I.T.I.S. "A. Avogadro" di Torino

Programma per calcolo dell'integrale col metodo dei rettangoli

Materia: Matematica

Autore: Paolo Cataldo

A.s. 2025/26

Indice generale

Introduzione

Obbiettivo del documento

L’obbiettivo sarà spiegare in modo semplice un programma in linguaggio Java che calcola attraverso il metodo dei rettangoli l’integrale di una funzione.

Fondamenti teorici

Definizione dell’integrale definito

Data una funzione *f(x)*, l’integrale definito in un certo intervallo [a, b] rappresenta l’area compresa tra il grafico della funzione *f(x)*, e le due rette verticali $x = a$ e $x = b$.

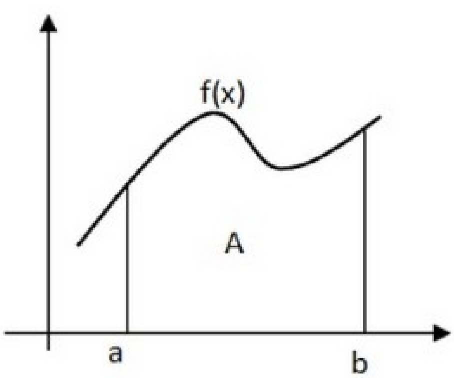


Figura 1: Rappresentazione grafica dell’area A sotto la curva della funzione f(x) nell’intervallo [a,b].

Non è affatto semplice determinare l'area di regioni delimitate da contorni curvilinei. L'idea è di approssimare la superficie con un'altra di cui sia facile calcolare l'area.

Il metodo dei rettangoli

Il metodo dei rettangoli mira a definire l'area della regione A approssimandola con rettangoli (di cui è facile calcolarne l'area).

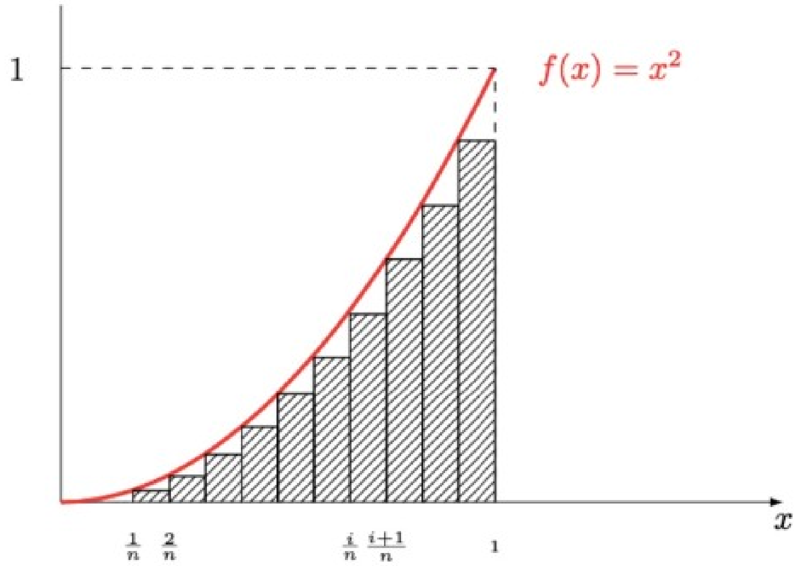


Figura 2: Esempio del metodo dei rettangoli sulla funzione f(x) = x^2

Per far ciò, dividiamo l'intervallo da *a* a *b* in tanti rettangoli di base *h*, che sarà uguale a *h = (b-a)/n* dove *n* corrisponde al numero di rettangoli in cui vogliamo dividere la superficie.

La base di ogni rettangolo corrisponderà quindi ad *h*, mentre l'altezza sarà il valore della funzione in un punto scelto del rettangolo. Sommando l'area dei rettangoli otterremo un'approssimazione dell'area esatta.

Se per calcolare l'altezza del rettangolo utilizziamo il valore all'inizio del rettangolo, otterremo un'area minore di quella reale, ovvero un'approssimazione per difetto.
Se invece usiamo come valore quello alla fine di ogni rettangolo otterremo un'area maggiore, ovvero un'approssimazione per eccesso.

Questo succede perché a seconda di come sale o scende la curva i rettangolo possono essere più piccoli o più grandi rispetto alla zona reale sotto la curva.

L'area approssimata per difetto sarà $sn = h(m1+m2+....+mn)$, dove n è uguale al numero di rettangoli e m è l'altezza di ogni rettangolo, cioè il valore della funzione calcolato nel punto sinistro di ogni rettangolo.

