & \vdots \\ 1 & x_N & \dots & x_N^M \end{pmatrix}\begin{pmatrix} c_0 \\ \vdots \\ c_M \end{pmatrix} - \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}\right)^t \left(\underbrace{\begin{pmatrix} 1 & x_1 & \dots & x_1^M \\ \vdots & & \ddots & \vdots \\ 1 & x_N & \dots & x_N^M \end{pmatrix}}_{A}\underbrace{\begin{pmatrix} c_0 \\ \vdots \\ c_M \end{pmatrix}}_{\vec{c}} - \underbrace{\begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}}_{\vec{y}}\right)$$

כלומר, מחפשים את

$$\min_{\vec{c}}(A\vec{c} - \vec{y})^t(A\vec{c} - \vec{y}) = \min_{\vec{c}} \vec{c}^t A^t A\vec{c} - 2\vec{y}^t A\vec{c} + \vec{y}^t\vec{y}$$

את זה אפשר לגזור לפי $\vec{c}$ ולקבל פתרון (אכן יש פיתרון כי הפונקציה היא סכום ריבועים ולכן בהכרח יש לה מינימום). הפיתרון שמתקבל נקרא **pseudo-inverse** והוא

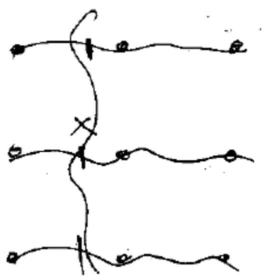
$$\vec{c} = (A^t A)^{-1} A^t \vec{y}$$

אינטרפולציה רב-ממדית

הפעם הקלט שלנו הוא ערכי פונקציה שמוגדרת על $\mathbb{R}^n$. כלומר, יש לנו $\{(\vec{x_i}, y_i)\}_{i=0}^{M}$ כאשר $\vec{x_i} \in \mathbb{R}^n$.

יש כמה אפשרויות שאפשר לנקוט בהן:

1. להרחיב את מה שעשינו במקרה החד-ממדי: לבחור פולינום ממעלה M ב-n משתנים ולנסות לפתור משוואות ולהתאים אותו. הבעיה העיקרית כאן (פרט לקושי החישובי) היא שיש מספר בחירות אפשריות של פולינום כזה ממעלה M. למשל, גם $x^2 + y^2$ וגם xy הם פולינומים ממעלה 2 ולא ברור אפריורי באיזה מהם עדיף לבחור.
2. לטפל בכל ממד בנפרד: אם הדגימות נתונות על סריג אחיד ניתן ראשית להתאים פולינומים לממד הראשון, אז להשתמש בתוצאה כדי לתפור את הממד השני וכך הלאה:



3. מתן השפעה גדולה יותר לנקודות שקרובות לנקודות שאנחנו רוצים לשערך...

Radial Basis Functions

Radial Basis Function היא פונקציה ממשית שמוגדרת על $\mathbb{R}^n$ ותלויה רק במרחק של הנקודה מראשית הצירים, או מנקודה אחרת. באופן כללי ניתן לומר שפונקציה כזאת היא מהצורה $\phi(x, c) = \phi(\|x - c\|)$.
בפונקציות כאלה אפשר להשתמש כדי לקרב את $\{(x_i, y_i)\}_{i=0}^{M}$ ע"י

$$f(x) = \sum_{i=0}^{M} w_i \phi(\|x - x_i\|)$$

אם $\phi(r) \xrightarrow[r\to\infty]{} 0$ אכן ככל שנקודה קרובה יותר לנקודת השיערוך ההשפעה שלה גדולה יותר. פונקציות אופייניות שמשתמשים בהן הם למשל $\phi(r) = r^k$ עבור $k = 2l - 1$ אי-זוגיים או $\phi(r) = e^{-\beta r^2}$ עבור $\beta > 0$ כלשהו.

כדי לקבל את המקדמים $\{w_i\}_{i=0}^{M}$ ניתן לפתור מערכת משוואות כמו קודם, כאשר הדרישה היא שלכל $i = 0, \dots, M$ מתקיים $f(x_i) = y_i$.

שיטת Shepard

אם $\phi(0) = 1$ והפונקציות דועכות כל כך מהר שכשהן מגיעות לשכנות הן מתאפסות, נקבע $w_i = y_i$. אז מתקבל הקירוב

$$f(x) = \sum_{i=0}^{M} y_i\, \phi(\|x - x_i\|)$$

אכן, מתקיים כאן $y_j \approx \left(\left\|x_j - x_i\right\|\right) f(x_j) = \sum_{i=0}^{M} y_i \phi$. לחילופין, ניתן לא להניח ש-$\phi(0) = 1$ ופשוט לנרמל במקום:

$$f(x) = \frac{\sum_{i=0}^{M} y_i\, \phi(\|x - x_i\|)}{\sum_{i=0}^{M} \phi(\|x - x_i\|)}$$

תכונה נוספת של קירוב זה היא שכל נקודת שיערוך חדשה היא צירוף קמור של $\{y_i\}_{i=0}^{M}$ ולכן

$$\min_i y_i \le f(x) \le \max_i y_i$$

אז הפונקציה לא בהכרח דועכת ל-0 באינסוף, אלא נשארת בתוך תחומי הגזרה. זוהי הווריאציה שמתכוונים אליה כשאומרים שיטת Shepard. זוהי נוסחה מפורשת של השערוך ולכן כדי לשערך נקודה יש צורך ב-$O(M)$ פעולות וזהו.

חלונות Parzen

נניח שיש אוסף תצפיות $\{x_i\}_{i=0}^{M} \subseteq \mathbb{R}^n$ ואנחנו מעוניינים לקרב את פונקצית הצפיפות שממנה נדגמו התצפיות, כלומר, נרצה לשערך את הסבירות לקבל דגימה בנקודה מסוימת $x \in \mathbb{R}^n$. נהוג לקרב את ההסתברות ע"י $\Phi(x) = \sum_{i=0}^{M} \phi(\|x - x_i\|)$. כדי לקבל פונקצית הסתברות ממש צריך גם לנרמל אבל אם רוצים רק להשוות בין נקודות שונות, זה מספיק.

אינטגרציה

הבעיה: נתונה פונקציה $f:\mathbb{R}\to\mathbb{R}$ ורוצים להעריך את $\int_a^b f(x)\,dx$. הבעיה היא כמובן שלא תמיד אנחנו יודעים ביטוי אנליטי לאינטגרל של $f(x)$.

שלבים בפתרון:

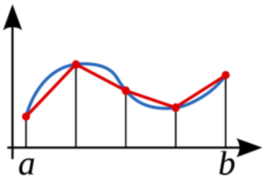
1. חלוקת תחום האינטגרציה למקטעים קטעים באורך h
2. קירוב הפונקציה f בכל קטע ע"י פונקציה פשוטה שאנחנו יודעים את האינטגרל שלה (למשל פולינום)
3. סיכום כל האינטגרלים במקטעים הקטנים

זה קצת מזכיר את הגדרת האינטגרל כסכומי רימן, אלא ששם מאחר שלקחנו $h\to 0$ קיבלנו בסוף תוצאה מדוייקת ואילו כאן אנחנו מקבלים רק קירוב. בדומה, נראה שככל שנחלק את תחום האינטגרציה לקטעים קטנים יותר, נקבל תשובה מדויקת יותר (אבל כמובן זה יעלה לנו באורך החישוב).

משפט: תהי $f:\mathbb{R}\to\mathbb{R}$ ונניח שנחשב את האינטגרל $\int_a^b f(x)\,dx$ ע"י קירוב הפונקציה במקטעים באורך h ע"י פולינומים מסדר N. איכות קירוב האינטגרל היא $O(h^{N+1})$.

שיטת הטרפז

נבצע קירוב מסדר ראשון בין כל זוג נקודות עוקבות:



הקירוב מסדר ראשון על קטע אחד נותן לנו בשימוש בנוסחה של שטח של טרפז

$$\int_{x_i}^{x_{i+1}} p(x)\,dx = h\left(\frac{y_{i+1}-y_i}{2}\right) = \frac{b-a}{n}\left(\frac{y_{i+1}-y_i}{2}\right)$$

כאשר n הוא מספר המקטעים. כשנסכם את התוצאות על כל הקטעים נקבל

$$\int_a^b f(x)\,dx \approx \frac{b-a}{n}\left(\frac{f(a)+f(b)}{2}+\sum_{i=1}^{n-1} f\left(a+k\frac{b-a}{n}\right)\right)$$

לפי המשפט למעלה איכות הקירוב של שיטת הטרפז היא $O(h^2)$.

שיטת סימפסון

נעשה אינטגרציה נומרית באמצעות קירובים ממעלה שנייה. הנוסחה שמתקבל היא

$$\int_a^b f(x)\,dx \approx \frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)$$

אם נגדיר $h=\frac{b-a}{n}$ נקבל מהמשפט הקודם שאיכות הקירוב הוא $O(h^3)$. אבל ניתוח קפדני יותר יכול להראות שלמעשה החסם טוב יותר.

משפט: איכות הקירוב של שיטת סימפסון היא $O(h^4)$.

למעשה, ככלל, אם נבצע את האינטרפולציה באמצעות פולינום ממעלה אי-זוגית M נקבל שאיכות הקירוב היא $O(h^{M+2})$. הסיבה לכך היא שמונום אי-זוגי הוא פונקציה אי-זוגית ולכן האינטגרל עליו מתאפס. לכן, כשמשתמשים בפולינום ממעלה אי-זוגית מקבלים דיוק נקודתי גבוה יותר אבל שגיאת האינטגרל לכל קטע אינה גדלה.

אינטגרציה רב-ממדית

כמו באינטרפולציה רב-ממדית, יש שתי אפשרויות שניתן לנקוט בהן:

1. להתאים פולינום רב-ממדי ולחשב את האינטגרל ממנו.
2. לחשב את האינטגרל בשלבים לכל ממד בנפרד. הצידוק המתמטי של השיטה הזו נובע ממשפט פוביני:
$$\int_{A\times B} f(x,y)\,d(x,y) = \int_A \left(\int_B f(x,y)\,dy \right) dx$$
גם במקרה זה מתקבלת שגיאה $O(h^{N+1})$ כאשר N סדר הקירוב.
3. שיטות סטטיסטיות

אינטגרציית מונטה-קרלו

אם אנחנו עובדים ב-$\mathbb{R}^d$ ואנחנו מעוניינים לדגום את תחום האינטגרציה בסריג אחיד עם מרחק h בין נקודות הדגימה בכל ציר יוצא שיש $N = O\left(h^{-d}\right)$ נקודות דגימה סה"כ והשגיאה שמתקבלת היא $E = O(h^{M+1})$. לכן $h = O\left(N^{-d}\right)$ ו-$E = O\left(N^{-\frac{M+1}{d}}\right)$ וכאן יש שני גדלים שמתחרים זה בזה כי ככל ש-M גדול יותר החסם קטן וככל שהממד d גדול יותר החסם דווקא גדל. בפועל בגלל אילוצי חישוב דווקא M חסום ע"י כ-8 ואילו d יכול להיות גדול מאוד! נרצה להתגבר על זה איכשהו. שיטת מונטה-קרלו נועדה בדיוק בשביל הבעיה הזאת.

נניח ש-g פונקציית התפלגות ו-$x_i \sim g(x)$ בלתי תלויים. אזי

$$I = \int_a^b f(x)\,dx = \int_a^b \frac{f(x)}{g(x)} g(x)\,dx \approx \frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)}{g(x_i)}$$

הקירוב הזה נובע מכך ש-$\int_a^b \frac{f(x)}{g(x)} g(x)\,dx = \mathbb{E}_g\left(\frac{f}{g}\right)$.

השגיאה שמתקבלת היא משתנה אקראי שהתוחלת שלו היא $E = \frac{1}{\sqrt{N}}\operatorname{std}\frac{f}{g}$. כש-$\frac{f}{g}$ לא משתנה הרבה הקבוע $\operatorname{std}\frac{f}{g}$ הוא נמוך ומתקבלת שגיאה נמוכה. נשים לב שכתלות ב-N, השיטה מתכנסת תמיד! אם בחרנו פונקציה g לא מוצלחת יכול להיות ש-$\operatorname{std}\frac{f}{g}$ יהיה גדול מאוד, אך הוא קבוע ואינו תלוי ב-N. ולכן קצב ההתכנסות של אינטגרציה מונטה-קרלו הוא תמיד $O\left(\frac{1}{\sqrt{N}}\right)$ ללא תלות בפונקציה f שרוצים לחשב את האינטגרל שלה וללא תלות בפונקציה g שנבחרה.

המקרה האופטימלי הוא $\frac{f}{g} \equiv const$ אבל אז בעצם נובע ש-$g(x) = \frac{f(x)}{\int f(x)\,dx}$. אילו היינו יודעים לחשב את זה אז לא הייתה לנו בעיה מלכתחילה. לכן, אנחנו רוצים למצוא פונקציה g שדומה ל-f כמה שיותר. g כזאת נקראת **trial distribution**. גם אחרי שנמצא g כזאת עדיין נשארת הבעיה לדגום ממנה משתנה אקראי אבל זה אפשרי בהנחה (הסבירה) שיש לנו גישה למשתנה אקראי יחיד.

ניתן להרחיב אתה שיטה הזו לפונקציות רב-ממדיות אבל אז צריך לדגום משתנה אקראי רב-ממדי. זה אפשרי אבל מסובך יותר.

Rejection Sampling

לפעמים הבעיה באינטגרציה היא לא שאנחנו לא יודעים לעשות אינטגרל לפונקציה אלא שתחום האינטגרציה הוא מסובך ואין פרמטריזציה נוחה לעבוד איתה. נניח למשל ש-$U \supseteq D$ כאשר U תחום פשוט שאנחנו יכולים לחשב את האינטגרל עליו. אז מתקיים

$$\int_D f(x)\,dx = \int_U 1_D(x) f(x)\,dx$$

כעת אפשר להשתמש בשיטת מונטה-קרלו:

$$\int_U f(x) 1_D(x)\,dx = \int_{\mathbb{R}^d} f(x) 1_D(x) \frac{1_U(x)}{|U|} |U|\,dx \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i) 1_D(x_i)}{\frac{1_U(x_i)}{|U|}} = \frac{|U|}{N} \sum_{i=1}^{N} f(x_i) 1_D(x_i)$$

כאשר $x_i \sim \frac{1_U(x)}{|U|}$. אז אפשר להשתמש בשיטת מונטה-קרלו אלא שלפני שמכניסים נקודה לסכום בודקים אם היא בתחום האינטגרציה שלנו או לא.

מערכות משוואות לא ליניאריות

<u>הבעיה</u>: נתונה פונקציה $f:\mathbb{R}^n\to\mathbb{R}$ ורוצים למצוא $x\in\mathbb{R}^n$ כך ש-$f(x)=0$.

שיטת החצייה

אם הפונקציה $f:\mathbb{R}\to\mathbb{R}$ רציפה ואנחנו יודעים שתי נקודות $a<b$ כך ש-$f(a)f(b)<0$ (כלומר סימן הפונקציה עליהן שונה) אז לפי משפט ערך הביניים קיימת נקודה $a<c<b$ כך ש-$f(c)=0$. אז אפשר פשוט לעשות חיפוש בינארי:

<u>BisectionMethod(f,a,b,ε)</u>

1. $a_0=a$
2. $b_0=b$
3. $n=0$
4. $c_n=\frac{a_n+b_n}{2}$
5. while $|f(c_n)|>\varepsilon$
   a. if $f(c_n)>0$
      i. $b_{n+1}=c_n$
      ii. $a_{n+1}=a_n$
   b. else if $f(c_n)<0$
      i. $b_{n+1}=b_n$
      ii. $a_{n+1}=c_n$
6. return c_n

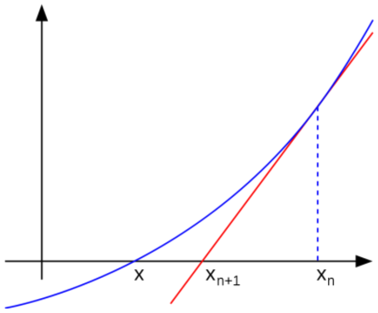
זאת שיטה איטרטיבית והיא משפרת את הקירוב בכל איטרציה. אם $f(r)=0$ אז $a_n<r<b_n$ לכל $n\in\mathbb{N}$ ולכן מאחר שבכל שלב אנחנו חוצים את קטע העניין נקבל ש-

$$e_n=|c_n-r|\le|b_n-a_n|=\frac{b-a}{2^n}\xrightarrow[n\to\infty]{}0$$

זה קצב התכנסות מעולה! אכן, זו שיטה טובה לפונקציות חד-ממדיות שעבורן יש לנו $a<b$ כנדרש. אלא שזה לא תמיד המצב ועוד יותר חמור מזה, לא ניתן להרחיב את השיטה לפונקציות רב-ממדיות.

שיטת Newton-Raphson

אם נניח ש-f גזירה ניתן לפתח שיטה שלא משתמשת בניחוש ראשוני. השיטה לא תמיד מתכנסת אבל כשהיא מתכנסת זה קורה מאוד מהר, ויתר על כן, היא מתרחבת למקרה הרב-ממדי בקלות. השיטה מתבססת על קירובים ליניאריים של הפונקציה.



בכל שלב מקרבים את הפונקציה ע"י המשיק שלה בנקודה. זוהי פונקציה ליניארית וניתן למצוא את השורש שלה. אז מצופה שהשורש של המשיק יהיה קרוב לשורש הפונקציה. משם ממשיכים הלאה:

<u>NewtonRaphson(f, f', x_0)</u>

1. $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$
2. $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

יש כמה נושאים חישוביים שצריך לתת עליהם את הדעת:

1. הפונקציה צריכה להיות גזירה בכל נקודה שבה האלגוריתם עובר. אחרת לא ברור כיצד להמשיך.
2. ברור מכלל הקידום של האלגוריתם שאם הנגזרת מתאפסת בנקודה כלשהי האלגוריתם עלול לקרוס ואז לא ברור כיצד להמשיך.
3. לפעמים יכול להיות קשה לחשב את הנגזרת וזה עלול להשפיע על זמן החישוב.
4. אם הערך ההתחלתי x_0 רחוק מהפיתרון השיטה עלולה לא להתכנס ולכן בד"כ מגבילים את מספר האיטרציות של האלגוריתם.
5. אם הנגזרת של הפונקציה לא רציפה השיטה עלולה לא להתכנס.

<u>**משפט**</u>: תהי $f: \mathbb{R} \to \mathbb{R}$ פונקציה גזירה ברציפות פעמיים ונניח ש-$f(r) = 0$. נסמן ב-$e_n = x_n - r$ את השגיאה המתקבלת באיטרציה n של שיטת ניוטון-רפסון. אזי לכל $n \in \mathbb{N}$ קיימת ξ_n כך ש-

$$e_{n+1} = \frac{e_n^2}{2} \frac{f''(\xi_n)}{f'(x_n)}$$

בפרט, אם $\frac{e_n}{2} \frac{f''(\xi_n)}{f'(x_n)} < 1$ לכל $n \in \mathbb{N}$ אז התהליך מתכנס. אם הנגזרות הראשונה והשנייה חסומות והניחוש הראשוני מספיק טוב אז התהליך מתכנס בקצב סופר-ריבועי, שזה יותר מהר משיטת החצייה.

היתרון המשמעותי של השיטה הזו הוא שאפשר להרחיב אותה באופן פשוט לפונקציות רב-ממדיות. אז כלל ההתקדמות פשוט נתון ע"י

$$x_{n+1} = x_n - \left(\nabla f(x_n)\right)^{-1} f(x_n)$$

כאשר

$$\nabla f = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & & \frac{\partial f_1}{\partial x_N} \\ & \ddots & \\ \frac{\partial f_N}{\partial x_1} & & \frac{\partial f_N}{\partial x_N} \end{pmatrix}$$

כמו שמקרה החד-ממדי היינו צריכים ש-$f'(x_n) \neq 0$, כאן כדי שהאלגוריתם יוכל להמשיך צריך להתקיים ש-$\nabla f(x_n)$ הפיכה. ניתן להראות שבתנאים מסוימים גם השיטה הזאת מתכנסת.

מערכות משוואות ליניאריות

<u>הבעיה</u>: נתונה מטריצה ריבועית הפיכה $A \in M_n(\mathbb{R})$ ונתון $b \in \mathbb{R}^n$ ורוצים למצוא $x \in \mathbb{R}^n$ כך ש-$Ax = b$.

