

אנליזה נומרית 1

© ארזים

16 במרץ 2017

אנליזה נומרית באה לבצע מתמטיקה "בחיים האמיתיים" - לחשב דברים בצורה מדוייקת, למשל על ידי מחשב, ובצורה יעילה. הבעיות בנושא כזה יכולות לנבוע למשל בגלל:

1. דיוק סופי - איך מייצגים, למשל, את $\sqrt{2}$? איך פותרים את $x^2 = 2$? האם התוצאה מושפעת מחוסר הדיוק? האם האלגוריתם מכניס אי דיוק?

דוגמא נתבונן במערכת המשוואות הבאה:

$$\begin{pmatrix} 1 & 1.0001 \\ 1.0001 & 1 \end{pmatrix} \begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
$$x_{n+1} = y_{n+1} = 0.499975$$

נניח שעשינו טעות של 0.1% במדידת x_n, y_n . כלומר

$$x_n = 0.999, y_n = 1.001$$

אם נפתור מחדש עכשיו, נקבל

$$x_{n+1} = 10.499975, y_{n+1} = -9.50002$$

הבעיה כאן היא שהבעיה שאנחנו מסנים לפתור (המטריצה) היא לא מדוייקת - שינוי קטן בקלט שלה משנה מאוד את הפלט שלה.

2. התיאוריה לא מותאמת לשימוש במחשב.

דוגמאות מציאת ערכים עצמיים, או חישוב

$$\int_{-\infty}^{\infty} e^{-x^2} dx = ?$$

3. אין תיאוריה.

דוגמאות מציאת ערכים עצמיים - צריך למצוא שורשים לפולינומים. מעל מעלה 5, אין נוסחה סגורה. בנוסף,

$$\int_{-\infty}^t e^{-x^2} dx$$

4. (לא יהיה בקורס) דגימה - איך הופכים תופעה רציפה לאוסף מספרים שייצג אותה היטב?

1 ייצוג מספרים בנקודה צפה

אנחנו נוהגים לייצג מספרים בבסיס 10:

$$(32.11)_{10} = 3 \cdot 10^1 + 2 \cdot 10^0 + 1 \cdot 10^{-1} + 1 \cdot 10^{-2}$$

במחשב נח הרבה יותר לנצג בבסיס 2 (יש מתח או לא, יש אור או לא, קל להבחין):

$$(101.11)_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

היינו יכולים לכתוב את אותו מספר בתור

$$(1.0111)_2 \cdot 2^{(10)_2}$$

גישה ישנה, שכבר אינה בשימוש, היא שיטת fixed point - מקצים מספר מסויים של ספרות לפני ואחרי הנקודה. הבעיה עם זה היא שנרצה לשנות את הדיוק לי הסקאלה שבה עובדים - ולכן השימוש היום הוא רק בנקודה צפה (floating point) - מניחים את הנקודה היכן שנוח, וזוכרים במה לכפול. כל מספר במחשב מיוצג על ידי

$$\underbrace{\pm}_{\text{sign}} 1. \underbrace{xxxx}_{\text{mantissa}} \cdot 2^{\overbrace{yyy}^{\text{exponent}}}$$

התקן לייצוג מספרים בעזרת נקודה צפה נקרא IEEE 754. בייצוג double precision: ביט אחד עבור sign, 52 ביטים עבור mantissa (53 כולל הספרה 1 הקבועה שבהתחלה), ועוד 11 ביטים עבור exponent. כל מספר הוא 64 ביטים. באקספוננט ניתן לייצג מספרים בתחום שבין 0 לבין 2047. כדי לאפשר אקספוננט שלילי, יש היסט של 1023, כלומר, אם נסמן את האקספוננט E , נפרש את $E = 1$ בתור -1022 , את $E = 2046$ בתור 1023, והאקספוננט הגדול ביותר שמייצג מספר הוא $E = 0$. משמש לייצוג המספר 0, בעוד $E = 2047$ משמש לייצוג מספרים מיוחדים.

בתור mantissa כל רצפי הביטים הם חוקיים. המספר הגדול ביותר שניתן לייצג בעזרת double precision:

$$1. \underbrace{111 \dots 1}_{52 \text{ bits}} \cdot 2^{1023} \approx 2^{1024} \approx 10^{307}$$

ביזרון הוא ישמר בתור

$$\underbrace{0}_{\text{sign}} \underbrace{11\dots 10111\dots 1}_{\text{exponent mantissa}}$$

המספר (הנורמלי) הכי קטן שאפשר לייצג:

$$1.00\dots 0 \cdot 2^{-1022} \approx 10^{-308}$$

בייצוג הבינארי יש 53 סיביות משמעותיות (52 של mantissa ועוד 1 קבוע בהתחלה).
בייצוג עשרוני, מספר הספרות שלרשותנו הוא $\log_{10} 2^{53} \approx 16$.