L'area approssimata per eccesso sarà $Sn = h(M1+M2+....+Mn)$, dove n è uguale al numero di rettangoli e M è l'altezza di ogni rettangolo, cioè il valore della funzione calcolato nel punto destro di ogni rettangolo.

Chiaramente più il numero di rettangoli sarà elevato, più il valore dell'area approssimata sarà preciso.

Quindi con n che tende ad infinito, sia sn che Sn saranno uguali all'area sotto la curva vera e propria.

Spiegazione del codice Java

Come eseguirlo

Per eseguire un programma scritto in Java è necessario seguire alcuni semplici passaggi:

1. Installare Java Development Kit (JDK)

Per verificare se è già installato, aprire il terminale (Prompt dei comandi su Windows, Terminale su Mac/Linux) e digitare `java -version`.

Se non è installato, può essere scaricato dal sito ufficiale della Oracle:

https://www.oracle.com/java/technologies/downloads/?er=221886

2. Scrivere il codice in un file di testo con estensione .java (il titolo dovrà essere uguale alla classe del codice, quindi in questo caso sarà CalcoloIntegrale.java).

3. Compilare il codice

Aprire il terminale nella cartella dove si trova il file .java.

Digitare il comando `javac Programma.java`.

Genererà un file .class (es. CalcoloIntegrale.class), che contiene il bytecode eseguibile dalla Java Virtual Machine (JVM).

4. Eseguire il programma

Sempre nel terminale, digitare `java CalcoloIntegrale`.

Il programma verrà eseguito.

Utilizzando un ambiente di sviluppo integrato (IDE), come IntelliJ, Eclipse o NetBeans, la compilazione e l'esecuzione avverranno tramite un semplice clic.

Il codice

Le librerie

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import net.objecthunter.exp4j.Expression;
import net.objecthunter.exp4j.ExpressionBuilder;
```

In Java, una libreria è un insieme di classi e metodi predefiniti che possiamo utilizzare per svolgere operazioni comuni senza doverle riscrivere da zero. Ci permettono di risparmiare tempo e di scrivere un codice leggibile.

Quando scriviamo un programma, spesso abbiamo bisogno di funzionalità aggiuntive, ad esempio per creare interfacce grafiche, gestire eventi o lavorare con file.

Per utilizzare una libreria, dobbiamo importare le classi che ci servono nel nostro programma tramite la parola chiave `import`.

Java offre una vasta gamma di librerie standard, ma possono essere anche utilizzate librerie esterne sviluppate da altri.

Vorrei infatti soffermarmi sulle ultime 2 librerie, `net.objecthunter.exp4j.Expression` e `net.objecthunter.exp4j.ExpressionBuilder`.

Queste classi fanno parte della libreria **exp4j**, una libreria esterna che ho scaricato e inserito nel progetto. È necessario scaricare il .jar da questo link e inserirlo nelle librerie del progetto. Permette di interpretare e calcolare espressioni matematiche inserite come stringhe in modo semplice e veloce. Senza essa, non potrei convertire facilmente una stringa in un'espressione da calcolare. Dovrei scrivere manualmente parte di codice per interpretare e calcolare le espressioni, cosa molto complessa e dispendiosa in termini di tempo.

GUI

La prima parte del codice è composta principalmente da metodi che servono a creare un'interfaccia grafica dove l'utente possa inserire i dati e visualizzare il risultato.

Creo una finestra di dimensione fissa (700x600 pixel) che contiene diversi elementi, come scritte, caselle per scrivere e un pulsante.

Gli elementi principali sono:

- Etichette (scritte), che indicano cosa bisogna inserire, ad esempio "Inserisci la funzione".
- Campi di testo, dove l'utente può scrivere la funzione matematica, i valori dell'intervallo e il numero di rettangoli da usare nel calcolo.
- Un bottone "CALCOLA", che serve per avviare il calcolo

- Area di testo, dove viene mostrato il risultato.

L'algoritmo

Una volta avviato il programma l'utente inserisce:

1. La funzione
2. L'intervallo [a, b]
3. Il numero di rettangoli

Dopo si preme il pulsante "CALCOLA".

Calcolo la lunghezza di ogni rettangolo tramite la formula citata sopra.