יש שני סוגי פתרונות לבעיה:

- **שיטות ישירות**: מבצעות מספר סופי של פעולות חשבוניות שבסופן מתקבל פתרון מדויק של המשוואה (עד כדי שגיאות round off).
- **שיטות איטרטיביות**: מתקרבות לפיתרון בשלבים אבל בשום נקודת זמן סופית (פרט למקרים בודדים) אין פיתרון מדויק.

שיטות ישירות

אלימינציה של גאוס

השיטה הישירה הכי פשוטה היא אלימינציה של גאוס. מפעילים אוסף של פעולות אלמנטריות על המטריצה $[A, b]$ כאשר כל פעולה משמרת את מרחב הפתרונות עד ש-A הופכת למטריצת הזהות. בשלב זה, הווקטור שהתקבל מ-b הוא הפתרון. הפעולות האלמנטריות הן:

- החלפת סדר שתי שורות
- כפל שורה בסקלר ששונה מאפס
- הוספת שורה לשורה אחרת

בעזרת פעולות אלה ניתן לאפס את כל האיברים שמחוץ לאלכסון הראשי במטריצה. השיטה עובדת ב-$O(n^3)$ כי צריך לאפס $O(n^2)$ איברים ולכל אחד מהם זה כרוך ב-$O(n)$ פעולות.

פירוק LU

אם נצליח לפרק את A למכפלה $A = LU$ כאשר L משולשית תחתונה ו-U משולשית עליונה מערכת המשוואות $Ax = b$ תהפוך ל-$L(Ux) = b$. אבל L משולשית תחתונה ולכן ע"י הצבה לאחור (back substitution) נוכל לפתור את המשוואה $Ly = b$ בקלות. ואז ע"י הצבה לאחור נוכל לפתור בקלות את המשוואה $Ux = y$ וזה הרי בדיוק מה שרצינו.

$$\begin{pmatrix} L_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ L_{n1} & \cdots & L_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$

ולכן,

$$L_{11} y_1 = b_1 \Rightarrow y_1 = \frac{b_1}{L_{11}}$$
$$L_{i1} y_1 + \cdots + L_{ii} y_i = b_i \Rightarrow y_i = \frac{b_i - \left(L_{i1} y_1 + \cdots + L_{i,i-1} y_{i-1}\right)}{L_{ii}}$$

ובאופן דומה, אם

$$\begin{pmatrix} U_{11} & \cdots & U_{n1} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & U_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

אז נקבל ע"י הצבה לאחור

$$U_{nn} x_n = y_n \Rightarrow x_n = \frac{y_n}{U_{nn}}$$

$$U_{ii}x_i + \cdots + U_{in}x_n = y_n \Rightarrow x_i = \frac{y_n - (U_{i,i+1}x_{i+1} + \cdots + U_{in}x_n)}{U_{ii}}$$

בהינתן L,U החישוב הזה לוקח $O(n^2)$.

כיצד נחשב את הפירוק? נניח שקיים פירוק

$$\begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{n1} & \cdots & A_{nn} \end{pmatrix} = \begin{pmatrix} L_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ L_{n1} & \cdots & L_{nn} \end{pmatrix} \begin{pmatrix} U_{11} & \cdots & U_{n1} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & U_{nn} \end{pmatrix}$$

אז $A_{11} = L_{11}U_{11}$ ולכן ניתן לבחור L_{11}, U_{11} כלשהם כך ש-$A_{11} = L_{11}U_{11}$. אבל $L_{21}U_{11} = A_{21}$ והרי כבר יש לנו את U_{11} ולכן ניתן למצוא את L_{21}. כך ניתן להמשיך ולמצוא את כל העמודה הראשונה של L ובאופן דומה את כל השורה הראשונה של U. עכשיו ניתן להמשיך באינדוקציה.

<u>LU-Decomposition(A)</u>

1. for $k = 1, \dots, n$
   a. $L_{kk}U_{kk} = A_{kk} - \sum_{i<k} L_{ki}U_{ik}$
   b. for $j = k+1, \dots, n$
      i. $L_{jk} = \frac{A_{jk} - \sum_{i<k} L_{ji}U_{ik}}{U_{kk}}$
   c. for $j = k+1, \dots, n$
      i. $U_{kj} = \frac{A_{kj} - \sum_{i<k} L_{ki}U_{ij}}{L_{kk}}$

הפירוק הזה לוקח $O(n^3)$ אז לכאורה אין לשיטה הזאת יתרון על פני אלינימציה של גאוס אבל אם סופרים את הפעולות בדקדקנות ניתן לגלות שיש הבדל בקבועים והשיטה הזאת משתמשת בכשני שליש מהפעולות של שיטת גאוס.

<u>הגדרה</u>: תת מטריצה עיקרית של מטריצה נתונה A היא תת מטריצה שהאלכסון שלה הוא חלק מהאלכסון הראשי של A.

<u>משפט</u>: תהי A מטריצה כך שכל תת מטריצה עיקרית שלה מהצורה $\begin{pmatrix} A_{11} & \cdots & A_{1k} \\ \vdots & \ddots & \vdots \\ A_{nk} & \cdots & A_{kk} \end{pmatrix}$ עבור $k \le n$

היא הפיכה. אזי קיים ל-A פירוק LU.

<u>משפט</u>: תהי A מטריצה הפיכה. אזי ל-A קיים פירוק LU אמ"מ כל תת מטריצה עיקרית שלה היא הפיכה.

ההבדל בין שני המשפטים הוא שבראשון לא דורשים שהמטריצה הפיכה. אכן, למטריצת האפס קיים פירוק LU על אף שאין לה אף תת מטריצה עיקרית הפיכה.

פירוק Cholesky

<u>משפט</u>: תהי $A \in M_n(\mathbb{R})$ מטריצה סימטרית. אזי A מוגדרת חיובית אמ"מ קיימת מטריצה משולשית תחתונה $L \in M_n(\mathbb{R})$ כך ש-$A = LL^t$ וערכי האלכסון של L חיוביים ממש.

פירוק זה נקרא **פירוק Cholseky** והוא יחיד. למעשה גם למטריצה סימטרית מוגדרת אי-שלילית קיים פירוק כזה אלא שאז ערכי האלכסון של L יכולים להיות גם אפס והפירוק לא יחיד בהכרח.

כמובן, מדובר כאן במקרה פרטי של פירוק LU ולכן בדיוק כמו שתואר קודם, ניתן לפתור מערכת משוואות כאשר המטריצה סימטריות ומוגדרת אי-שלילית באמצעות פירוק Cholesky.

CholeskyDecomposition(A)

1. for $k = 1, \dots, n$
   a. $L_{kk} = \sqrt{A_{kk} - \sum_{i=1}^{k-1} L_{ki}^2}$
   b. for $i = k+1, \dots, n$
      i. $L_{ik} = \frac{A_{ik} - \sum_{j=1}^{k-1} L_{ij} L_{kj}}{L_{kk}}$

גם הפירוק הזה לוקח $O(n^3)$.

שיטות איטרטיביות

שיטה איטרטיבית היא שיטה שבה בכל איטרציה משפרים את הפתרון שיש לנו ויש כל מיני מדדים כדי לקבוע מתי לעצור. מעתה, נסמן את ממד המטריצה ב-N ואת מספר האיטרציה ב-n.

מטריצות פירוק

נניח שמערכת המשוואות שלנו היא $Ax = b$ ונניח ש-Q מטריצה הפיכה כלשהי. אזי מתקיים

$$\begin{aligned} & Ax = b \\ \Leftrightarrow\ & Qx - Qx + Ax = b \\ \Leftrightarrow\ & Qx = (Q - A)x + b \\ \Leftrightarrow\ & x = Q^{-1}(Q - A)x + Q^{-1}b \end{aligned}$$

זה כמובן מייצג בדיוק את אותה מערכת המשוואות אבל את הביטוי הזה אפשר להפוך לשיטה איטרטיבית. נתחיל מניחוש ראשוני כלשהו x^0 ונתקדם לפי הכלל

$$x^{n+1} = Q^{-1}(Q - A)x^n + Q^{-1}b$$

כמובן, צריך ש-Q תהיה קלה להיפוך (אחרת לא באמת קיצרנו את התהליך) והכי חשוב – נרצה שהתהליך יתכנס!

במקרה הכללי כל איטרציה כרוכה במכפלת מטריצות בווקטורים ולכן לוקחת $O(N^2)$. אבל אם המטריצות דלילות (כלומר, בכל שורה יש $O(1)$ איברים שאינם אפס) אז המכפלה לוקחת $O(N)$ ואז אם מספר האיטרציות עד ההתכנסות קטן מ-N אנחנו במצב טוב ושיפרנו את הביצועים לעומת השיטות הישירות.

מטריצה Q כזאת נקראת **מטריצת פירוק**. כדי שנוכל לדבר על איכות וקצב התכנסות נכניס את מושג **הנורמה האופרטורית**

עבור מטריצה $A \in M_N(\mathbb{R})$ נגדיר את הנורמה האופרטורית שלה ע"י

$$\|A\|_k = \max_{x \neq 0} \frac{\|Ax\|_k}{\|x\|_k}$$

כאשר $\|x\|_k = \sqrt[k]{\sum_{i=1}^{N} x_i^k}$. ברור ש-$\|A\|_k = 0$ אמ"מ $A = 0$. באופן כללי, ניתן להוכיח שזו אכן הגדרה של פונקצית נורמה.

כמה טענות שכדאי לדעת לגבי הנורמה האופרטורית:

- עבור מטריצות סימטריות מתקיים $\|A\|_2 = \rho(A)$ – הרדיוס הספקטרלי של A. כלומר, $\|A\|_2$ שווה לערך העצמי בעל הערך המוחלט הגבוה ביותר של A.

- עבור מטריצות נורמליות (כלומר $A^tA = AA^t$) מתקיים $\|A\|_2 = \sqrt{\rho(A^tA)}$
- לכל $x \in \mathbb{R}^N$ ולכל $n \in \mathbb{N}$ מתקיים $\|A\|_k^n \|x\|_k \geq \|A^n x\|_k$

נבחן כעת את האבולוציה של השגיאה. נניח ש-$Ax = b$. אזי

$$\begin{aligned} e^{n+1} &= x^{n+1} - x = \\ &= Q^{-1}(Q - A)x^n + Q^{-1}b - x = \\ &= x^n - Q^{-1}Ax^n + A^{-1}b - x = \\ &= x^n - x - Q^{-1}(Ax^n - b) = \\ &= e^n - Q^{-1}(A(e^n + x) - b) = \\ &= e^n - Q^{-1}(Ae^n + Ax - b) = \\ &= (I - Q^{-1}A)e^n \end{aligned}$$

ומכאן ש-

$$\begin{aligned} \|e^{n+1}\|_k = \|(I - Q^{-1}A)e^n\|_k &\leq \|I - Q^{-1}A\|_k \|e^n\|_k \\ &\leq \|I - Q^{-1}A\|_k \|I - Q^{-1}A\|_k \|e^{n-1}\|_k \leq \\ &\vdots \\ &\leq \|I - Q^{-1}A\|_k^n \|e^0\|_k \end{aligned}$$

ולכן כדי שהתהליך יתכנס מספיק שיתקיים $\|I - Q^{-1}A\|_k < 1$. יתר על כן, מספיק שזה יהיה נכון לנורמת k כלשהי משום שידוע שכל הנורמות שקולות.

נדון כעת בשתי דרכים לבחור את מטריצת הפירוק.

שיטת Jacobi

בשיטת Jacobi בוחרים את $Q = \operatorname{diag} A$. כלומר, מתקבל האלגוריתם הבא:

<u>Jacobi(A, b, x^0)</u>

1. for $n = 1, \dots, K$
   a. for $i = 1, \dots, N$
      i. $x_i^{n+1} = -\sum_{j \neq i} \frac{A_{ij}}{A_{ii}} x_j^n + \frac{b_i}{A_{ii}}$

באלגוריתם K הוא מספר האיטרציות שנקבע מראש. אידיאלית, מוטב היה להפסיק כאשר התהליך מתכנס, אבל קשה לבחור מדד טוב להתכנסות. למשל, ניתן להמשיך באיטרציות כל עוד $\|Ax^n - b\| \geq \varepsilon$, אבל אם מדובר בווקטורים קטנים הנורמה הזאת יכולה להיות קטנה גם כאשר התהליך עוד לא התכנס. ובכל זאת, מאחר שאין משהו טוב יותר, זה המדד שנוהג להשתמש בו.

כל איטרציה באלגוריתם לוקחת $O(N^2)$ עבור מטריצות מלאות, אך אם המטריצה דלילה ניתן לרדת ל-$O(N)$. ואם מספר האיטרציות קטן ולא תלוי ב-N אז אנחנו במצב טוב.

<u>**הגדרה**</u>: נאמר שמטריצה A היא **דומיננטית אלכסונית** אם לכל $1 \leq i \leq N$ מתקיים $|A_{ii}| \geq \sum_{j \neq i} |A_{ij}|$.

<u>**משפט**</u>: שיטת Jacobi מתכנסת עבור מטריצות דומיננטיות אלכסונית ולכל ניחוש ראשוני בקצב $\|I - Q^{-1}A\|_\infty$.

שיטת Gauss-Seidel

בשיטת Gauss-Seidel בוחרים $Q = \text{lower}\, A$ – החלק המשולשי התחתון של המטריצה. אז מתקבל האלגוריתם הבא:

Gauss-Seidel(A, b, x^0)

1. until convergence do
   a. for $i = N, \dots, 1$
      i. $x_i^{n+1} = \frac{b_i - \sum_{j=i+1}^{N} A_{ij} x_j^n - \sum_{j=1}^{i-1} A_{ij} x_j^{n+1}}{A_{ii}}$

גם כאן הסיבוכיות של על איטרציה היא $O(N^2)$ למטריצות מלאות ו-$O(N)$ למטריצות דלילות.

משפט: שיטת Gauss-Seidel מתכנסת עבור מטריצות דומיננטיות אלכסונית ולכל ניחוש ראשוני בקצב $\|I - Q^{-1}A\|_\infty$.

היתרון העיקרי של השיטה הזו הוא שהיא ניתנת למימוש in-place וללא הקצאת זיכרון נוסף. בשיטת Jacobi צריך תמיד להחזיק את x^n ואת x^{n+1}. יתר על כן, שיטת Gauss-Seidel מתכנסת מעט מהר יותר. מאידך, יש מקרים שבהם Jacobi מתכנסת ואילו Gauss-Seidel אינה מתכנסת. בכל מקרה, עבור מטריצות דומיננטיות אלכסונית, שתי השיטות מתכנסות לכל וקטור b ולכל ניחוש ראשוני x^0.

Successive Over Relaxation

ראינו שכדי לפתור את מערכת המשוואות $Ax = b$ ניתן לבצע תהליך איטרטיבי המוגדר ע"י נוסחת הנסיגה $x^{n+1} = Q^{-1}\left((Q - A)x^n + b\right)$. כמו כן ראינו שכדי שהתהליך יתכנס צריך שתהיה נורמה שעבורה מתקיים

$$\|I - Q^{-1}A\|_k < 1$$

כמובן, ניתן לכפול את Q בכל קבוע חיובי ולכן אפשר גם להסתכל על התנאי

$$\|I - \alpha Q^{-1}A\| < 1$$

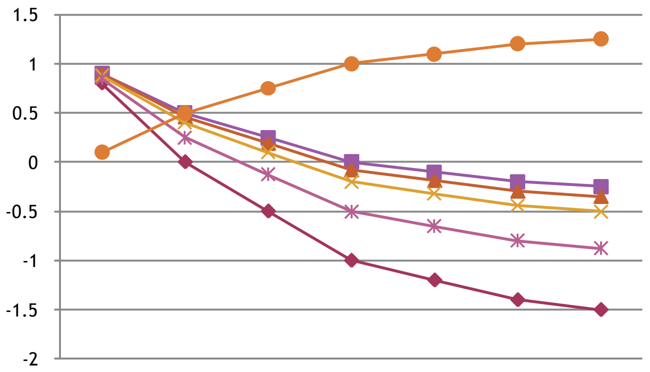
נניח כעת ש-A, Q מוגדרות חיובית ונראה מה התנאי להתכנסות. נתבונן למשל בנורמה אוקלידית. אז

$$\|I - \alpha Q^{-1}A\|_2 \le \rho(I - \alpha Q^{-1}A) = 1 - \alpha\lambda_{\min}$$

כאשר $\lambda_{\min}$ ערך עצמי מינימלי של A. נרצה למצוא את ה-α האופטימלי, כלומר אנחנו מעוניינים ב-

$$\text{argmin}_\alpha \max_i |1 - \alpha\lambda_i|$$

שהרי בגלל ש-A, Q מוגדרות חיובית נובע ש-$\rho(I - \alpha Q^{-1}A) = \max_i |1 - \lambda_i|$. נתבונן בגרף של הערכים העצמיים של $Q^{-1}A$, כלומר נתבונן בהם ממוינים מגדול לקטן:



הגרף העולה מייצג את הערכים העצמיים של $Q^{-1}A$ והגרפים היורדים מייצגים את הערכים העצמיים של $I-\alpha Q^{-1}A$ עבור ערכים שונים של α. אנחנו רוצים לבחור α כזה שהגרף המתאים לו לא ייצא מגבולות $[-1,1]$ וכן $1-\alpha\lambda_{\min} = -1+\alpha\lambda_{\max}$ כי אז הערך המוחלט יהיה הכי קטן שאפשר, כלומר הגרף רחוק במידה שווה מ-1 ומ--1. α כזה מתקבל כאשר $\alpha = \frac{2}{\lambda_{\max}+\lambda_{\min}}$. קצב ההתכנסות שמתקבל במקרה זה הוא

$$1-\frac{2\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}} = \frac{\frac{\lambda_{\max}}{\lambda_{\min}}-1}{\frac{\lambda_{\max}}{\lambda_{\min}}+1} = \frac{\kappa(A)-1}{\kappa(A)+1}$$

כאשר $\kappa(A) = \frac{\lambda_{\max}}{\lambda_{\min}}$ ערך הנקרא **מספר המצב** של המטריצה A.

בפועל בוחרים בד"כ $\alpha = \frac{2}{\lambda_{\max}}$ גם כי זה אומר שצריך לחשב פחות וגם כי הרבה פעמים $\lambda_{\min}$ קרוב מאוד לאפס ולכן זניח.

Steepest Descent

נניח כעת שמערכת המשוואות נתונה ע"י $Ax=b$ כאשר A סימטרית ומוגדרת חיובית. במקרה כזה ניתן להגדיר מכפלה פנימית על $\mathbb{R}^N$ ע"י $\langle x,y\rangle_A = x^tAy$ ואז הנורמה המושרית היא $\|x\|_A^2 = x^tAx$.

הפונקציה $f(x) = \frac{x^tAx}{2} - x^tb$ היא תבנית ריבועית ויש לה מינימום כאשר $Ax=b$: מאחר ש-A סימטרית ומוגדרת חיובית נובע שגם A^{-1} היא סימטרית ומוגדרת חיובית ולכן,

$$\begin{aligned}\tilde{f}(x) &= \|Ax-b\|_{A^{-1}}^2 = \\ &= (Ax-b)^tA^{-1}(Ax-b) = \\ &= (x^tA^t-b^t)A^{-1}(Ax-b) = \\ &= (x^tA^t-b^t)(x-A^{-1}b) = \\ &= (x^tA-b^t)(x-A^{-1}b) = \\ &= x^tAx - x^tb - b^tx + b^tA^{-1}b = \\ &= x^tAx - 2x^tb + b^tA^{-1}b\end{aligned}$$

אבל $b^tA^{-1}b$ מספר קבוע ולכן $\operatorname{argmin} f(x) = \operatorname{argmin}\tilde{f}(x)$ והוא x כך ש-$Ax=b$.

לכן, כדי לפתור את מערכת המשוואות אנחנו נרצה למזער את הפונקציה $f(x)$. את זה נעשה בשיטה איטרטיבית. הרעיון הוא שבכל שלב נבחר כיוון התקדמות כלשהו d^n ונמזער את הפונקציה לאורך כיוון זה. בשיטת SD בוחרים בכל שלב את הכיוון להיות

$$d^n = r^n = b - Ax^n$$

ע"י הצבה בפונקציה וגזירה ניתן לקבל שהמינימום מתקבל כאשר מתקדמים $\alpha = \frac{(r^n)^tr^n}{(r^n)^tAr^n}$ בכיוון d^n. מתקבל אז האלגוריתם הבא:

<u>SteepestDescent(A,b,x^0)</u>

1. until convergence do
   a. $r^n = b - Ax^n$
   b. $\alpha = \frac{(r^n)^tr^n}{(r^n)^tAr^n}$

c. $x^{n+1} = x^n + \alpha r^n$

תכונות של האלגוריתם:

1. כל איטרציה כרוכה בביצוע מכפלת מטריצות בווקטורים ולכן לוקחת $O(N)$ אם המטריצות דלילות ו-$O(N^2)$ במקרה הכללי. אם מספר האיטרציות בלתי תלוי ב-N אז זו גם הסיבוכיות של כל האלגוריתם.
2. כיווני החיפוש r^n מקיימים שתי תכונות:
$$r^n \perp r^{n+1}$$
$$r^n \perp_A e^{n+1}$$
3. קצב ההתכנסות הוא $\omega_{SD} = \frac{\kappa(A)-1}{\kappa(A)+1}$ כאשר $1 \geq \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} = \kappa(A)$ מספר המצב של A. כלומר, $\|e^{n+1}\|_A \leq \|e^n\|_A \omega_{SD}$. כאשר מספר המצב נמוך קצב ההתכנסות גבוה. בהמשך נראה שיטות שבאמצעותן נגרום להורדת מספר המצב של מטריצה.

Conjugate Gradients

בשיטת ה-Steepest Descent הרבה פעמים יוצא שבהרבה איטרציות התהליך מתקדם באותו כיוון. היה טוב אילו בכל שלב היינו מתקדמים בכיוון אחר, כלומר, בכל שלב היינו מתקדמים בדיוק במידה הנכונה באותו כיוון. למשל אילו ידענו למצוא כיוונים אורתוגונליים ולדעת בדיוק כמה צריך להתקדם בהן אז אחרי N צעדים היינו מסיימים ומקבלים תוצאה מדויקת.

אכן נפעל בשיטה זו אלא שלא נשתמש באורתוגונאליות רגילה. נאמר ששני וקטורים x, y הם A-אורתוגונליים אם $\langle x, y\rangle_A = x^t A y = 0$.

נסמן

$$e^n = x^n - x$$
$$x^{n+1} = x^n + \alpha d^n$$
$$r^n = b - Ax^n$$

אפשר לגזור ולראות ש $\operatorname{argmin}_\alpha \|e^{n+1}\|_A = \operatorname{argmin}_\alpha \|e^n + \alpha d^n\|_A = \frac{(d^n)^t r^n}{(d^n)^t A d^n}$.

יתר על כן, אם $\{d^i\}_{i=0}^N$ קבוצה A-אורתוגונלית אז לכל $n > k$ מתקיים $e^n \perp_A d^k$:

$$e^{k+1} = x^{k+1} - x = x^k + \alpha d^k - x = e^k + \alpha d^k$$

ולכן

$$\langle e^{k+1}, d^k\rangle_A = \langle e^k + \alpha d^k, d^k\rangle_A = \langle e^k, d^k\rangle_A + \alpha\langle d^k, d^k\rangle_A =$$
$$= \left(e^k\right)^t A d^k + \frac{\left(d^k\right)^t r^k}{(d^k)^t A d^k}\left(d^k\right)^t A d^k =$$
$$= \left(e^k\right)^t A d^k + \left(d^k\right)^t r^k =$$
$$\underset{r^k=-Ae^k}{=} \left(e^k\right)^t A d^k - \left(d^k\right)^t A e^k \underset{A=A^t}{=} 0$$

וכעת, באינדוקציה ובשימוש בכך ש-$\{d^i\}_{i=0}^N$ קבוצה A-אורתוגונלית נקבל:

$$\langle e^{k+2}, d^k\rangle_A = \langle e^{k+1} + \alpha d^{k+1}, d^k\rangle_A = \langle e^{k+1}, d^k\rangle_A + \alpha\langle d^{k+1}, d^k\rangle_A = 0 + \alpha \cdot 0 = 0$$

אז אם $\{d^i\}_{i=0}^N$ קבוצה A-אורתוגונלית קיבלנו שע"י בחירת $\alpha = \frac{(d^n)^t r^n}{(d^n)^t A d^n}$ נקבל שהשגיאה בסוף כל שלב n היא A-אורתוגונלית ל-$\{d_i\}_{i=0}^{n-1}$. מאחר שיש לכל היותר $N+1$ כיוונים אורתוגונאליים אחרי $N+1$ שלבים השגיאה בהכרח תתאפס וסיימנו.

נשארת אם כן השאלה כיצד למצוא כיוונים A-אורתוגונליים כאלה. זה נעשה באמצעות תהליך גראם-שמידט על הווקטורים $\{r_i\}_{i=0}^n$. חישובים מייגעים מראים שלא באמת צריך לעשות את כל החישוב שצורך $O(n^3)$ בכל שלב וניתן להסתפק בפחות חישובים (כי כל מיני איברים מתאפסים).

סה"כ מתקבל האלגוריתם הבא:

<u>ConjugateGradients(A, b, x^0)</u>

1. $d^0 = r^0 = b - Ax^0$
2. until convergence do
   a. $\alpha = \frac{(d^n)^t r^n}{(d^n)^t A d^n}$
   b. $x^{n+1} = x^n + \alpha d^n$
   c. $r^{n+1} = r^n - \alpha A d^n$
   d. $\beta = \frac{(r^{n+1})^t r^{n+1}}{(r^n)^t r^n}$
   e. $d^{n+1} = r^{n+1} + \beta d^n$

באופן הזה כל איטרציה כרוכה בכמה פעולות כפל מטריצה בווקטור. אם המטריצות דלילות זה לוקח $O(N)$ ואם הן מלאות זה לוקח $O(N^2)$. אם מספר האיטרציות שאנחנו בוחרים לעשות קבוע ולא תלוי בגודל המטריצה זו הסיבוכיות של כל האלגוריתם וחסכנו הרבה!

הוכחת סדר ההתכנסות של האלגוריתם היא מסובכת אבל אפשר להוכיח ש-

$$\|e^{n+1}\|_A \le 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^n \|e^0\|_A$$

אז גם במקרה הזה נעדיף שמספר המצב של המטריצה יהיה כמה שיותר נמוך.

Preconditioning

יש בעיות שבהן מספר המצב $\kappa(A)$ מאוד גבוה ואולי אפילו תלוי בממד המטריצה. במקרה כזה השיטות שתיארנו אמנם מתכנסות אך לאט מאוד. היינו רוצים למצוא מערכת משוואות שקולה עם מספר מצב נמוך יותר.

נשים לב שאם $Ax = b$ ו-Q הפיכה אז $Q^{-1}Ax = Q^{-1}b$. ולשתי המשוואות יש בדיוק אותם פתרונות. אז נרצה למצוא Q כזאת שהופכת את מערכת המשוואות לפשוטה ובעלת מספר מצב נמוך. אלא שכעת $Q^{-1}A$ לא בהכרח סימטרית או מוגדרת חיובית (אפילו אם Q היא כזאת) ולכן אפילו אם הורדנו את מספר המצב זה לא עוזר כי לא ניתן להשתמש ב-CG או ב-SD.

לחילופין, ניתן להתבונן במערכת המשוואות $E^t A E \hat{x} = E^t b$. במקרה זה $E\hat{x}$ הוא פתרון של מערכת המשוואות המקורית. אלא שהפעם $E^t A E$ סימטרית ומוגדרת חיובית והתקווה שהיא שמספר המצב שלה נמוך יותר.

שתי הפעולות שתיארנו נקראות **preconditioning** והמטריצות נקראת **מטריצות בידוד**. נשים לב שאם נגדיר $Q^{-1} = EE^t$ אז אם $Q^{-1}Av = \lambda\, v$ אז $E^tAEE^{-1}v = \lambda E^{-1}v$ ולכן אין על פניו יתרון לשימוש במטריצות בידוד מסוג מסוים.

אם A מטריצה סימטרית ומוגדרת חיובית יש לה מטריצה מלכסנת אורתוגונאלית V כך ש-$D = V^tAV$ אלכסונית עם אלכסון חיובי ממש. אם נבחר $E = V\sqrt{D^{-1}}$ זו תהיה מטריצת בידוד מצוינת כי אז

$$E^tAE = \left(V\sqrt{D^{-1}}\right)^t AV\sqrt{D^{-1}} = \sqrt{D^{-t}}V^tAV\sqrt{D^{-1}} = \sqrt{D^{-1}}D\sqrt{D^{-1}} = I$$

אבל בפועל כמובן כמעט בלתי אפשרי למצוא את המטריצה המלכסנת הזו...

בגדול מציאת מטריצת בידוד מתבצעת בשני שלבים:

1. **בידוד** – מציאת הצירים V (בקירוב) ש-A מכווצת או מותחת ובחירת $D = \mathrm{diag}(V^tAV)$
2. **תיקון** – תיקון של המטריצה בצירים אלה, כלומר מתיחת הצירים שכווצו ולהפך $\left(V\sqrt{D^{-1}}\right)^t AV\sqrt{D^{-1}}$

אם V^tAV הייתה אלכסונית אז המצב היה אידיאלי. אם החישוב הוא מקורב חשוב שהקירוב יהיה טוב כי אם נבחר את מערכת הצירים לא טוב יכול להיות שלמעשה לא נשפר בכלל את המצב.

נגדיר את האנרגיה של וקטור x להיות $E(x) = x^tAx = \|x\|_A^2$. נשים לב שאם v וקטור עצמי מנורמל (לפי נורמה סטנדרטית) של A אז $E(v) = v^tAv = v^t\lambda v = \lambda v^tv = \lambda$.לכן, אם אנחנו בוחרים את כל הוקטורים להיות עם אנרגיה דומה זוהי בחירה גרועה של מערכת צירים. אנחנו מחפשים מערכת צירים חדשה שיש לה אנרגיות מעורבות – גם גבוהות וגם נמוכות ואז בשלב התיקון באמת יש משמעות למתיחה ולכיווץ. אם אנחנו מכירים איזו תכונה שמשפיעה על האנרגיה אפשר לנסות להשתמש בה כמדריך לבחירת כיוונים.

נשים לב שאם בוחרים $V = I$ אז $D = \mathrm{diag}\,A$ וקיבלנו למעשה את שיטת Jacobi. כלומר, שיטת Jacobi היא בעצם פשוט משתמשת במטריצת בידוד מסוימת כדי לפתור את הבעיה.

Transformed Preconditioned Conjugate Gradients

ראינו שכדי לפתור את $Ax = b$ מספיק לפתור את $E^tAE\hat{x} = E^tb$. מאחר ש-E^tAE היא סימטרית ומוגדרת חיובית ניתן לפתור את המשוואה באמצעות CG. אם נציב את הנתונים האלה באלגוריתם נקבל את האלגוריתם שנקרא Transformed Preconditioned CG:

<u>TransformedPCG$(A, b, \hat{x}^0, E)$</u>

1. $\hat{d}^0 = \hat{r}^0 = E^tb - E^tAE\hat{x}^0$
2. until convergence do
   a. $\alpha = \frac{(\hat{d}^n)^t\hat{r}^n}{(\hat{d}^n)^tE^tAE\hat{d}^n}$
   b. $\hat{x}^{n+1} = \hat{x}^n + \alpha\hat{d}^n$
   c. $\hat{r}^{n+1} = \hat{r}^n - \alpha E^tAE\hat{d}^n$
   d. $\beta = \frac{(\hat{r}^{n+1})^t\hat{r}^{n+1}}{(\hat{r}^n)^t\hat{r}^n}$
   e. $\hat{d}^{n+1} = \hat{r}^n + \beta\hat{d}^n$

Untransformed Preconditioned Conjugate Gradients

בגרסה הקודמת של האלגוריתם יש לחשב את E מתוך Q וזה יכול לקחת הרבה זמן. ובסוף גם צריך לחלץ את הפתרון למשוואה המקורית. ע"י כמה חילופי משתנים

$$\hat{r} = E^t r, \hat{x} = E^{-1}x, \hat{d} = E^{-1}d$$

ניתן לקבל את הגרסה Untransformed Preconditioned CG שמחזירה אותנו למשתנים המקוריים ונותנת לנו מיד את הפתרון x של המשוואה המקורית (כזכור, $x = E\hat{x}$):

<u>UntransformedPCG()</u>

1. $r^0 = b - Ax^0$
2. $d^0 = EE^t r^0 = Q^{-1} r^0$
3. until convergence do
    a. $\alpha = \frac{(d^n)^t Q^{-1} r^n}{(d^n)^t A d^n}$
    b. $x^{n+1} = x^n + \alpha d^n$
    c. $r^{n+1} = r^n - \alpha A d^n$
    d. $\beta = \frac{\left(\hat{r}^{n+1}\right)^t Q^{-1} \hat{r}^{n+1}}{(\hat{r}^n)^t Q^{-1} \hat{r}^n}$
    e. $d^{n+1} = Q^{-1} r^n + \beta d^n$

זה משפר את המצב משום שכעת לא צריך לפרק את Q^{-1} ל-EE^t ומספיק רק לדעת שקיים פירוק כזה.
לאחר מכן ניתן פשוט להריץ את האלגוריתם הזה ומיד לקבל את הפתרון של המשוואה המקורית!

ערכים ווקטורים עצמיים

<u>הבעיה</u>: נתונה מטריצה ריבועית $A \in M_N(\mathbb{R})$ ואנחנו רוצים למצוא את כל הוקטורים והערכים העצמיים שלה.

השיטה הישירה היא לפתור את הפולינום $\det(A - xI) = 0$ אבל כשמדובר בממדים גבוהים זה פרקטית בלתי אפשרי.

שיטת החזקות

שיטת החזקות מאפשרת למצוא את הערך העצמי המקסימלי בערך מוחלט. השיטה מתחילה מניחוש התחלתי אקראי.

<u>PowerMethod(A, u^0)</u>

1. Until convergence do
   a. $u^{n+1} = Au^n$
   b. $\lambda^{n+1} = \|u^{n+1}\|$
   c. $u^{n+1} = \frac{u^{n+1}}{\|u^{n+1}\|}$

אחרי שהאלגוריתם מתכנס u^n הוא וקטור עצמי עם ערך עצמי λ^n, שהרי אם התהליך התכנס אז $u^{n+1} = Au^n = u^n$ לפני הנירמול, ואחרי שבוחרים $\lambda^n = \|u^n\|$ ומנרמלים את u^n בדיוק מתקיים מה שצריך.

נראה שעבור מטריצות לכסינות התהליך אכן מתכנס ומתקבל הערך העצמי המקסימלי בערך מוחלט. נניח שהערכים העצמיים הם $|\lambda_1| > \cdots > |\lambda_N|$ ויש להם בסיס וקטורים עצמיים מתאימים $v_1, \dots, v_N$. אז אפשר לכתוב $u^0 = \sum_{i=1}^N a_i v_i$. לכן

$$u^{n+1} = Au^n = \sum_{i=1}^N \frac{a_i \lambda_i^n v_i}{c_n}$$

כאשר c_n קבוע נירמול. בכל מקרה, מאחר ש-$|\lambda_N| > \cdots > |\lambda_1|$, עבור n גדול נובע ש-$|\lambda_N|^n \gg \cdots \gg |\lambda_1|^n$ ולכן אם הניחוש הראשוני אקראי ניתן להניח ש-$a_1 \neq 0$ ואז האיבר המשמעותי בסכום הוא $a_1 \lambda_1^n v_1$ ושאר המחוברים הולכים ומתאפסים:

$$\begin{aligned} A^k u_0 &= a_1 A^k v_1 + a_2 A^k v_2 + \cdots + a_N A^k v_N = \\ &= a_1 \lambda_1^k v_1 + a_2 \lambda_2^k + \cdots + a_N \lambda_N^k v_N = \\ &= a_1 \lambda_1^k \left(v_1 + \frac{a_2}{a_1} \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 + \cdots + \frac{a_N}{a_1} \left(\frac{\lambda_N}{\lambda_1} \right)^k v_2 \right) \end{aligned}$$

מאחר ש-$1 > \left|\frac{\lambda_j}{\lambda_i}\right|$ ל-$i > j$ נובע שהביטוי בסוגריים שואף ל-v_1 וההתכנסות היא גיאומטרית בקצב $\left|\frac{\lambda_2}{\lambda_1}\right|$. אז אם שני הערכים העצמיים המקסימליים קרובים זה לזה השיטה מתכנסת לאט מאוד.

באופן פשטני, מצאנו את הערך העצמי המקסימלי. אבל ניתן לשדרג קצת את השיטה ולאפשר למצוא כמה ערכים עצמיים מרביים.

<u>m-PowerMethod$(A, u_1^0, \dots, u_k^0)$</u>

1. for $k = 1, \dots, m$
   a. until convergence do

i. $u_k^{n+1} = Au_k^n$

ii. $u_k^{n+1} = u_k^{n+1} - \sum_{j=1}^{k-1} u_j \langle u_j, u_k^{n+1} \rangle$

iii. $u_k^{n+1} = \frac{u_k^{n+1}}{\|u_k^{n+1}\|}$

b. $u_k = u_k^n$

הרעיון הוא שבכל פעם נוריד מהווקטור המשוער את כל הרכיבים בכיוונים של הווקטורים העצמיים שכבר נמצאו. ככה המקדמים שלהם ייעלמו מהפיתוח ונישאר רק עם הערך המקסימלי מבין הרכיבים שנותרו.

אלטרנטיבה פשוטה יותר היא פשוט לבחור וקטור ניחוש ראשוני שמאונך לכל הוקטורים העצמיים הקודמים אבל השיטה הזאת מאוד לא יציבה נומרית.

שינוי קל בפרמטרים של השיטה מאפשר לנו למצוא ערכים עצמיים מינימליים. אם נניח בנוסף ש-A מוגדרת חיובית, כלומר כל הערכים העצמיים שלה חיוביים ממש אז שיטת החזקות מוצאת את הערך העצמי המקסימלי ממש ולא רק מקסימלי בערכו המוחלט. לכן אם נפעיל את השיטה על $\lambda_{\max} \quad I - A$ נקבל את הערך העצמי המינימלי.

שיטת החזקות ההפוכה

אם נפעיל את שיטת החזקות על המטריצה $(A - \mu I)^{-1}$ נקבל את הערך העצמי המקסימלי של $(A - \mu I)^{-1}$. אבל הערכים העצמיים של מטריצה זו הם בדיוק $\frac{1}{\lambda_i - \mu}$ אז המקסימום מתקבל כאשר λ_i קרוב ל-μ:

<u>InversePowerMethod(A, μ, u^0)</u>

1. until convergence do
   a. solve $(A - \mu I)u^{n+1} = u^n$ //i.e. $u^{n+1} = (A - \mu I)^{-1} u^n$
   b. $\lambda^{n+1} = \|u^{n+1}\|$
   c. $u^{n+1} = \frac{u^{n+1}}{\|u^{n+1}\|}$

נניח ש-λ_1, λ_2 הם שני הערכים העצמיים הקרובים ביותר ל-μ. אז קצב ההתכנסות הוא $\frac{\lambda_2 - \mu}{\lambda_1 - \mu}$. ע"י בחירת μ כמה שיותר קרוב ל-λ_1 ניתן לקבל קצב התכנסות יותר ויותר גבוה. כמובן, אילו היינו יודעים את λ_1 לא הייתה לנו בעיה לפתור בכלל אבל בכל מקרה יש כל מיני כללי אצבע שאומרים איך לבחור את μ בצורה טובה.

אז קצב ההתכנסות יכול להיות מאוד גבוה אבל המחיר הוא שהשיטה כרוכה בהיפוך מטריצה או פיתרון מערכת משוואות בכל איטרציה. זה יכול לקחת הרבה זמן.

Jacobi Iteration

עד כה עבדנו על מטריצות כלליות. אם נניח שהמטריצה A סימטרית נוכל לפתח שיטה שמוצאת את כל הערכים העצמיים והרבה יותר יציבה נומרית מהקודמות.

אם A סימטרית אז לכל מטריצת סיבוב

$$V = R_\theta^{ij} = \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & \cos\theta & \dots & \sin\theta & \dots & 0\}i \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \dots & -\sin\theta & \dots & \cos\theta & \dots & 0\}j \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & \underbrace{0}_{i} & \dots & \underbrace{0}_{j} & \dots & 1 \end{pmatrix}$$

גם $A^{'} = V^t AV$ היא מטריצה סימטרית והיא דומה ל-A. ומתקיים

$$\begin{aligned} A_{ii}^{'} &= c^2A_{ii} - 2scA_{ij} + s^2A_{jj} \\ A_{jj}^{'} &= s^2A_{ii} = 2scA_{ij} + c^2A_{jj} \\ A_{ij}^{'} &= A_{ji}^{'} = (c^2 - s^2)A_{ij} + sc(A_{ii} - A_{jj}) \\ k \neq i,j &\Rightarrow A_{ik}^{'} = A_{ki}^{'} = cA_{ik} - sA_{jk} \\ k \neq i,j &\Rightarrow A_{jk}^{'} = A_{kj}^{'} = sA_{ik} + cA_{jk} \\ k,l \neq i,j &\Rightarrow A_{kl}^{'} = A_{kl} \end{aligned}$$

עבור $A_{ij} \neq 0$ ניתן למצוא θ כך ש-$A_{ij}^{'} = 0$ ע"י פתרון $\tan 2\theta = \frac{2A_{ij}}{A_{jj} - A_{ii}}$. בשיטת Jacobi חוזרים על התהליך הזה עד שהמטריצה הופכת להיות כמעט אלכסונית. אז האיברים שעל האלכסון שלה מהווים קירוב טוב לערכים העצמיים האמיתיים של המטריצה.

מטריצות הסיבוב הנ"ל נקראות **מטריצות Givens**.

JacobiIteration()

1. $V = I$
2. while $A_{ij} \neq 0, i \neq j$
   a. find R_θ^{ij} such that $\left[R_\theta^{ij} A \left(R_\theta^{ij}\right)^t\right]_{ij} = 0$
   b. $V = R_\theta^{ij} V$
   c. $A = R_\theta^{ij} A \left(R_\theta^{ij}\right)^t$
3. output $\mathrm{diag}\, A, V$

אפשר להראות שהשיטה באמת מתכנסת.

כל איטרציה באלגוריתם לוקחת $O(N)$ כי רק $O(N)$ איברים משתנים בכל סיבוב. אבל כדי שהשיטה תהיה יעילה כדאי לבחור בכל שלב A_{ij} מקסימלי מחוץ לאלכסון וזה כרוך בבדיקה של $O(N^2)$ איברים במטריצה. ויש $O(N^2)$ איטרציות. אז אם אנחנו בוחרים A_{ij} כלשהו השיטה יוצאת $O(N^3)$ אבל אם מחפשים את המקסימום זה אפילו $O(N^4)$.

שיטת ארנולדי

שיטת ארנולדי היא שיטה שמוצאת ערכים עצמיים של מטריצות כלליות (לא בהכרח סימטריות). שיטת החזקות לא משתמשת בערכי הביניים שהיא מחשבת. בשיטת ארנולדי מנסים להשתמש בכל המידע שחישבנו. נסתכל בווקטורים $b, Ab, A^2, \dots, A^{n-1}b$. בעזרת תהליך גראם-שמידט ניתן להפוך אותם לבסיס אורתוגונאלי של המרחב $K_n = \mathrm{span}\{b, Ab, \dots, A^{n-1}b\}$. הבסיס הזה מקרב את n הווקטורים העצמיים עם הערכים העצמיים הכי גדולים. הבעיה בתהליך כפי שהוא תואר למעלה היא שהחישוב הישיר הזה מאוד

לא יציב. במקום זה שיטת ארנולדי משתמשת בתהליך גראם-שמידט מיוצב שמייצר בסיס אורתונורמלי למרחב הזה.

<u>Arnoldi(A)</u>

1. let $q_1 \in \mathbb{R}^N$ such that $\|q_1\| = 1$
2. for $k = 2,3, \dots$
   a. $q_k = Aq_{k-1}$
   b. for $j = 1, \dots, k-1$
      i. $h_{j,k-1} = q_j^t q_k$
      ii. $q_k = q_k - h_{j,k-1} q_j$
   c. $h_{k,k-1} = \|q_k\|$
   d. $q_k = \frac{q_k}{h_{k,k-1}}$

התהליך מפסיק כאשר q_k הוא וקטור האפס ואז לא ניתן יותר לקבל וקטורים אורתונורמליים. נסמן ב-Q_n את המטריצה שהעמודות שלה הן n הווקטורים הראשונים $q_1, \dots, q_n$ שנמצאו ע"י התהליך. ונסמן ב-H_n את המטריצה המתקבלת ע"י התהליך (עם ערכי אפס במקומות שלא חושבו). אפשר להראות שמתקיים אז $H_n = Q_n^t A Q_n$. זה לא בדיוק דמיון מטריצות כי Q_n לא ריבועית אבל מפה נובע שאפשר לקרב את הערכים העצמיים ע"י חישוב של הערכים העצמיים של H_n. אם n לא מאוד גבוה ניתן לחשב את כל הערכים העצמיים שלה.

כל איטרציה פנימית של האלגוריתם כרוכה בפעולות על וקטורים ולכן לוקחת $O(N)$ פעולות. ואילו האיטרציה ה-k של האלגוריתם כרוכה בכפל מטריצה בווקטור ובלולאה הפנימית ולכן אם מדובר במטריצות דלילות לוקחת $O(N) + O(kN)$. אם יש סה"כ m איטרציות אז הסיבוכיות של האלגוריתם היא $O\big(m(N+mN)\big) = O(Nm^2)$. אם המטריצה היא מלאה אז יוצא $O(m^2N^2)$ כי כפל המטריצה לוקח יותר זמן. סיבוכיות הזיכרון היא $O(mN)$. לבסוף צריך גם להוסיף לזה את הזמן שלוקח לחשב את הערכים העצמיים של המטריצה המתקבלת H_m.

שיטת Lanczos

כאשר המטריצה סימטרית החישובים בשיטת ארנולדי הופכים קצת פשוטים יותר. במקרה זה ניתן לחשב ולהראות שהמטריצה H שמתקבלת היא מתאפסת פרט לאלכסון הראשי והשניים המשניים ויתר על כן, היא סימטרית. כלומר

$$H_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & & 0 \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & & & \\ & & & \ddots & \beta_{m-1} & \\ & & & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ 0 & & & & \beta_m & \alpha_m \end{pmatrix}$$

ולכן כעת בשיטת ארנולדי כל הלולאה הפנימית לוקחת $O(N)$ ולא $O(mN)$. כך כל השיטה יורדת ל-$O(mN)$ למטריצות דלילות ו-$O(mN^2)$ למטריצות מלאות.

פירוק QR

בהינתן מטריצה A, אילו יכולנו לפרק אותה ל-$A = QR$ כאשר Q מטריצת סיבוב ו-R מטריצה משולשית עליונה יכולנו להשתמש בפירוק הזה כדי לחשב ערכים עצמיים של A:

$$A_0 = A$$

$$A_{k+1} = R_k Q_k = Q_k^t Q_k R_k Q_k = Q_k^t A_k Q_k$$

ולכן כל המטריצות A_k דומות ויש להן ערכים עצמיים זהים. תחת תנאים מסוימים הסדרה A_k מתכנסת למטריצה משולשית ואז בעיית מציאת הערכים העצמיים נפתרת.

נותר להראות כיצד לחשב פירוק QR. יש שלוש דרכים לבצע זאת.

פירוק QR באמצעות תהליך גראם-שמידט

נבצע תהליך גראם-שמידט על העמודות של המטריצה A ונקבל את הווקטורים $q_{*1}, \dots, q_{*n}$. כמובן המטריצה שנוצרת ע"י וקטורים אלה היא מטריצת סיבוב. ע"י כמה מניפולציה אלגבריות ניתן להראות שהמטריצה $R = Q^t A$ היא משולשית עליונה. ומאחר ש-Q מטריצת סיבוב אכן מתקיים $A = QQ^t A = QR$.

היתרון בשיטה הזו הוא שהוא לא מסתמך על אף תכונה של המטריצה המקורית. את התהליך הזה ניתן לבצע לכל מטריצה ולקבל פירוק כמו שרצינו. ועלות הפירוק היא למעשה עלות החישוב של תהליך גראם-שמידט שזה $O(n^3)$.

פירוק QR באמצעות Householder Reflection

שיקוף Householder הוא טרנספורמציה לינארית שמשקפת וקטור סביב מישור כלשהו. נבנה מטריצות $Q_1, \dots, Q_n$ כך ש-$Q_n \cdot \ldots \cdot Q_1 A = R$ מטריצה משולשית עליונה ואז אם ניקח $Q = Q_1^t \dots Q_n^t$ נסיים.

נגדיר

$$v = \frac{a_{*1} - \|a_{*1}\| e_1}{\|a_{*1} - \|a_{*1}\| e_1\|}$$
$$Q_1 = I - 2vv^t$$

אפשר לחשב ולראות שמתקיים

$$(I - 2vv^t)A = \begin{pmatrix} * & \\ 0 & \\ \vdots & * \\ 0 & \end{pmatrix}$$

כלומר נקבל מטריצה שבעמודה הראשונה שלה כל האיברים פרט לראשון מתאפסים. כעת נמשיך באינדוקציה. נבצע את אותו התהליך עבור תת המטריצה שמתקבלת ע"י הורדת השורה ועמודה הראשונה ונקבל מטריצה $\tilde{Q}_2$. אז נגדיר

$$Q_2 = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \vdots & & \tilde{Q}_2 & \\ 0 & & & \end{pmatrix}$$

כך נוכל להמשיך עד שנקבל את מה שרצינו.

העלות החישובית נובעת בעיקר מהכפלה $Q_n \dots Q_1 A$. אבל Q_i נוצרות ע"י vv^t שהיא מטריצה שכל השורות שלה הן כפולות אחת של השנייה. מכפלה במטריצה כזאת ניתן לבצע ב-$O(n^2)$. לכן כל שלב באלגוריתם לוקח $O(n^2)$ ולבסוף כדי לחשב את Q צריך לכפול n כאלה מה שמביא אותנו ל-$O(n^3)$.

פירוק QR באמצעות מטריצות Givens

מטריצת Givens היא מטריצת סיבוב כמו שהשתמשנו בשיטת Jacobi iteration. ניתן לעשות תהליך דומה לאלימינציה של גאוס שבו בכל איטרציה מאפסת איבר מתחת לאלכסון הראשי של A. אז מכפלת כל מטריצות הסיבוב שהשתמשנו בהן נותנת את Q וההפעלה שלהן על A נותנת את R. הסיבוכיות המתקבלת

היא $O(n^3)$ כי יש $O(n^2)$ איברים שצריך לאפס ולכל אחד מהם צריך $O(n)$ פעולות (ראינו קודם שמכפלה שמטריצת סיבוב כזאת משנה רק $O(n)$ איברים במטריצה).

אופטימיזציה

<u>הבעיה</u>: נתונה פונקציה רציפה $f:\mathbb{R}^d \to \mathbb{R}$ ורוצים למצוא נקודת מינימום מקומית x.

Bracketing

כאשר הפונקציה מוגדרת על $\mathbb{R}$, כלומר הבעיה חד-ממדית, ניתן לפעול במעין חיפוש.

נתחיל עם שלשה $a < b < c$ כך ש-$f(b) < f(a), f(c)$ ונבחר $x \in (a,b)$ כלשהו. יש שתי אפשרויות:

- אם $f(x) < f(b)$ נמשיך באותו אופן עם השלשה $a < x < b$
- אם $f(x) \geq f(b)$ נמשיך באותו אופן עם השלשה $x < b < c$

כך בכל שלב אנחנו נמצאים באותו מצב התחלתי שבו יש שלוש נקודות והאמצעית מקבל ערך נמוך יותר מהקיצוניות. בכל של המרחק בין נקודות הקצה קטן ולבסוף מתכנסים לנקודת מינימום מקומית.

למעלה בחרנו נקודה $x \in (a,b)$ אבל ברור שבאותה מידה ניתן היה לבחור נקודה $x \in (b,c)$. למעשה, עדיף לשלב וניתן להשתכנע שככלל בכל שלב עדיף לבחור נקודה בתוך הקטע הרחב יותר. עדיין לא ברור באיזו נקודה בדיוק כדאי לבחור.

סביר לחשוב שכדאי שבכל שלב הבעיה תהיה דומה לבעיה הקודמת מבחינת היחסים בין מרחקי הנקודות. חישוב שמתבסס על דמיון עצמי זה מראה שבכל שלב כדאי לבחור את הנקודה x בתוך הקטע הרחב כך שהיא מחלקת אותו לפי יחס הזהב $\phi = \frac{\sqrt{5}+1}{2}$.

וריאציה אחרת לבחירת x מתבססת על אינטרפולציה של הפונקציה. ישנן שלוש נקודות אז ניתן להתאים אותן ע"י פרבולה ומאחר שנקודות הקצה גבוהות יותר, הפרבולה תהיה בוכה ותקבל מינימום. שם אפשר לבחור את הנקודה החדשה x. במקרה זה יש סכנה לא להתרחק הרבה מ-b ולמעשה "להיתקע במקום" במקום להתקדם. בפרט, אם במקרה b היא נקודת המינימום של הפרבולה, אז לעולם לא נמצא נקודה אחרת.

לכן, בפועל בד"כ משלבים בין האסטרטגיות.

Downhill Simplex

את השיטה הקודמת אי אפשר להרחיב לפונקציות רב-ממדיות. Downhill Simplex הוא כלל אצבע שמאפשר להתמודד עם פונקציות רב-ממדיות וההנחה היחידה היא שהפונקציה המדוברת היא רציפה. הבעיה עם השיטה היא שמדובר רק כלל אצבע והשיטה עלולה להתכנס גם לנקודות שאינו נקודות קיצון.

נניח שיש לנו $n+1$ נקודות $x_1, \dots, x_{n+1}$. השיטה עובדת ע"י הפעלת הפעולות הבאות:

1. מסדרים את הנקודות כך ש-$f(x_1) \leq \cdots \leq f(x_{n+1})$
2. מחשבים את מרכז הכובד $x_0 = \frac{\sum_{i=1}^{n+1} x_i}{n+1}$ של הנקודות
3. **שיקוף** (reflection): מחשבים את הנקודה המשוקפת $x_r = x_0 + \alpha(x_0 - x_{n+1})$ עבור $\alpha > 0$.
   אם $f(x_1) \leq f(x_r) < f(x_n)$ אז מחליפים את x_{n+1} ב-x_r וממשיכים שוב לשלב (1).
4. **הרחבה** (expansion): אם $f(x_r) < f(x_1)$ בודקים אם אפשר להמשיך: מחשבים את הנקודה המורחבת $x_e = x_0 + \gamma(x_0 - x_{n+1})$. אם $f(x_e) < f(x_r)$ מחליפים את x_{n+1} ב-x_e וחוזרים לשלב (1). אחרת, מחליפים את x_{n+1} ב-x_r וחוזרים לשלב (1).
   אם $f(x_r) \geq f(x_1)$ (כלומר לא הצלחנו לשפר) ממשיכים לשלב הבא.
5. **כיווץ** (contraction): אז בהכרח מתקיים $f(x_r) \geq f(x_n)$. מחשבים את הנקודה המכווצת $x_c = x_{n+1} + \rho(x_0 - x_{n+1})$. אם $f(x_c) \leq f(x_{n+1})$ מחליפים את x_{n+1} ב-x_c וחוזרים לשלב (1). אחרת, ממשיכים לשלב הבא.

6. **רדוקציה** (reduction): לכל $2 \le i \le n+1$ מחשבים $x_i = x_1 + \sigma(x_i - x_1)$ וחוזרים לשלב (1).

ערכים אופייניים של $\alpha, \gamma, \rho, \sigma$ הם $\alpha = 1, \gamma = 2, \rho = \frac{1}{2}, \sigma = \frac{1}{2}$.

בשלב השיקוף, מאחר ש-x_{n+1} היא הנקודה הכי גרועה מצפים למצוא ערך נמוך יותר בשיקוף שלה. בשלב ההרחבה אם הצלחנו לשפר מנסים לבדוק אם אפשר להוריד את הערך ע"י הליכה נוספת באותו כיוון ובשלב הכיווץ, אם $f(x_r) > f(x_n)$ מצפים למצוא ערך נמוך יותר בתוך האזור שנוצר ע"י כל הנקודות.

לנקודות ההתחלתיות יש השפעה על התקדמות התהליך ותמיד כדאי לבחור נקודות שנראות מתאימות לבעיה הספציפית שמנסים לפתור.

Line Minimization

נתונה פונקציה $f: \mathbb{R}^n \to \mathbb{R}$. אם קיים ממזער חד-ממדי שיכול לפתור בעיות מהצורה $\min_\alpha f(x + \alpha d)$ עבור וקטורי כיוון $d \in \mathbb{R}^n$ ניתן להשתמש בו כדי למזער את הפונקציה הרב-ממדית. יש כל מיני ממזערים כאלה. למשל, bracketing, steepest descent, שימוש בשיטת Newton-Raphson כדי למצוא שורש של הנגזרת (אם הפונקציה גזירה) וכו'. בכל מקרה העניין החשוב לדון בו הוא אסטרטגיית בחירת כיווני החיפוש.

שיטת Powell עובדת בכמה איטרציות. מתחילים מנקודה כלשהי x_0. ראשית, כיוון החיפוש הראשון הוא $d_1 = e_1$ וקטור היחידה הראשון. מחשבים את $\alpha = \text{argmin}_\alpha f(x_0 + \alpha d_1)$. מגדירים $x_1 = x_0 + \alpha d_1$ וממשיכים משם למזער בכיוון $d_2 = e_2$. כך ממשיכים עד שעוברים על כל וקטורי היחידה ונוצרות נקודות $x_0, x_1, \dots, x_n$. כעת בוחרים d_i באקראי ומחליפים אותו ב-$x_n - x_0$ ועכשיו שוב ממזערים לפי כל הכיוונים. ככה ממשיכים עד להתכנסות.

הבעיה העיקרית עם השיטה הזו היא שאם הייתה ירידה משמעותית בערך הפונקציה לפי אחד הכיוונים $x_n - x_0$ יהיה מאוד קרוב לכיוון הזה ואז ע"י בחירה אקראית של כיווני חיפוש והחלפתם בכיוון הזה אנחנו נאבד את היכולת לחפש בכיוונים אחרים. פתרון אפשרי לזה הוא שיטת Powell המותאמת. שם לא בוחרים באקראי את הכיוון לאותו נחליף אלא בוחרים דווקא את הכיוון שתרם הכי הרבה למזעור הפונקציה. כלומר, בוחרים את $i = \text{argmax}_i |f(x_i) - f(x_{i-1})|$. זה בדיוק פותר את הבעיה הנ"ל.

שימוש בנגזרות

אם הפונקציה דיפרנציאלית פעמיים ניתן לקרב אותה באמצעות פולינום ריבועי:

$$f(x+h) = f(x) + \nabla f(x) h + \frac{1}{2} h^t H_f(x) h + o(\|h\|^3)$$

כאשר $\left[H_f(x_0)\right]_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}(x_0)$ מטריצת ההסיאן. כדי למזער את $f(x+h)$ מספיק למזער את

$$F_x(h) = \nabla f(x) h + \frac{1}{2} h^t H_f(x) h$$

וזאת פונקציה ליניארית שאותה אנחנו יודעים למזער ע"י CG או ע"י SD.

כעת נוכל לעשות תהליך איטרטיבי. נתחיל מ-x_0 כלשהו ומשם בכל שלב נקבל $x_n = x_{n-1} + h$ כאשר h ממזער את $F_{x_{n-1}}(h)$.

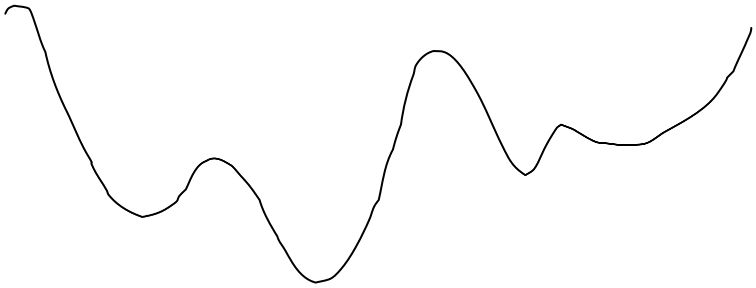
Simulated Annealing

עד כה התעסקנו במציאת נקודות קיצון מקומיות. עכשיו נראה איך אפשר למצוא מינימום גלובאלי באמצעות שיטה אקראית.

בהינתן קבוע T ניתן להתהלך על הפונקציה בהילוך מקרי באופן הבא: מתחילים בנקודה x^0 כלשהי. בכל שלב מחשבים $x^* = x^n + r$ כאשר $r \sim \mathcal{N}(0, \varepsilon^2)$ משתנה אקראי נורמלי. בסיכוי $p = e^{\frac{f(x^n) - f(x^*)}{T}}$ קובעים $x^{n+1} = x^*$ ואחרת נשארים במקום עם $x^{n+1} = x^n$.

הקבוע T נקרא **טמפרטורה** ואם הוא נמוך אז נעבור לנקודה החדשה x^* אמ"מ $f(x^*) < f(x^n)$. אם הטמפרטורה גבוהה מדי אז ברוב המקרים נעבור לנקודה החדשה.

בשיטת ה-simulated annealing מתחילים מטמפרטורה גבוהה ולאט לאט מקטינים אותה. כאשר הטמפרטורה גבוהה המערכת יכולה "לקפוץ" רחוק ולצאת מנקודת מינימום לוקאלית שהיא נמצאת בה אך אינה מינימום גלובאלי, למשל אם הפונקציה נראית כך



השיטה יכולה לעזור.

משוואות דיפרנציאליות רגילות

משוואה דיפרנציאלית היא משוואה שמגדירה יחס בין פונקציה לבין נגזרותיה. משוואה דיפרנציאלית היא **רגילה** אם הפונקציה היא במשתנה יחיד, כלומר לא מעורבות במשוואה נגזרות חלקיות. **סדר** המשוואה הוא סדר הנגזרת הכי גבוהה שמופיעה במשוואה.

ידועות שיטות אנליטיות לפתרון משוואות דיפרנציאליות מכמה צורות אבל באופן כללי זה יכול להיות מאוד מסובך ואולי אפילו לא אפשרי. לכן שיטות נומריות לפתרון משוואות דיפרנציאליות הן כלי חשוב מאוד.

<u>**משפט**</u>: נתבונן במשוואה $y'(t) = f\big(t, y(t)\big)$ כאשר f פונקציה רציפה בפרמטר הראשון וליפשיצית במשתנה השני. אזי לכל תנאי התחלה $y(t_0) = y_0$ קיימת פונקציה יחידה $y(t)$ גזירה ברציפות המקיימת את המשוואה ואת תנאי ההתחלה.

זהו משפט קיום שמבטיח שיש שאם הפונקציה "מספיק יפה" יש פתרון למשוואות דיפרנציאליות מסדר ראשון. ע"י מניפולציות אלגבריות ניתן להפוך כל משוואה מסדר גבוה יותר למערכת משוואות מסדר ראשון שניתן לפתור באופן סדרתי. ולכן מתקבל גם משפט קיום למשוואות מסדר גבוה ונובע שמספיק לטפל במשוואות מהצורה $y'(t) = f\big(t, y(t)\big)$.

נניח אם כן שיש לנו משוואה כזאת ונניח שאנחנו עובדים בדיסקרטיזציה של h בזמן. אז יש לנו וקטור זמן $t_j = jh$. נסמן ב-$y_j = y(t_j)$ את הערכים האמיתיים של הפונקציה בנקודות הזמן שלנו ונסמן ב-w_j את הקירובים שאנחנו נחשב.

שיטות אוילר

קירוב טיילור של y מסדר ראשון בנקודה t_j נותן לנו

$$w_{j+1} \approx y_{j+1} = y_j + hy_j' = y_j + hf\big(t_j, y_j\big) \approx w_j + hf\big(t_j, w_j\big)$$

כלומר

$$\frac{w_{j+1} - w_j}{h} = f\big(t_j, w_j\big)$$

קירוב זה נקרא **שיטת אוילר הקדמית** זוהי שיטה מפורשת כי אחרי שחישבנו את w_j ניתן באופן ישיר לחשב מהשוויון הנ"ל את w_{j+1}.

מאידך, שוב בשימוש בקירוב טיילור, ניתן לכתוב

$$\begin{aligned} w_j \approx y_j &= y\big(y_{j+1} - h\big) = y\big(t_{j+1}\big) - hy'\big(t_{j+1}\big) = \\ &= y_{j+1} - hf\big(t_{j+1}, y_{j+1}\big) \approx w_{j+1} - hf\big(t_{j+1}, w_{j+1}\big) \end{aligned}$$

ושוב ע"י העברת אגפים נקבל

$$\frac{w_{j+1} - w_j}{h} = f\big(t_{j+1}, w_{j+1}\big)$$

זוהי **שיטת אוילר האחורית** והיא נותנת ייצוג סתום של הקירוב. בכל מקרה משוואות האלה נקראות **משוואות הפרשים**.

על אף שבשיטת אוילר האחורית קשה יותר לבודד את השערון של הפונקציה, השיטה עובדת בד"כ טוב יותר וניתן להשתמש בה אפילו בדיסקרטיזציות גסות יותר.

נדבר כעת על שיטת אוילר הקדמית, אם כי ניתן היה לעשות את אותו הניתוח עבור השיטה האחורית.

נגדיר את **שגיאת הקיטוע** (truncation error)

$$\tau_j = \frac{y_{j+1}-y_j}{h} - f\left(t_j, y_j\right)$$

זהו מדד לכמה שהפונקציה מקיימת את נוסחת ההפרשים.

שגיאת הקיטוע המקומית היא $r_j = \tau_j h = y_{j+1} - y_j - hf\left(t_j, y_j\right)$.

נאמר שמשוואת הפרשים היא **עקבית** אם שגיאת הקיטוע שואפת לאפס ככל שהדיסקרטיזציה קטנה, כלומר אם $\tau_j \underset{h\to 0}{\longrightarrow} 0$. נשים לב שמשוואת אוילר הקדמית היא אכן עקבית מאחר שבאמצעות קירוב טיילור נקבל

$$\tau_j = \frac{y_{j+1}-y_j}{h} - f\left(t_j, y_j\right) = \frac{y_j + hf\left(t_j, y_j\right) + o(h^2) - y_j}{h} - f\left(t_j, y_j\right) =$$
$$= f\left(t_j, y_j\right) - f\left(t_j, y_j\right) + o(h) = o(h) \underset{h\to 0}{\longrightarrow} 0$$

נגדיר את **השגיאה הגלובלית** להיות $e_j = y_j - w_j$ ונאמר ששיטה מתכנסת אם $e_{\frac{T}{h}} \underset{h\to 0}{\longrightarrow} 0$ עבור זמן קבוע T כלשהו.

אפשר להראות שהשגיאה הגלובלית של שיטת אוילר הקדמית עולה מעריכית עם הזמן T. היא אמנם מתכנסת לאפס ככל שהדיסקרטיזציה קטנה יותר, אך אם הדיסקרטיזציה קבועה החסם על השגיאה עולה מעריכית. לכן, כאשר יש תהליכים שמתוארים ע"י משוואות דיפרנציאליות (כמו למזל מזג האוויר) קשה לתת תחזיות מדויקות לזמנים רחוקים.

שיטת Mid-Step

נשתמש בקירוב טיילור מסדר שני:

$$y_{j+1} = y_{j+\frac{1}{2}} + \frac{h}{2} y'_{j+\frac{1}{2}} + \frac{h^2}{8} y''_{j+\frac{1}{2}} + o(h^3)$$
$$y_j = y_{j+\frac{1}{2}} - \frac{h}{2} y'_{j+\frac{1}{2}} + \frac{h^2}{8} y''_{j+\frac{1}{2}} + o(h^3)$$
$$\Rightarrow \frac{y_{j+1}-y_j}{h} = y'_{j+\frac{1}{2}} + o(h^2)$$

זה נותן מוטיבציה להשתמש במשוואה ההפרשים $\frac{y_{j+1}-y_j}{h} = f\left(t_{j+\frac{1}{2}}, w_{j+\frac{1}{2}}\right)$. אבל קירוב טיילור מסדר ראשון מאפשר לנו להגדיר $w_{j+\frac{1}{2}} = w_j + \frac{h}{2} f\left(t_j, w_j\right)$, אז סה"כ מתקבל לנו

$$\frac{y_{j+1}-y_j}{h} = f\left(t_{j+\frac{1}{2}}, w_{j+\frac{1}{2}}\right) = f\left(t_{j+\frac{1}{2}}, w_j + \frac{h}{2} f\left(w_j, t_j\right)\right)$$

ושגיאת הקיטוע היא

$$\tau_j = \frac{y_{j+1}-y_j}{h} - f\left(t_{j+\frac{1}{2}}, y + \frac{h}{2} f\left(y_j, t_j\right)\right)$$

אפשר להראות שגם שיטה זו היא עקבית.

נספח א – פתרונות למבחנים

2008 מועד א

<u>חלק א</u>

1. בהינתן m ערכי פונקציה מפוזרים במרחב, מהו מספר הפעולות ששיטת Shepard מבצעת לאורך הערכת נקודת אינטרפולציה חדשה?
   a. $O(1)$
   b. **$O(m)$ (השיערוך מתבצע ע"י סיכום של m מחוברים שהחישוב של כל אחד מהם הוא $O(1)$, אם להתעלם מהצורך לחשב נורמה ב-$\mathbb{R}^d$)**
   c. $O(m^2)$
   d. $O(\log m)$
2. האם הקונבולוציה עם הפילטר $[0.2,0.5,0.75,1,0.75,0.5,0.2]$ מממשת אינטרפולציה פולינומיאלית ממעלה כלשהי?
   לא! כי היא לא מצליחה אפילו לעשות אינטרפולציה של פונקציה קבועה.
3. מהו סדר השגיאה של פולינום אינטרפולציה P ממעלה m לפונקציה f כאשר נקודות האינטרפולציה מפוזרות במרווחים שווים בקטע $[0,1]$? ומהו הדיוק של האינטגרל של f בקטע ע"י אינטגרל הפולינום?
   a. $|f(x)-P(x)|=O(h^m), |I_f-I_P|=O(h^m)$
   b. $|f(x)-P(x)|=O(h^{m+1}), |I_f-I_P|=O(h^m)$
   c. $|f(x)-P(x)|=O(h^m), |I_f-I_P|=O(h^{m+1})$
   d. **$|f(x)-P(x)|=O(h^{m+1}), |I_f-I_P|=O(h^{m+1})$**
4. מהו סדר השגיאה של אינטגרציית Newton-Cotes כאשר משתמשים בפולינום למקוטעין מסדר m במקטעים מאורך h על מנת לקרב אינטגרל על קטע בעל אורך קבוע?
   a. $O(h)$
   b. $O(h^{m-1})$
   c. $O(h^m)$
   d. **$O(h^{m+1})$**
5. בשיטת מונטה-קרלו השגיאה יורדת עם מספר הדגימות N כמו
   a. $\frac{1}{N}$
   b. $\frac{1}{N^2}$
   c. **$\frac{1}{\sqrt{N}}$**
   d. אינה יורדת
6. לאיזו מהפונקציות הבאות (המוגדרות בקטע $[0,1]$) נצפה לשגיאה נמוכה ביותר בשיטת מונה-קרלו כאשר בוחרים את $g\equiv 1$?
   a. $f(x)=\begin{cases}1, & 0<x<\frac{1}{2}\\ 0, & \text{else}\end{cases}$
   b. $f(x)=\begin{cases}1, & 0<x<\frac{1}{4},\frac{1}{2}<x<\frac{3}{4}\\ 0, & \text{else}\end{cases}$
   c. **$f(x)=\begin{cases}1, & 0<x<\frac{1}{10}\\ 0, & \text{else}\end{cases}$ (כי הפונקציה הזאת הכי דומה לפונקציה קבועה)**
7. איזו טענה נכונה לגבי שיטת החצייה?
   a. שיטה זו מתאימה לפונקציות לא רציפות

b. **שיטה זו מתכנסת בקצב גיאומטרי עם פקטור $\frac{1}{2}$**
c. שיטה זו לא מתאימה לפונקציות לא גזירות
d. השיטה אינה איטרטיבית

8. התהליך $x^{n+1} = \frac{x^n}{2} + \frac{1}{x^n}$ מתאר את איטרצית ניוטון עבור
a. $f(x) = x^2 - 2$ **$(x^n - \frac{f(x^n)}{f'(x^n)} = x^n - \frac{\frac{x^n}{2} + \frac{1}{x^n}}{\frac{1}{2} - \frac{1}{(x^n)^2}} = \frac{x^n}{2} + \frac{1}{x^n})$**
b. $f(x) = x - 2$
c. $f(x) = x^2 - 1$
d. $f(x) = \frac{x^2}{2} + 1$

9. לכל מטריצה מוגדרת חיובית ניתן לחשב פירוק LU.
כן! אם המטריצה מוגרת חיובית אז כל מטריצה עיקרית שלה היא מוגדרת חיובית ולכן ניתן לפרק.

01. בעזרת פירוק LU של המטריצה המלאה $A \in M_n(\mathbb{R})$ ניתן לחשב את מערכת המשוואות $Ax = LUx = b$ בזמן:
a. $O(n)$
b. **$O(n^2)$ (כי אם כבר יש לנו את הפירוק רק צריך לעשות הצבה לאחור פעמיים)**
c. $O(n^3)$
d. אין הפירוק מקל את החישוב עבור מטריצות מלאות

11. מהי מורכבות החישוב של שיטות Gauss-Seidel ו-Jacobi לפתרון מערכת משוואות עם מטריצה מלאה?
a. $O(N)$
b. **$O(N^2)$ (בהנחה שמספר האיטרציות בלתי תלוי בממש המטריצה)**
c. $O(N^3)$
d. שיטות אלה לא מתכנסות עבור מטריצות מלאות

21. האם מספר המצב של המטריצה $\begin{pmatrix} 1 & 10 \\ 10 & 1 \end{pmatrix}$ ניתן לשיפור ע"י מטריצות הבידוד הבאות?

31. שיטת Jacobi ללכסון מטריצה הינה שיטה איטרטיבית (בניגוד לישירה)?
כן!

41. איזו מהמשוואות הבאות עקבית עם המשוואה הדיפרנציאלית
$$y^{n+1} = y^n + \Delta t(ay^{n+1} + (a + \Delta t)y^n + 1 + \Delta t) + 2(\Delta t)^2$$
a. $y'(t) = ay(t) + 3$
b. $y'(t) = 2ay(t) + 1 + 3t$
c. **$y'(t) = 2ay(t) + 1$ כי:**
$$\frac{y^{n+1} - y^n}{\Delta t} = ay^{n+1} + (a + \Delta t)y^n + 1 + \Delta t + 2\Delta t \xrightarrow[\Delta t \to 0]{} ay + ay + 1$$
d. $y'(t) = \frac{a+1}{y(t)} + t + 3$

<u>חלק ב</u>

1. בהינתן מטריצה A שאינה סימטרית או מוגדרת חיובית, נציע דרך לפתור את $Ax = b$ בעזרת שיטת ה-Conjugate gradients. נניח שהמטריצה הפיכה. אזי מספיק לפתור את מערכת המשוואות $A^tAx = A^tx$. המטריצה A^tA היא סימטרית ומוגדרת חיובית ולכן ניתן להשתמש בשיטות שלמדנו כדי לפתור אותה. החיסרון העיקרי בשיטה הזו הוא שמספר המצב של המטריצה החדשה גדל ריבועית.
2. בהינתן מטריצה בעלת ערכים עצמיים ב-$[0,1)$ נציע דרך לחישוב הערך העצמי הקטן ביותר שגדול מאפס בשימוש בשיטת החזקות ההפוכה בזמן $O(-\log_2 \varepsilon)$ פעמים כאשר ε הדיוק המבוקש. נבצע

חיפוש בינארי. נתחיל מלחפש את הערך העצמי הכי קרוב ל-$\frac{1}{2}$. אם התקבלה תוצאה $\lambda < \frac{1}{2}$ נמשיך לחפש בקטע $[0,\lambda]$. אחרת, נעבור לקטע $\left[0,\frac{1}{2}-\lambda\right]$. בכל קטע מחפשים את הערך שהכי קרוב למרכז הקטע. ככה אורך הקטעים קטן פי כשניים ולבסוף נתכנס לתוצאה.

3. נניח שהפונקציה f גזירה וחסומה בקטע $[0,1]$ למעט בנקודה $x=\frac{1}{2}$ שבה הפונקציה אינה רציפה. נקרב את האינטגרל בקטע ע"י שיטת הטרפז. נניח שמחשבים באמצעות h-ים קטנים. אז בכל קטע שמכיל את $\frac{1}{2}$ השגיאה תהיה $O(h)$ ועל כל האחרים $O(h^2)$, לכן סה"כ $O(h)$. כעת, אילו אורך הקטע היה גדול מ-1 ניתן היה להשתמש גם ב-h-ים גדולים מ-1 ועבורם דווקא השגיאה הריבועית תהיה דומיננטית.

2008 מועד ב

חלק א

1. איזו מהטענות הבאות נכונה לגבי שיטת Shepard?
   - **a. אינה חורגת מערכי קיצון**
   - b. חורגת מערכי קיצון אך משמרת סימן
   - c. חורגת מערכי קיצון ולא משמרת סימן
   - d. דורשת פתרון של מערכת משוואות

2. האם הקונבולוציה עם הפילטר $[-0.0625,0,0.5625,1,0.5625,0,-0.0625]$ מממשת אינטרפטולציה פולינומיאלית ממעלה ראשונה או יותר?

   הפילטר לא מצליח לחשב אינטרפולציה של פונקציה קבועה ולכן הוא לא יכול לייצג אינטרפולציה ממעלה כלשהי.

3. מהו סדר השגיאה של שיטת סימפסון כאשר משתמשים במקטעים מאורך h כדי לקרב אינטגרל על $[a,b]$?
   - a. $O(h)$
   - b. $O(h^2)$
   - c. $O(h^3)$
   - **d.** $\boldsymbol{O(h^4)}$

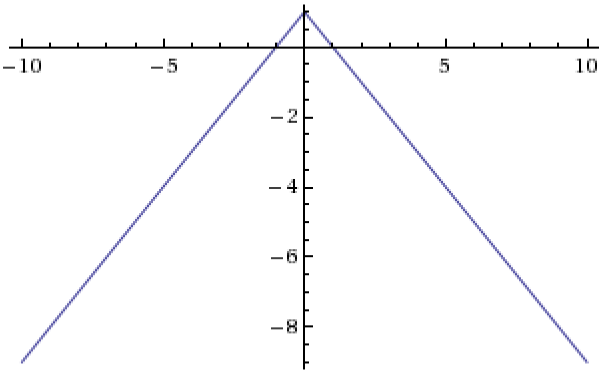
4. מהו קצב ירידת השגיאה של שיטת מונטה-קרלו עבור אינטגרציית הפונקציה $f(x)=\sin\frac{1}{x}$ בקטע $[0,1]$?
   - a. $\frac{1}{N}$
   - b. $\frac{1}{N^2}$
   - **c.** $\boldsymbol{\frac{1}{\sqrt{N}}}$
   - d. אינה יורדת

5. התהליך $x^{n+1}=2x^n-\frac{(x^n)^2}{2}$ מתאר איטרצית ניוטון עבור
   - **a.** $\boldsymbol{f(x)=\frac{1}{x}-\frac{1}{2}}$ $\boldsymbol{\left(x^n-\frac{f(x^n)}{f'(x_n)}=x_n-\frac{\frac{1}{x^n}-\frac{1}{2}}{-\frac{1}{(x^n)^2}}=x^n+x^n-\frac{(x^n)^2}{2}=2x^n-\frac{(x^n)^2}{2}\right)}$
   - b. $f(x)=x-\frac{1}{2}$
   - c. $f(x)=\frac{1}{x^2}-\frac{1}{2}$
   - d. $f(x)=\frac{1}{x^2}+2$

6. כיצד תפעל שיטת ניוטון לפתרון $f(x)=1-|x|$?
   - **a. לא תתכנס לפתרון מאחר שתנאי השיטה אינם מתקיימים**
   - b. תקרב את הפיתרון אך לא במדויק

c. **תפתור את המשוואה בצעד אחד**

התשובה תלויה בנקודה שמתחילים ממנה. אם נתחיל ב-0 הפונקציה לא גזירה שם ולכן אי אפשר להמשיך. אבל אם נתחיל בכל נקודה שאינה אפס, המשיק לפונקציה שם מתלכד עם הפונקציה והבעיה תיפתר בצעד אחד:



7. בהינתן מטריצה מוגדרת אי-שלילית A, באילו שיטות ניתן לפתור את $Ax = b$?
   a. **חילוץ גאוס**
   b. פירוק LU
   c. Conjugate Gradients
   d. Gauss-Seidel
8. מהי מורכבות החישוב של שיטת conjugate gradients?
   a. $O(N)$ עבור מטריצות מלאות
   b. $O(N^2)$ עבור מטריצות דלילות
   c. $O(N^3)$ עבור מטריצות מלאות
   d. **$O(N)$ עבור מטריצות דלילות (בהנחה שמספר האיטרציות אינו תלוי ב-N)**
9. מי מהמטריצות הבאות מהווה מטריצת בידוד הטובה ביותר עבור המטריצה $\begin{pmatrix}1 & 1\\ 0 & 0.1\end{pmatrix}$? השאלה בוטלה.
10. האם ניתן להשתמש בשיטת החזקות על מטריצה שאינה מוגדרת חיובית?
   כן, אבל התוצאה תהיה הערך העצמי בעל הערך המוחלט הגבוה ביותר ולא בהכרח הערך העצמי הגבוה ביותר.
11. איזו מהמשוואות הבאות מקרבת את הפתרון עבור $y'(t) = ay(t) + t + \frac{1}{y(t)} + 1$?
   a. $\boldsymbol{y^{n+1} = y^n + \Delta t + \Delta t\left(ay^{n+1} + \frac{1}{y^{n-1}} + \Delta t\right) + t\Delta t + y^n(\Delta t)^2}$
   b. $y^{n+1} = y^n + \Delta t + \Delta t a y^n + \frac{1}{y^n} + \Delta t + t\Delta t$
   c. $y^{n+1} = y^n + \Delta t + \Delta t\left(ay^n + \frac{1}{y^n} + \Delta t\right) + t$
   d. $y^{n+1} = y^n + \Delta t(1 + t) + \Delta t\left(ay^n + \frac{\Delta t}{y^n} + \Delta t\right)$

<u>חלק ב</u>

1. נתאר שיטה לקירוב אינטגרל של פונקציה חלקה פרט למספר סופי של נקודות. במקרה כזה ניתן לחלק את תחום האינטגרציה למספר סופי של קטעים שבכל אחד מהם הפונקציה חלקה. הכול קטע כזה ניתן להשתמש בכל שיטה שלמדנו (למשל שיטת הטרפז תיתן דיוק של $O(h^2)$). לבסוף נותר רק לסכום את האינטגרלים מכל קטע. מאחר שמספר הקטעים סופי וקבוע הדיוק עדיין נשאר כמו הדיוק של השיטה שהשתמשנו בה בשביל החישוב בתוך כל קטע.

2. נתון תהליך איטרטיבי $Qx^{n+1} = (Q - A)x^n + b$ שבו למטריצת החלוקה Q יש אותם וקטורים עצמיים כמו ל-A עם אותם הערכים העצמיים עד כדי פקטור שבין 2 ל-3. התהליך מתכנס אם קיימת נורמה שבה $\|I - Q^{-1}A\|_k < 1$. מתקיים

$$\|I - Q^{-1}A\| = \max_{\|x\|=1} \|(I - Q^{-1}A)x\| \geq \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector}}} \|(I - Q^{-1}A)v\| =$$

$$= \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector}}} \|v - Q^{-1}Av\| = \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector}}} \|v - Q^{-1}\lambda v\| =$$

$$= \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector}}} \left\|v - \frac{\lambda}{\mu}v\right\| = \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector}}} \left|1 - \frac{\lambda}{\mu}\right| \|v\| = \max_{\substack{\|v\|=1 \\ v \text{ is eigenvector} \\ \text{with eigenvalue } \lambda \text{ in } A \\ \text{and } \mu \text{ in } Q}} \left|1 - \frac{\lambda}{\mu}\right|$$

כעת, אם הכוונה היא שהמטריצות לכסינות (וזו אכן הייתה הכוונה) אז הביטוי הזה ממש שווה לנורמה האופרטורית (כי מספיק להסתכל על וקטורים עצמיים) ואז אכן נובע שהשיטה מתכנסת וקצב ההתכנסות הוא לפחות $1 - \frac{1}{3} = \frac{2}{3}$.

3. בהינתן מטריצה בעלת מספר מצב גבוה הנובע ממספר קטן של וקטורים עצמיים קטנים במיוחד (והשאר בקירוב 1), נחפש פיתרון ל-$Ax = b$ ע"י מציאת משפר מצב ע"י מציאת הוקטורים האלה בעזרת שיטת החזקות ההפכית. זו לא שיטה יעילה משום שבשיטת החזקות ההפוכה צריך לפתור מערכת משוורות שכרוכה ב-A, וזה בלאו הכי מה שאנחנו צריכים לעשות. אז כבר אפשר היה להפוך את A מלכתחילה עם גאוס וזהו.

4. מהו סדר ההתכנסות של שיטת ניוטון? את זה עשינו קודם. אם רוצים לראות מה קורה כאשר הנגזרת השנייה מתאפסת אפשר לעשות משהו דומה ולהוסיף עוד איבר בפיתוח טיילור של הפונקציה סביב השורש. אז יוצר שהשיטה מתכנסת אפילו יותר מהר!

נספח ב – סיכום

אינטרפולציה

<u>הבעיה</u>: נתונה קבוצה סופית של ערכים של נקודות על גרף של פונקציה $\{(x_i, y_i)\}_{i=1}^M$ כאשר $x_i \in [a,b]$ לכל $1 \le i \le M$. בהינתן $x \in \mathbb{R}$ נרצה לשערך את $f(x)$. אם $x \in [a,b]$ הבעיה נקראת **אינטרפולציה** ואם $x \notin [a,b]$ הבעיה נקראת **אקסטרפולציה**. תמיד נניח שהפונקציה המדוברת היא רציפה.

<u>הגדרה</u>: נתונות $\{(x_i, y_i)\}_{i=1}^M$. **אינטרפולציה ליניארית** $\hat{f}$ היא מהצורה $\hat{f}(x, c_1, \dots, c_M) = \sum_{i=1}^M g_i(x) c_i$ כאשר g_i פונקציות כלשהן. **סדר האינטרפולציה** הוא $M-1$. **אינטרפולציה פולינומיאלית** היא אינטרפולציה ליניארית שבה בוחרים $g_i(x) = x^{i-1}$. נתמקד באינטרפולציה פולינומיאלית.

התנאים $\hat{f}(x_i) = y_i$ באינטרפולציה פולינומיאלית נותנים את מערכת המשוואות הבאה:

$$\begin{pmatrix} 1 & & x_1^{M-1} \\ & \ddots & \\ 1 & & x_M^{M-1} \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_M \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_M \end{pmatrix}$$

זוהי מטריצת ון-דר-מונדה. כמובן, סביר להניח שכל ה-x_i-ים שונות ולכן למערכת המשוואות הזו קיים פיתרון והוא יחיד.

אם נפעל באופן נאיבי לפתרון מערכת המשוואות (למשל ע"י **אלימינציה של גאוס**) אז שלב התפירה נפתר בסיבוכיות $O(M^3)$ ולאחר שכבר יש לנו את המקדמים של הפולינום, כל הערכה לוקחת זמן $O(M)$ ואין צורך לתפור מחדש.

גישה אחרת לפיתרון הבעיה עוקפת את שלב התפירה ע"י שימוש ב**פולינום לגרנז'** שהוא פשוט הפיתרון הידוע מראש של מערכת המשוואות הזאת:

$$p(x) = \sum_{i=1}^{M} y_i \frac{\prod_{j \ne i}(x - x_j)}{\prod_{j=1}^{M}(x_i - x_j)}$$

כאן עלות ההערכה היא $O(M^2)$ אבל אין את שלב התפירה. כמובן, ניתן היה מהנוסחה הזו לחלץ את המקדמים של הפיתרון אבל זה כרוך בפתיחת סוגריים והעלות תהיה $O(M2^M)$.

בשתי השיטות שראינו למעלה יש בעיה של יציבות נומרית. בד"כ לא רצוי לחבר מספרים מסדרי גודל שונים כי הדיוק של החישוב נפגע מאוד בגלל אופן הייצוג של המספרים במחשב בפרט, הבעיה חמורה מאוד כאשר סדר האינטרפולציה גבוה ואז עלול להיות הבדל מאוד גדול בין x ל-x^{M-1}. הרעיון הוא לבנות את הפולינום בשלבים ובכל פעם לעשות חישובים על מספרים בסדרי גודל דומים.

ב**שיטת Neville** בונים סדרת פולינומים $p_{i,\dots,i+m}$ כך שלכל $j = 1, \dots, m$, $p_{i,\dots,i+m}(x_j) = y_i$:

$$p_i(x) = y_i$$
$$p_{i,\dots,i+m}(x) = \frac{(x - x_i)p_{i+1,\dots,i+m}(x) + (x_{i+m} - x)p_{i,\dots,i+m-1}(x)}{x_{i+m} - x_i}$$

לבסוף, הפולינום $p_{1,\dots,M}(x)$ הוא הפולינום שמעניין אותנו.

גם בשיטה הזו מוותרים על שלב התפירה וניגשים מייד להערכה. עלות ההערכה היא $O(M^2)$, כמו בשיטת הפולינום של לגרנז', אלא שהיתרון כאן הוא הדיוק הנומרי.

משפט: נניח שנתונות נקודות $\{(x_i, y_i)\}_{i=0}^{M}$ כאשר $x_{i+1} - x_i = \delta$ על גרף של פונקציה f כך ש-$x_i \in [a,b]$ לכל $0 \le i \le M$ ו-p מתלכדת עם f על נקודות אלה. נניח של-f יש $M+1$ נגזרות בקטע $[a,b]$. יהי p הפולינום ממעלה M שמתלכד עם הפונקציה בנקודות הנתונות. אזי

$$|p(x) - f(x)| \le \frac{1}{M}\delta^{M+1}\left|f^{(M+1)}(\xi)\right| = O(\delta^{M+1})$$

חשוב לזכור שבפועל, בהרבה מקרים נגזרות גבוהות של פונקציות נוטות "להשתגע" ואז החסם שלנו לא מאוד מוצלח. זה לא אומר שהשגיאה תהיה גדולה בפועל אבל הניתוח התיאורטי שלנו לא מאפשר לראות זאת ואין הבטחה שהקירוב יהיה טוב יותר.

כדי להתגבר על הבעיה ניתן לעשות את התפירה לקבוצות קטנות יותר של נקודות. יש שתי אפשרויות:

1. חלוקת הקטע הגדול לתת קטעים זרים התחומים ע"י N נקודות בכל אחד מהם ושימוש באינטרפולציה ממעלה $N-1$ בכל קטע. הפונקציה המתקבל היא כמובן רציפה אבל היא לא בטוח גזירה בנקודות החפיפה.
2. בחירה שרירותית של N נקודות וחישוב האינטרפולציה רק עליהן. אם אנחנו צופים שנרצה לשערך רק בנקודה אחת זו יכולה להיות אפשרות טובה.

על שתי האלטרנטיבות למעלה, ובעיקר על הראשונה, אולי נרצה לכפות גם אילוצים שקשורים לנגזרת. אם יש לנו מידע גם על הנגזרות של הפונקציה בנקודות הדגימה אפשר לשלב אותן במשוואות. במקרה זה נצטרך להשתמש בפולינומים ממעלה N. אם אין לנו מידע מוקדם על הנגזרות נשמש בתוצאות הביניים כדי להסיק מידע על הנגזרות וכך נוכל לכפות על הפתרון שלנו להיות גזיר גם בנקודות הדגימה. כלומר, נפתור את המשוואות עבור הקטע הראשון. זה ייתן לנו פולינום f_1 ממעלה $N-1$ שמתלכד עם N נקודות הדגימה הראשונות. ואז כדי לחשב את f_2 נוסיף לאילוצים שלנו גם את המשוואה $f_1^{'}(x_{N-1}) = f_2^{'}(x_{N-1})$. כך נוכל להמשיך הלאה ולתפור פונקציה לכל הקטע. שיטה זו נקראת **natural splines**.

נניח כעת שנתונות לנו נקודות על גרף הפונקציה במרווחים אחידים ויתר על כן, הנקודות שאותן נרצה לשערך תמיד יחתכו את הקטעים הקטנים באותו מיקום יחסי. במקרה כזה ניתן להוכיח שהאינטרפולציה ניתנת לייצוג כקונבולוציה עם פילטר מסוים.

אפשרות אחרת להתמודד עם בעיית הסדרים הגבוהים היא למצוא פולינום שרק בקירוב מתלכד עם הפונקציה בנקודות הנתונות. אם מחפשים פולינום ממעלה N אשר ממזער את השגיאה הריבועית $E = \sum_{i=1}^{N}(p(x_i) - y_i)^2$ מקבלים את התוצאה $\vec{c} = (A^t A)^{-1} A^t \vec{y}$ שנקראת **pseudo-inverse** של המטריצה A.

נעבור לפונקציות רב-ממדיות. הפעם הקלט שלנו הוא ערכי פונקציה שמוגדרת על $\mathbb{R}^n$. כלומר, יש לנו $\{(\vec{x_i}, y_i)\}_{i=0}^{M}$ כאשר $\vec{x_i} \in \mathbb{R}^n$.

יש כמה אפשרויות שאפשר לנקוט בהן:

1. להרחיב את מה שעשינו במקרה החד-ממדי: לבחור פולינום ממעלה M ב-n משתנים ולנסות לפתור משוואות ולהתאים אותו. הבעיה העיקרית כאן (פרט לקושי החישובי) היא שיש מספר בחירות אפשריות של פולינום כזה ממעלה M.
2. לטפל בכל ממד בנפרד: אם הדגימות נתונות על סריג אחיד ניתן ראשית להתאים פולינומים לממד הראשון, אז להשתמש בתוצאה כדי לתפור את הממד השני וכך הלאה.
3. מתן השפעה גדולה יותר לנקודות שקרובות לנקודות שאנחנו רוצים לשערך...

Radial Basis Function היא פונקציה ממשית שמוגדרת על $\mathbb{R}^n$ ותלויה רק במרחק של הנקודה מראשית הצירים, או מנקודה אחרת. באופן כללי ניתן לומר שפונקציה כזאת היא מהצורה $\phi(x,c) = \phi(\|x-c\|)$.
בפונקציות כאלה אפשר להשתמש כדי לקרב את $\{(x_i,y_i)\}_{i=0}^{M}$ ע"י

$$f(x) = \sum_{i=0}^{M} w_i \phi(\|x - x_i\|)$$

אם $\phi(r) \xrightarrow[r\to\infty]{} 0$ אכן ככל שנקודה קרובה יותר לנקודת השיערוך ההשפעה שלה גדולה יותר. כדי לקבל את המקדמים $\{w_i\}_{i=0}^{M}$ ניתן לפתור מערכת משוואות כמו קודם, כאשר הדרישה היא שלכל $i = 0, \dots, M$ מתקיים $f(x_i) = y_i$.

לחילופין, **שיטת Shepard** אינה דורשת פתרון משוואות ופשוט מחליטים לקרב את הפונקציה ע"י

$$f(x) = \frac{\sum_{i=0}^{M} y_i\, \phi(\|x - x_i\|)}{\sum_{i=0}^{M} \phi(\|x - x_i\|)}$$

זוהי נוסחה מפורשת של השערוך ולכן כדי לשערך נקודה יש צורך ב-$O(M)$ פעולות וזהו. כל נקודת שיערוך חדשה היא צירוף קמור של $\{y_i\}_{i=0}^{M}$ ולכן $\min_i y_i \le f(x) \le \max_i y_i$.

באופן דומה ניתן לקרב פונקציות צפיפות של נקודות תצפית. נניח שיש אוסף תצפיות $\{x_i\}_{i=0}^{M} \subseteq \mathbb{R}^n$ ואנחנו מעוניינים לקרב את פונקצית הצפיפות שממנה נדגמו התצפיות, כלומר, נרצה לשערך את הסבירות לקבל דגימה בנקודה מסוימת $x \in \mathbb{R}^n$. נהוג לקרב את ההסתברות ע"י $\Phi(x) = \sum_{i=0}^{M} \phi(\|x - x_i\|)$. כדי לקבל פונקצית הסתברות ממש צריך גם לנרמל אבל אם רוצים רק להשוות בין נקודות שונות, זה מספיק. שיטה זו נקראת **חלונות Parzen**.

אינטגרציה

<u>הבעיה</u>: נתונה פונקציה $f:\mathbb{R}\to\mathbb{R}$ ורוצים להעריך את $\int_a^b f(x)\,dx$.

שלבים בפתרון:

1. חלוקת תחום האינטגרציה למקטעים קטעים באורך h
2. קירוב הפונקציה f בכל קטע ע"י פונקציה פשוטה שאנחנו יודעים את האינטגרל
3. סיכום כל האינטגרלים במקטעים הקטנים

<u>משפט</u>: תהי $f:\mathbb{R}\to\mathbb{R}$ ונניח שנחשב את האינטגרל $\int_a^b f(x)\,dx$ ע"י קירוב הפונקציה במקטעים באורך h ע"י פולינומים מסדר N. אזי איכות קירוב האינטגרל היא $O(h^{N+1})$.

ב**שיטת הטרפז** נבצע קירוב מסדר ראשון בין כל זוג נקודות עוקבות ונחשב את השטח באמצעות נוסחת שטח של טרפז. אחרי סיכום כל הקטעים נקבל

$$\int_a^b f(x)\,dx \approx \frac{b-a}{n}\left(\frac{f(a)+f(b)}{2} + \sum_{i=1}^{n-1} f\left(a + k\frac{b-a}{n}\right)\right)$$

מהמשפט נובע שאיכות הקירוב של שיטת הטרפז היא $O(h^2)$.

שיטת סימפסון מקרבת באמצעות קירובים ממעלה שנייה. מהמשפט הקודם נובע שאיכות הקירוב היא $O(h^3)$. אבל ניתוח קפדני יותר יכול להראות שלמעשה החסם טוב יותר והוא $O(h^4)$.

עבור פונקציות רב-ממדיות, כמו באינטרפולציה רב-ממדית, יש שתי אפשרויות שניתן לנקוט בהן:

1. להתאים פולינום רב-ממדי ולחשב את האינטגרל ממנו.
2. לחשב את האינטגרל בשלבים לכל ממד בנפרד. הצידוק המתמטי של השיטה הזו נובע ממשפט פוביני:
$$\int_{A\times B} f(x,y)\,d(x,y)=\int_A\left(\int_B f(x,y)\,dy\right)dx$$
גם במקרה זה מתקבלת שגיאה $O(h^{N+1})$ כאשר N סדר הקירוב.

לחילופין, ניתן לנקוט בשיטות סטטיסטיות. נניח ש-g פונקצית התפלגות ו-$x_i \sim g(x)$ בלתי תלויים. אזי

$$I=\int_a^b f(x)\,dx=\int_a^b \frac{f(x)}{g(x)}g(x)\,dx\approx \frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)}{g(x_i)}$$

השגיאה שמתקבלת היא משתנה אקראי שהתוחלת שלו היא $E=\frac{1}{\sqrt{N}}\mathrm{std}\frac{f}{g}$. כש-$\frac{f}{g}$ לא משתנה הרבה הקבוע $\mathrm{std}\frac{f}{g}$ הוא נמוך ומתקבלת שגיאה נמוכה. נשים לב שכתלות ב-N, השיטה מתכנסת תמיד! אם בחרנו פונקציה g לא מוצלחת יכול להיות ש-$\mathrm{std}\frac{f}{g}$ יהיה גדול מאוד, אך הוא קבוע ואינו תלוי ב-N. ולכן קצב ההתכנסות של אינטגרציה מונטה-קרלו הוא תמיד $O\left(\frac{1}{\sqrt{N}}\right)$ ללא תלות בפונקציה f שרוצים לחשב את האינטגרל שלה וללא תלות בפונקציה g שנבחרה.

לפעמים הבעיה באינטגרציה היא לא שאנחנו לא יודעים לעשות אינטגרל לפונקציה אלא שתחום האינטגרציה הוא מסובך ואין פרמטריזציה נוחה לעבוד איתה. נניח למשל ש-$D\subseteq U$ כאשר U תחום פשוט שאנחנו יכולים לחשב את האינטגרל עליו. כעת אפשר להשתמש בשיטת מונטה-קרלו:

$$\int_U f(x)1_D(x)\,dx=\int_{\mathbb{R}^d} f(x)1_D(x)\frac{1_U(x)}{|U|}|U|\,dx\approx\frac{1}{N}\sum_{i=1}^{N}\frac{f(x_i)1_D(x_i)}{\frac{1_U(x_i)}{|U|}}=\frac{|U|}{N}\sum_{i=1}^{N}f(x_i)1_D(x_i)$$

כאשר $x_i\sim\frac{1_U(x)}{|U|}$. אז אפשר להשתמש בשיטת מונטה-קרלו אלא שלפני שמכניסים נקודה לסכום בודקים אם היא בתחום האינטגרציה שלנו או לא.

מערכות משוואות לא ליניאריות

<u>הבעיה</u>: נתונה פונקציה $f:\mathbb{R}^n\to\mathbb{R}$ ורוצים למצוא $x\in\mathbb{R}^n$ כך ש-$f(x)=0$.

שיטת החצייה מתבססת על משפט ערך הביניים עבור פונקציות רציפות. אם הפונקציה $f:\mathbb{R}\to\mathbb{R}$ רציפה ואנחנו יודעים שתי נקודות $a<b$ כך ש-$f(a)f(b)<0$ (כלומר סימן הפונקציה עליהן שונה) אז לפי משפט ערך הביניים קיימת נקודה $a<c<b$ כך ש-$f(c)=0$. אז אפשר פשוט לעשות חיפוש בינארי. זאת שיטה איטרטיבית והיא משפרת את הקירוב בכל איטרציה. בכל שלב קטע העניין קטן פי שניים ולכן השיטה מתכנסת בקצב גיאומטרי עם קבוע $\frac{1}{2}$. הצרה בשיטה הזו היא שלא ניתן להרחיב אותה באופן טבעי לפונקציות רב ממדיות.

שיטת ניוטון-רפסון עובדת לפונקציות גזירות וניתן להרחיב אותה באופן טבעי גם לרב-ממד. השיטה לא תמיד מתכנסת אבל כשהיא מתכנסת זה קורה מאוד מהר. השיטה מתבססת על קירובים ליניאריים של הפונקציה. בכל שלב מקרבים את הפונקציה ע"י המשיק שלה בנקודה. זוהי פונקציה ליניארית וניתן למצוא את השורש שלה. אז מצופה שהשורש של המשיק יהיה קרוב לשורש הפונקציה. משם ממשיכים הלאה:

NewtonRaphson(f, f', x_0)

1. $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$
2. $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

משפט: תהי $f: \mathbb{R} \to \mathbb{R}$ פונקציה גזירה ברציפות פעמיים ונניח ש-$f(r) = 0$. נסמן ב-$e_n = x_n - r$ את השגיאה המתקבלת באיטרציה n של שיטת ניוטון-רפסון. אזי לכל $n \in \mathbb{N}$ קיימת ξ_n כך ש-

$$e_{n+1} = \frac{e_n^2}{2} \frac{f''(\xi_n)}{f'(x_n)}$$

בפרט, אם $\frac{e_n}{2} \frac{f''(\xi_n)}{f'(x_n)} < 1$ לכל $n \in \mathbb{N}$ אז התהליך מתכנס. אם הנגזרות הראשונה והשנייה חסומות והניחוש הראשוני מספיק טוב אז התהליך מתכנס בקצב סופר-ריבועי.

היתרון המשמעותי של השיטה הזו הוא שאפשר להרחיב אותה באופן פשוט לפונקציות רב-ממדיות. אז כלל ההתקדמות פשוט נתון ע"י

$$x_{n+1} = x_n - \left(\nabla f(x_n)\right)^{-1} f(x_n)$$

כאשר

$$\nabla f = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & & \frac{\partial f_1}{\partial x_N} \\ & \ddots & \\ \frac{\partial f_N}{\partial x_1} & & \frac{\partial f_N}{\partial x_N} \end{pmatrix}$$

כמו שמקרה החד-ממדי היינו צריכים ש-$f'(x_n) \neq 0$, כאן כדי שהאלגוריתם יוכל להמשיך צריך להתקיים ש-$\nabla f(x_n)$ הפיכה. ניתן להראות שבתנאים מסוימים גם השיטה הזאת מתכנסת.

מערכות משוואות ליניאריות

הבעיה: נתונה מטריצה ריבועית הפיכה $A \in M_n(\mathbb{R})$ ונתון $b \in \mathbb{R}^n$ ורוצים למצוא $x \in \mathbb{R}^n$ כך ש-$Ax = b$.

יש שני סוגי שיטות לפתרון הבעיה:

- **שיטות ישירות**: מבצעות מספר סופי של פעולות חשבוניות שבסופן מתקבל פתרון מדויק של המשוואה (עד כדי שגיאות round off).
- **שיטות איטרטיביות** מתקרבות לפיתרון בשלבים אבל בשום נקודת זמן סופית (פרט למקרים בודדים) אין פיתרון מדויק.

אלימינציה של גאוס

השיטה הישירה הכי פשוטה היא אלימינציה של גאוס. מפעילים אוסף של פעולות אלמנטריות על המטריצה $[A, b]$ כאשר כל פעולה משמרת את מרחב הפתרונות עד ש-A הופכת למטריצת הזהות. השיטה עובדת ב-$O(n^3)$. אפשר לבצע זאת לכל וקטור b מחדש או לחילופין ניתן לחשב פעם אחת את ההפכית של A ואז להסתפק בכפל מטריצות כדי לפתור משוואות חדשות.

פירוק LU

אם נצליח לפרק את A למכפלה $A = LU$ כאשר L משולשית תחתונה ו-U משולשית עליונה מערכת המשוואות $Ax = b$ תהפוך ל-$L(Ux) = b$. אבל L משולשית תחתונה ולכן ע"י הצבה לאחור (back substitution) נוכל לפתור את המשוואה $Ly = b$ בקלות. ואז ע"י הצבה לאחור נוכל לפתור בקלות את המשוואה $Ux = y$ וזה הרי בדיוק מה שרצינו. בהינתן L, U החישוב הזה לוקח $O(n^2)$.

<u>הגדרה</u>: תת מטריצה עיקרית של מטריצה נתונה A היא תת מטריצה שהאלכסון שלה הוא חלק מהאלכסון הראשי של A.

<u>משפט</u>: תהי A מטריצה כך שכל תת מטריצה עיקרית שלה מהצורה $\begin{pmatrix} A_{11} & \cdots & A_{1k} \\ \vdots & \ddots & \vdots \\ A_{nk} & \cdots & A_{kk} \end{pmatrix}$ עבור $k \leq n$

היא הפיכה. אזי קיים ל-A פירוק LU.

<u>משפט</u>: תהי A מטריצה הפיכה. אזי ל-A קיים פירוק LU אמ"מ כל תת מטריצה עיקרית שלה היא הפיכה.

אם קיים פירוק ניתן לחשב אותו בזמן $O(n^3)$ מה שלא נראה כשיפור משמעותי לעומת אלינימציה של גאוס אבל ספירה קפדנית של הפעולות מראה שיש הבדל בקבועים והשיטה הזאת משתמשת בכשני שליש מהפעולות של שיטת גאוס.

<u>פירוק Cholesky</u>

<u>משפט</u>: תהי $A \in M_n(\mathbb{R})$ מטריצה סימטרית. אזי A מוגדרת חיובית אמ"מ קיימת מטריצה משולשית תחתונה $L \in M_n(\mathbb{R})$ כך ש-$A = LL^t$ וערכי האלכסון של L חיוביים ממש.

פירוק זה נקרא **פירוק Cholseky** והוא יחיד. למעשה גם למטריצה סימטרית מוגדרת אי-שלילית קיים פירוק כזה אלא שאז ערכי האלכסון של L יכולים להיות גם אפס והפירוק לא יחיד בהכרח.

כמובן, מדובר כאן במקרה פרטי של פירוק LU ולכן בדיוק כמו שתואר קודם, ניתן לפתור מערכת משוואות כאשר המטריצה סימטריות ומוגדרת אי-שלילית באמצעות פירוק Cholesky. גם הפירוק הזה לוקח $O(n^3)$.

<u>מטריצות פירוק</u>

נניח שמערכת המשוואות שלנו היא $Ax = b$ ונניח ש-Q מטריצה הפיכה כלשהי. אזי מתקיים

$$Ax = b \Leftrightarrow x = Q^{-1}(Q - A)x + Q^{-1}b$$

את הביטוי הזה אפשר להפוך לשיטה איטרטיבית. נתחיל מניחוש ראשוני כלשהו x^0 ונתקדם לפי הכלל

$$x^{n+1} = Q^{-1}(Q - A)x^n + Q^{-1}b$$

במקרה הכללי כל איטרציה כרוכה במכפלת מטריצות ולכן לוקחת $O(N^2)$. אבל אם המטריצות דלילות (כלומר, בכל שורה יש $O(1)$ איברים שאינם אפס) אז המכפלה לוקחת $O(N)$ ואז אם מספר האיטרציות עד ההתכנסות קטן מ-N אנחנו במצב טוב ושיפרנו את הביצועים לעומת השיטות הישירות.

מטריצה Q כזאת נקראת **מטריצת פירוק**. השגיאה עבור התהליך הנ"ל מקיימת

$$\|e^{n+1}\|_k = \|(I - Q^{-1}A)e^n\|_k \leq \|I - Q^{-1}A\|_k^n \|e^0\|_k$$

ולכן כדי שהתהליך יתכנס מספיק שיתקיים $\|I - Q^{-1}A\|_k < 1$. יתר על כן, מספיק שזה יהיה נכון לנורמת k כלשהי משום שידוע שכל הנורמות שקולות.

בשיטת Jacobi בוחרים את $Q = \text{diag}\, A$. נאמר שמטריצה A היא **דומיננטית אלכסונית** אם לכל $1 \le i \le N$ מתקיים $|A_{ii}| \ge \sum_{j \ne i} |A_{ij}|$.

<u>משפט</u>: שיטת Jacobi מתכנסת עבור מטריצות דומיננטיות אלכסונית ולכל ניחוש ראשוני בקצב $||I - Q^{-1}A||_\infty$.

בשיטת Gauss-Seidel בוחרים $Q = \text{lower}\, A$ – החלק המשולשי התחתון של המטריצה.

<u>משפט</u>: שיטת Gauss-Seidel מתכנסת עבור מטריצות דומיננטיות אלכסונית ולכל ניחוש ראשוני בקצב $||I - Q^{-1}A||_\infty$.

היתרון העיקרי של השיטה הזו הוא שהיא ניתנת למימוש in-place וללא הקצאת זיכרון נוסף. בשיטת Jacobi צריך תמיד להחזיק את x^n ואת x^{n+1}. יתר על כן, שיטת Gauss-Seidel מתכנסת מעט מהר יותר. מאידך, יש מקרים שבהם Jacobi מתכנסת ואילו Gauss-Seidel אינה מתכנסת. בכל מקרה, עבור מטריצות דומיננטיות אלכסונית, שתי השיטות מתכנסות לכל וקטור b ולכל ניחוש ראשוני x^0.

אמרנו שכדי שהתהליך יתכנס צריך שתהיה נורמה שעבורה מתקיים

$$||I - Q^{-1}A||_k < 1$$

כמובן, ניתן לכפול את Q בכל קבוע חיובי ולכן אפשר גם להסתכל על התנאי

$$||I - \alpha Q^{-1}A|| < 1$$

אם A, Q מוגדרות חיובית בחירה אופטימלית היא $\alpha = \frac{2}{\lambda_{\max} + \lambda_{\min}}$. קצב ההתכנסות שמתקבל במקרה זה הוא $\frac{\kappa(A)-1}{\kappa(A)+1}$ כאשר $\kappa(A) = \frac{\lambda_{\max}}{\lambda_{\min}}$.

בפועל בוחרים בד"כ $\alpha = \frac{2}{\lambda_{\max}}$ גם כי זה אומר שצריך לחשב פחות וגם כי הרבה פעמים $\lambda_{\min}$ קרוב מאוד לאפס ולכן זניח.

<u>Steepest Descent</u>

נניח כעת שמערכת המשוואות נתונה ע"י $Ax = b$ כאשר A סימטרית ומוגדרת חיובית. במקרה כזה ניתן להראות שהפונקציה $f(x) = \frac{x^t A x}{2} - x^t b$ מקבלת מינימום בדיוק כאשר $Ax = b$. לכן, כדי לפתור את מערכת המשוואות אנחנו נרצה למזער את הפונקציה $f(x)$. את זה נעשה בשיטה איטרטיבית. הרעיון הוא שבכל שלב נבחר כיוון התקדמות כלשהו d^n ונמזער את הפונקציה לאורך כיוון זה.

<u>SteepestDescent(A, b, x^0)</u>

2. until convergence do
   a. $r^n = b - Ax^n$
   b. $\alpha = \frac{(r^n)^t r^n}{(r^n)^t A r^n}$
   c. $x^{n+1} = x^n + \alpha r^n$

תכונות של האלגוריתם:

1. כל איטרציה כרוכה בביצוע פעולות על מטריצות ולכן לוקחת $O(N)$ אם המטריצות דלילות ו-$O(N^2)$ במקרה הכללי. אם מספר האיטרציות בלתי תלוי ב-N אז זו גם הסיבוכיות של כל האלגוריתם.

2. כיווני החיפוש r^n מקיימים שתי תכונות:

$$r^n \perp r^{n+1}$$
$$r^n \perp_A e^{n+1}$$

3. קצב ההתכנסות הוא $\omega_{SD} = \frac{\kappa(A)-1}{\kappa(A)+1}$ כאשר $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \geq 1$ מספר המצב של A. כלומר, $\|e^{n+1}\|_A \leq \|e^n\|_A \omega_{SD}$. כאשר מספר המצב נמוך קצב ההתכנסות גבוה. בהמשך נראה שיטות שבאמצעותן נגרום להורדת מספר המצב של מטריצה.

Conjugate Gradients

נפעל בצורה דומה ל-SD אלא שנבחר את וקטורי הכיוון בצורה קצת שונה.

ConjugateGradients(A, b, x^0)

3. $d^0 = r^0 = b - Ax^0$
4. until convergence do
   a. $\alpha = \frac{(d^n)^t r^n}{(d^n)^t A d^n}$
   b. $x^{n+1} = x^n + \alpha d^n$
   c. $r^{n+1} = r^n - \alpha A d^n$
   d. $\beta = \frac{(r^{n+1})^t r^{n+1}}{(r^n)^t r^n}$
   e. $d^{n+1} = r^{n+1} + \beta d^n$

באופן הזה כל איטרציה כרוכה בכמה פעולות כפל מטריצה. אם המטריצות דלילות זה לוקח $O(N)$ ואם הן מלאות זה לוקח $O(N^2)$. אם מספר האיטרציות שאנחנו בוחרים לעשות קבוע ולא תלוי בגודל המטריצה זו הסיבוכיות של כל האלגוריתם וחסכנו הרבה!

תכונות האלגוריתם:

1. כיווני החיפוש d^n יוצרים קבוצה A-אורתונורמלית ולכל n מתקיים $e^{n+1} \perp_A \{d^i\}_{i=0}^n$. לכן, אחרי לכל היותר N צעדים השיטה תתכנס ונקבל תוצאה מדויקת.
2. סדר ההתכנסות של האלגוריתם הוא

$$\|e^{n+1}\|_A \leq 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^n \|e^0\|_A$$

Preconditioning

יש בעיות שבהן מספר המצב $\kappa(A)$ מאוד גבוה ואולי אפילו תלוי בממד המטריצה. במקרה כזה השיטות שתיארנו אמנם מתכנסות אך לאט מאוד. היינו רוצים למצוא מערכת משוואות שקולה עם מספר מצב נמוך יותר.

יש שני סוגי preconditioning שהם למעשה שקולים. אם $Ax = b$ ו-Q הפיכה אז $Q^{-1}Ax = Q^{-1}b$. ולשתי המשוואות יש בדיוק אותם פתרונות. אז נרצה למצוא Q כזאת שהופכת את מערכת המשוואות לפשוטה ובעלת מספר מצב נמוך. לחילופין, ניתן להתבונן במערכת המשוואות $E^tAE\hat{x} = E^tb$. במקרה זה $E\hat{x}$ הוא פתרון של מערכת המשוואות המקורית. אלא שהפעם E^tAE סימטרית ומוגדרת חיובית (ולכן ניתן להשתמש בשיטות הקודמות שלמדנו) והתקווה שהיא שמספר המצב שלה נמוך יותר. המטריצות הנ"ל מקרות **מטריצות בידוד**.

בגדול מציאת מטריצת בידוד מתבצעת בשני שלבים:

1. **בידוד** – מציאת הצירים V (בקירוב) ש-A מכווצת או מותחת ובחירת $D = \text{diag}(V^tAV)$
2. **תיקון** – תיקון של המטריצה בצירים אלה, כלומר מתיחת הצירים שכווצו ולהפך $\left(V\sqrt{D^{-1}}\right)^t AV\sqrt{D^{-1}}$

אם V^tAV הייתה אלכסונית אז המצב היה אידיאלי. אם החישוב הוא מקורב חשוב שהקירוב יהיה טוב כי אם נבחר את מערכת הצירים לא טוב יכול להיות שלמעשה לא נשפר בכלל את המצב.

נגדיר את האנרגיה של וקטור x להיות $E(x) = x^tAx = \|x\|_A^2$. נשים לב שאם v וקטור עצמי מנורמל (לפי נורמה סטנדרטית) של A אז $E(v) = v^tAv = v^t\lambda v = \lambda v^tv = \lambda$.לכן, אם אנחנו בוחרים את כל הוקטורים להיות עם אנרגיה דומה זוהי בחירה גרועה של מערכת צירים. אנחנו מחפשים מערכת צירים חדשה שיש לה אנרגיות מעורבות – גם גבוהות וגם נמוכות ואז בשלב התיקון באמת יש משמעות למתיחה ולכיווץ. אם אנחנו מכירים איזו תכונה שמשפיעה על האנרגיה אפשר לנסות להשתמש בה כמדריך לבחירת כיוונים.

מאחר ש-E^tAE היא סימטרית ומוגדרת חיובית ניתן לפתור את המשוואה באמצעות CG. אם נציב את הנתונים האלה באלגוריתם נקבל את האלגוריתם שנקרא **Transformed Preconditioned CG**:

<u>TransformedPCG($A, b, \hat{x}^0, E$)</u>

1. $\hat{d}^0 = \hat{r}^0 = E^tb - E^tAE\hat{x}^0$
2. until convergence do
   a. $\alpha = \frac{(\hat{d}^n)^t\hat{r}^n}{(\hat{d}^n)^tE^tAE\hat{d}^n}$
   b. $\hat{x}^{n+1} = \hat{x}^n + \alpha\hat{d}^n$
   c. $\hat{r}^{n+1} = \hat{r}^n - \alpha E^tAE\hat{d}^n$
   d. $\beta = \frac{(\hat{r}^{n+1})^t\hat{r}^{n+1}}{(\hat{r}^n)^t\hat{r}^n}$
   e. $\hat{d}^{n+1} = \hat{r}^n + \beta\hat{d}^n$

בגרסה הקודמת של האלגוריתם יש לחשב את E מתוך Q וזה יכול לקחת הרבה זמן. ע"י כמה חילופי משתנים

$$\hat{r} = E^tr, \hat{x} = E^{-1}x, \hat{d} = E^{-1}d$$

ניתן לקבל את הגרסה **Untransformed Preconditioned CG** שמחזירה אותנו למשתנים המקוריים ונותנת לנו מיד את הפתרון x של המשוואה המקורית (כזכור, $x = E\hat{x}$):

<u>UntransformedPCG()</u>

1. $r^0 = b - Ax^0$
2. $d^0 = EE^tr^0 = Q^{-1}r^0$
3. until convergence do
   a. $\alpha = \frac{(d^n)^tQ^{-1}r^n}{(d^n)^tAd^n}$
   b. $x^{n+1} = x^n + \alpha d^n$
   c. $r^{n+1} = r^n - \alpha Ad^n$
   d. $\beta = \frac{(\hat{r}^{n+1})^tQ^{-1}\hat{r}^{n+1}}{(\hat{r}^n)^tQ^{-1}\hat{r}^n}$
   e. $d^{n+1} = Q^{-1}r^n + \beta d^n$

זה משפר את המצב משום שכעת לא צריך לפרק את Q ל-EE^t ומספיק רק לדעת שקיים פירוק כזה. לאחר מכן ניתן פשוט להריץ את האלגוריתם הזה ומיד לקבל את הפתרון של המשוואה המקורית!

ערכים ווקטורים עצמיים

<u>**הבעיה**</u>: נתונה מטריצה ריבועית $A \in M_N(\mathbb{R})$ ואנחנו רוצים למצוא את כל הוקטורים והערכים העצמיים שלה.

השיטה הישירה היא לפתור את הפולינום $\det(A - xI) = 0$ אבל כשמדובר בממדים גבוהים זה פרקטית בלתי אפשרי.

שיטת החזקות מאפשרת למצוא את הערך העצמי המקסימלי בערך מוחלט עבור מטריצות לכסינות. השיטה מתחילה מניחוש התחלתי אקראי.

<u>PowerMethod(A, u^0)</u>

1. Until convergence do
   a. $u^{n+1} = Au^n$
   b. $\lambda^{n+1} = \|u^{n+1}\|$
   c. $u^{n+1} = \frac{u^{n+1}}{\|u^{n+1}\|}$

אחרי שהאלגוריתם מתכנס u^n הוא וקטור עצמי עם ערך עצמי λ^n, שהרי אם התהליך התכנס אז $u^{n+1} = Au^n = u^n$ לפני הנירמול, ואחרי שבוחרים $\lambda^n = \|u^n\|$ ומנרמלים את u^n בדיוק מתקיים מה שצריך. ההתכנסות היא גיאומטרית בקצב $\left|\frac{\lambda_2}{\lambda_1}\right|$ (כאשר λ_1, λ_2 שני הערכים העצמיים המקסימליים בערך מוחלט). אז אם שני הערכים העצמיים המקסימליים קרובים זה לזה השיטה מתכנסת לאט מאוד.

ניתן לשדרג קצת את השיטה ולאפשר למצוא כמה ערכים עצמיים מרביים.

<u>m-PowerMethod$(A, u_1^0, \dots, u_k^0)$</u>

1. for $k = 1, \dots, m$
   a. until convergence do
      i. $u_k^{n+1} = Au_k^n$
      ii. $u_k^{n+1} = u_k^{n+1} - \sum_{j=1}^{k-1} u_j \langle u_j, u_k^{n+1} \rangle$
      iii. $u_k^{n+1} = \frac{u_k^{n+1}}{\|u_k^{n+1}\|}$
   b. $u_k = u_k^n$

הרעיון הוא שאחרי שבכל פעם נוריד מהווקטור המשוער את על הרכיבים בכיוונים של הווקטורים העצמיים שכבר נמצאו. ככה המקדמים שלהם ייעלמו מהפיתוח ונישאר רק עם הערך המקסימלי מבין הרכיבים שנותרו.

שינוי קל בפרמטרים של השיטה מאפשר לנו למצוא ערכים עצמיים מינימליים. אם נניח בנוסף ש-A מוגדרת חיובית, כלומר כל הערכים העצמיים שלה חיוביים ממש אז שיטת החזקות מוצאת את הערך העצמי המקסימלי ממש ולא רק מקסימלי בערכו המוחלט. לכן אם נפעיל את השיטה על $\lambda_{\max} \quad I - A$ נקבל את הערך העצמי המינימלי.

אם נפעיל את שיטת החזקות על המטריצה $(A - \mu I)^{-1}$ נקבל את הערך העצמי המקסימלי של $(A - \mu I)^{-1}$. אבל הערכים העצמיים של מטריצה זו הם בדיוק $\frac{1}{\lambda_i - \mu}$ אז המקסימום מתקבל כאשר λ_i קרוב ל-μ. זה נותן את **שיטת החזקות ההפוכה**. נניח ש-λ_1, λ_2 הם שני הערכים העצמיים הקרובים ביותר ל-μ.

אז קצב ההתכנסות הוא $\frac{\lambda_2-\mu}{\lambda_1-\mu}$. ע"י בחירת μ כמה שיותר קרוב ל-λ_1 ניתן לקבל קצב התכנסות יורת ויותר גבוה. כמובן, אילו היינו יודעים את λ_1 לא הייתה לנו בעיה לפתור בכלל אבל בכל מקרה יש כל מיני כללי אצבע שאומרים איך לבחור את μ בצורה טובה.

אז קצב ההתכנסות יכול להיות מאוד גבוה אבל המחיר הוא שהשיטה כרוכה בהיפוך מטריצה או פיתרון מערכת משוואות בכל איטרציה. זה יכול לקחת הרבה זמן.

עד כה עבדנו על מטריצות כלליות. אם נניח שהמטריצה A סימטרית נוכל לפתח שיטה שמוצאת את כל הערכים העצמיים והרבה יותר יציבה נומרית מהקודמות. **Jacobi Iteration** מחפשת מטריצות סיבוב אשר משמרות את הערכים העצמיים של A אך הצמדה בהן מאפסת איברים שמחוץ לאלכסון הראשי.
בסוף התהליך האלכסון מייצג קירוב של הערכים העצמיים ומכפלת כל מטריצות הסיבוב נותנת קירוב של הוקטורים העצמיים המתאימים.

<u>JacobiIteration()</u>

1. $V = I$
2. while $A_{ij} \neq 0, i \neq j$
   a. find R_θ^{ij} such that $\left[R_\theta^{ij} A \left(R_\theta^{ij}\right)^t\right]_{ij} = 0$
   b. $V = R_\theta^{ij} V$
   c. $A = R_\theta^{ij} A \left(R_\theta^{ij}\right)^t$
3. output $\operatorname{diag} A, V$

כל איטרציה באלגוריתם לוקחת $O(N)$ כי מכפלה במטריצת סיבוב משנה רק $O(N)$ מאיברים. אבל כדי שהשיטה תהיה יעילה כדאי לבחור בכל שלב A_{ij} מקסימלי מחוץ לאלכסון וזה כרוך בבדיקה של $O(N^2)$ איברים במטריצה. ויש $O(N^2)$ איטרציות. אז אם אנחנו בוחרים A_{ij} כלשהו השיטה יוצאת $O(N^3)$ אבל אם מחפשים את המקסימום זה אפילו $O(N^4)$.

שיטת ארנולדי היא שיטה שמוצאת ערכים עצמיים של מטריצות כלליות (לא בהכרח סימטריות). השיטה מייצרת שתי מטריצות – מטריצה ריבועית H ומטריצה מלבנית Q כך ש-$H = Q^tAQ$. הערכים העצמיים של H מקרבים את הערכים העצמיים של A ו-Q מייצגת את הוקטורים העצמיים המתאימים. מאחר ש-H קטנה הרבה יותר מ-A (למעשה היא בגודל $m \times m$ כאשר m מספר הערכים העצמיים שרוצים לחשב) ניתן למצוא את כל הערכים העצמיים שלה. עבור מטריצות מלאות הסיבוכית היא $O(m^2N^2)$ ועבור דלילות היא $O(mN^2)$. סיבוכיות הזיכרון היא בכל מקרה $O(mN)$. ולבסוף, צריך להוסיף את הזמן שלוקח לחשב את הערכים העצמיים של H.

כאשר המטריצה סימטרית החישובים בשיטת ארנולדי הופכים קצת פשוטים יותר. במקרה זה ניתן לחשב ולהראות שהמטריצה H שמתקבלת היא מתאפסת פרט לאלכסון הראשי והשניים המשניים ויתר על כן, היא סימטרית. ולכן באלגוריתם החישוב של H לוקח פחות זמן. כך כל השיטה יורדת ל-$O(mN)$ למטריצות דלילות ו-$O(mN^2)$ למטריצות מלאות. שיטת ארנולדי עבור מטריצות סימטריות נקראת **שיטת Lanczos**.

בהינתן מטריצה A, אילו יכולנו לפרק אותה ל-$A = QR$ כאשר Q מטריצת סיבוב ו-R מטריצה משולשית עליונה יכולנו להשתמש בפירוק הזה כדי לחשב ערכים עצמיים של A:

$$A_0 = A$$
$$A_{k+1} = R_kQ_k = Q_k^tQ_kR_kQ_k = Q_k^tA_kQ_k$$

ולכן כל המטריצות A_k דומות ויש להן ערכים עצמיים זהים. תחת תנאים מסוימים הסדרה A_k מתכנסת למטריצה משולשית ואז בעיית מציאת הערכים העצמיים נפתרת.

נותר להראות כיצד לחשב **פירוק QR**. יש שלוש דרכים לבצע זאת.

- פירוק QR באמצעות תהליך **גראם-שמידט**: נבצע תהליך גראם-שמידט על העמודות של המטריצה A ונקבל את הווקטורים $q_{*1},\dots,q_{*n}$. כמובן המטריצה שנוצרת ע"י וקטורים אלה היא מטריצת סיבוב. ע"י כמה מניפולציה אלגבריות ניתן להראות שהמטריצה $R=Q^tA$ היא משולשית עליונה. ומאחר ש-Q מטריצת סיבוב אכן מתקיים $A=QQ^tA=QR$.
היתרון בשיטה הזו הוא שהוא לא מסתמך על אף תכונה של המטריצה המקורית. את התהליך הזה ניתן לבצע לכל מטריצה ולקבל פירוק כמו שרצינו. ועלות הפירוק היא למעשה עלות החישוב של תהליך גראם-שמידט שזה $O(n^3)$.
- פירוק QR באמצעות **Householder Reflection**: ניתן להגדיר סדרה של מטריצות Q_n כך ש-$Q_n\cdot\ldots\cdot Q_1A=R$ מטריצה משולשית עליונה ואז אם ניקח $Q=Q_1^t\dots Q_n^t$ נסיים. העלות החישובית של השיטה נובעת בעיקר מהכפל $Q_n\dots Q_1A$. אבל Q_i מוגדרות כך שהכפל בהן מאוד פשוט ולוקח $O(n^2)$ במקום $O(n^3)$. כל שלב באלגוריתם לוקח $O(n^2)$ ולבסוף כדי לחשב את Q צריך לכפול n כאלה מה שמביא אותנו ל-$O(n^3)$.
- פירוק QR באמצעות **מטריצות Givens**: מטריצת Givens היא מטריצת סיבוב כמו שהשתמשנו בשיטת Jacobi iteration. ניתן לעשות תהליך דומה לאלימינציה של גאוס שבו בכל איטרציה מאפסת איבר מתחת לאלכסון הראשי של A. אז מכפלת כל מטריצות הסיבוב שהשתמשנו בהן נותנת את Q וההפעלה שלהן על A נותנת את R. הסיבוכיות המתקבלת היא $O(n^3)$.

אופטימיזציה

<u>הבעיה</u>: נתונה פונקציה רציפה $f:\mathbb{R}^d\to\mathbb{R}$ ורוצים למצוא נקודת מינימום מקומית x.

כאשר הפונקציה מוגדרת על $\mathbb{R}$, כלומר הבעיה חד-ממדית, ניתן לפעול במעין חיפוש. נתחיל עם שלשה $a<b<c$ כך ש-$f(b)<f(a),f(c)$ ונבחר $x\in(a,b)$ כלשהו. יש שתי אפשרויות:

- אם $f(x)<f(b)$ נמשיך באותו אופן עם השלשה $a<x<b$
- אם $f(x)\geq f(b)$ נמשיך באותו אופן עם השלשה $x<b<c$

כך בכל שלב אנחנו נמצאים באותו מצב התחלתי שבו יש שלוש נקודות והאמצעית מקבל ערך נמוך יותר מהקיצוניות. בכל של המרחק בין נקודות הקצה קטן ולבסוף מתכנסים לנקודת מינימום מקומית.

למעלה בחרנו נקודה $x\in(a,b)$ אבל ברור שבאותה מידה ניתן היה לבחור נקודה $x\in(b,c)$. סביר לחשוב שכדאי שבכל שלב הבעיה תהיה דומה לבעיה הקודמת, מבחינת היחסים בין מרחקי הנקודות. חישוב שמתבסס על דמיון עצמי זה מראה שבכל שלב כדאי לבחור את הנקודה x בתוך הקטע הרחב כך שהיא מחלקת אותו לפי יחס הזהב $\phi=\frac{\sqrt{5}+1}{2}$. שיטה זו נקראת **bracketing**.

וריאציה אחרת לבחירת x מתבססת על אינטרפולציה של הפונקציה. ישנן שלוש נקודות אז ניתן להתאים אותן ע"י פרבולה ומאחר שנקודות הקצה גבוהות יותר, הפרבולה תהיה בוכה ותקבל מינימום. שם אפשר לבחור את הנקודה החדשה x. במקרה זה יש סכנה לא להתרחק הרבה מ-b ולמעשה "להיתקע במקום" במקום להתקדם. בפרט, אם במקרה b היא נקודת המינימום של הפרבולה, אז לעולם לא נמצא נקודה אחרת. לכן, בפועל בד"כ משלבים בין האסטרטגיות

את השיטה הקודמת אי אפשר להרחיב לפונקציות רב-ממדיות. **Downhill Simplex** הוא כלל אצבע שמאפשר להתמודד עם פונקציות רב-ממדיות וההנחה היחידה היא שהפונקציה המדוברת היא רציפה. הבעיה עם השיטה היא שמדובר רק כלל אצבע והשיטה עלולה להתכנס גם לנקודות שאינן נקודות קיצון.

האלגוריתם מתחיל מקבוצה של $n+1$ נקודות כך ש-$f(x_1) \le \dots \le f(x_{n+1})$ ובאופן איטרטיבי מחפש נקודות חדשות שניתן להחליף ככה שערכי הפונקציה יקטנו והסימפלקס שנוצר ע"י הנקודות יתכנס לאפס. לנקודות ההתחלתיות יש השפעה על התקדמות התהליך ותמיד כדאי לבחור נקודות שנראות מתאימות לבעיה הספציפית שמנסים לפתור.

אם יש לנו אלגוריתם למיזעור פונקציה חד-ממדית ניתן להשתמש בו כדי למזער פונקציות בכמה ממדים. פשוט ננסה למזער אותה בכל מיני כיוונים. **שיטת Powell** עובדת בכמה איטרציות. מתחילים מנקודה כלשהי x_0. ראשית, כיוון החיפוש הראשון הוא $d_1 = e_1$ וקטור היחידה הראשון. מחשבים את $\alpha = \text{argmin}_\alpha f(x_0 + \alpha d_1)$. מגדירים $x_1 = x_0 + \alpha d_1$ וממשיכים משם למזער בכיוון $d_2 = e_2$. כך ממשיכים עד שעוברים על כל וקטורי היחידה ונוצרות נקודות $x_0, x_1, \dots, x_n$. כעת בוחרים d_i באקראי ומחליפים אותו ב-$x_n - x_0$ ועכשיו שוב ממזערים לפי כל הכיוונים. ככה ממשיכים עד להתכנסות.

הבעיה העיקרית עם השיטה הזו היא שאם הייתה ירידה משמעותית בערך הפונקציה לפי אחד הכיוונים $x_n - x_0$ יהיה מאוד קרוב לכיוון הזה ואז ע"י בחירה אקראית של כיווני חיפוש והחלפתם בכיוון הזה אנחנו נאבד את היכולת לחפש בכיוונים אחרים. פתרון אפשרי לזה הוא **שיטת Powell המותאמת** שם לא בוחרים באקראי את הכיוון לאותו נחליף אלא בוחרים דווקא את הכיוון שתרם הכי הרבה למזעור הפונקציה. כלומר, בוחרים את $i = \text{argmax}_i |f(x_i) - f(x_{i-1})|$. זה בדיוק פותר את הבעיה הנ"ל.

עד כה התעסקנו במציאת נקודות קיצון מקומיות. עכשיו נראה איך אפשר למצוא מינימום גלובאלי באמצעות שיטה אקראית.

בהינתן קבוע T ניתן להתהלך על הפונקציה בהילוך מקרי באופן הבא: מתחילים בנקודה x^0 כלשהי. בכל שלב מחשבים $x^* = x^n + r$ כאשר $r \sim \mathcal{N}(0, \varepsilon^2)$ משתנה אקראי נורמלי. בסיכוי $p = e^{\frac{f(x^n) - f(x^*)}{T}}$ קובעים $x^{n+1} = x^*$ ואחרת נשארים במקום עם $x^{n+1} = x^n$. בשיטת ה-**simulated annealing** מתחילים מטמפרטורה גבוהה ולאט לאט מקטינים אותה.

משוואות דיפרנציאליות רגילות

משוואה דיפרנציאלית היא משוואה שמגדירה יחס בין פונקציה לבין נגזרותיה. משוואה דיפרנציאלית היא **רגילה** אם הפונקציה היא במשתנה יחיד, כלומר לא מעורבות במשוואה נגזרות חלקיות. **סדר** המשוואה הוא סדר הנגזרת הכי גבוהה שמופיעה במשוואה.

אנחנו נדבר על משוואות מהצורה $y'(t) = f(t, y(t))$ כאשר f פונקציה רציפה בפרמטר הראשון וליפשיצית במשתנה השני. אזי לכל תנאי התחלה $y(t_0) = y_0$ מובטח שקיימת פונקציה יחידה $y(t)$ גזירה ברציפות המקיימת את המשוואה ואת תנאי ההתחלה.

נניח אם כן שיש לנו משוואה כזאת ונניח שאנחנו עובדים בדיסקרטיזציה של h בזמן. אז יש לנו וקטור זמן $t_j = jh$. נסמן ב-$y_j = y(t_j)$ את הערכים האמיתיים של הפונקציה בנקודות הזמן שלנו ונסמן ב-w_j את הקירובים שאנחנו נחשב.

קירוב טיילור של y מסדר ראשון בנקודה t_j נותן לנו

$$\frac{w_{j+1} - w_j}{h} = f(t_j, w_j)$$

קירוב זה נקרא **שיטת אוילר הקדמית** זוהי שיטה מפורשת כי אחרי שחישבנו את w_j ניתן באופן ישיר לחשב מהשוויון הנ"ל את w_{j+1}.

מאידך, שוב בשימוש בקירוב טיילור, ניתן לקבל

$$\frac{w_{j+1} - w_j}{h} = f(t_{j+1}, w_{j+1})$$

זוהי **שיטת אוילר האחורית** והיא נותנת ייצוג סתום של הקירוב. בכל מקרה משוואות האלה נקראות **משוואות הפרשים**.

נגדיר את **שגיאת הקיטוע** (truncation error)

$$\tau_j = \frac{y_{j+1} - y_j}{h} - f(y_j, t_j)$$

זהו מדד לכמה שהפונקציה מקיימת את נוסחת ההפרשים. נאמר שמשוואת הפרשים היא **עקבית** אם שגיאת הקיטוע שואפת לאפס ככל שהדיסקרטיזציה קטנה, כלומר אם $\tau_j \xrightarrow[h\to 0]{} 0$. ניתן להוכיח שמשוואות אוילר הן שתיהן עקביות.

נגדיר את **השגיאה הגלובלית** להיות $e_j = y_j - w_j$ ונאמר ששיטה מתכנסת אם $e_{\frac{T}{h}} \xrightarrow[h\to 0]{} 0$ עבור זמן קבוע T כלשהו. אפשר להראות שהשגיאה הגלובלית של שיטת אוילר הקדמית עולה מעריכית עם הזמן T. היא אמנם מתכנסת לאפס ככל שהדיסקרטיזציה קטנה יותר, אך אם הדיסקרטיזציה קבועה החסם על השגיאה עולה מעריכית.

ניתן גם לקרב את הפונקציה ע"י קירוב מסדר שני. אז מתקבלת שיטת ה-**mid step**:

$$\frac{y_{j+1} - y_j}{h} = f\left(t_{j+\frac{1}{2}}, w_j + \frac{h}{2} f(w_j, t_j)\right)$$

ושגיאת הקיטוע היא

$$\tau_j = \frac{y_{j+1} - y_j}{h} - f\left(t_{j+\frac{1}{2}}, y + \frac{h}{2} f(y_j, t_j)\right)$$

אפשר להראות שגם שיטה זו היא עקבית.