```
double lunghezzaRettangoli = (b - a) / (double) rettangoli;
```

Successivamente, calcolo *sn* e *Sn* tramite 2 cicli:

sn:

```
for(int i = 0; i < rettangoli; i++){
    double xi =  a + i * lunghezzaRettangoli;
    double fxi = expr.setVariable("x", xi).evaluate();
    s += fxi * lunghezzaRettangoli;
}
```

- Il ciclo scorre uno per uno tutti i rettangoli e per ogni rettangolo, trova la posizione orizzontale (cioè la base del rettangolo sull'asse x).

  `double xi = a + i * lunghezzaRettangoli;`

  `a` è l'inizio dell'intervallo su cui stiamo calcolando l'integrale (il punto di partenza sull'asse x).

  `lunghezzaRettangoli` è quanto è larga la base dei rettangoli.

  `i` è il numero del rettangolo che stiamo considerando (partendo da 0, poi 1, poi 2 e così via. Il ciclo si fermerà quando il numero dei rettangoli sarà finito, quindi, in informatica, il ciclo continua finché `i < rettangoli`).

Quindi, per trovare la posizione di un rettangolo, il programma parte da a e aggiunge tante volte la larghezza di un rettangolo, quante ne serve per arrivare al rettangolo `i`.

```
a      x1      x2      x3      x4      b
|--------|-------|-------|-------|--------|
   ↑      ↑      ↑      ↑      ↑
  x0     x1     x2     x3     x4
```

Ogni segmento tra due linee verticali è la base di un rettangolo.

La posizione xi di ogni rettangolo si calcola così:

xi = a + i * larghezza_rettangolo

- Calcola l'altezza del rettangolo valutando la funzione in quel punto.

  `double fxi = expr.setVariable("x", xi).evaluate();`

  `fxi` corrisponde al valore della funzione nel punto xi calcolato precedentemente.

  `expr.setVariable("x", xi).evaluate()`: rappresenta la funzione che vogliamo valutare, cioè la formula f(x) inserita. Sostituiamo il valore "x" con xi e calcoliamo il valore. Il risultato sarà memorizzato nella variabile `fxi`, che corrisponderà quindi all'altezza del rettangolo.

- Calcola l'area del rettangolo (base * altezza) e somma quest'area al totale di tutte le altre aree dei rettangoli

  `s += fxi * lunghezzaRettangoli;`

  `s` indica la somma di tutte le aree dei rettangoli. Il simbolo += significa che stiamo aggiungendo al valore di s l'operazione riportata, quindi l'area del rettangolo.

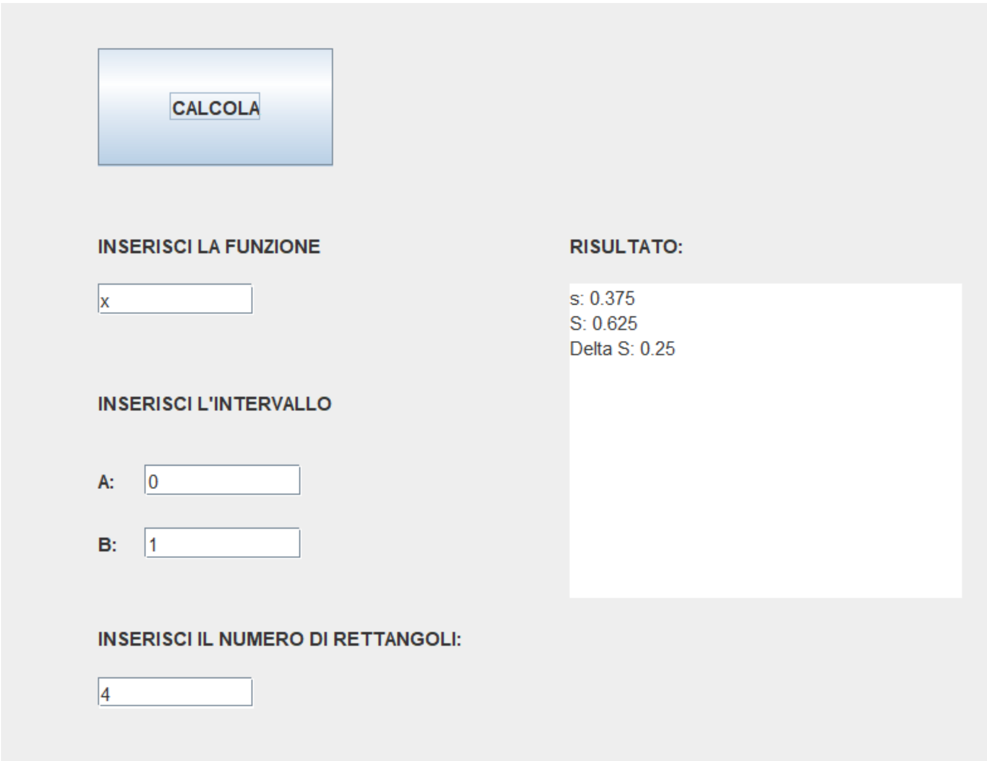
Sn:

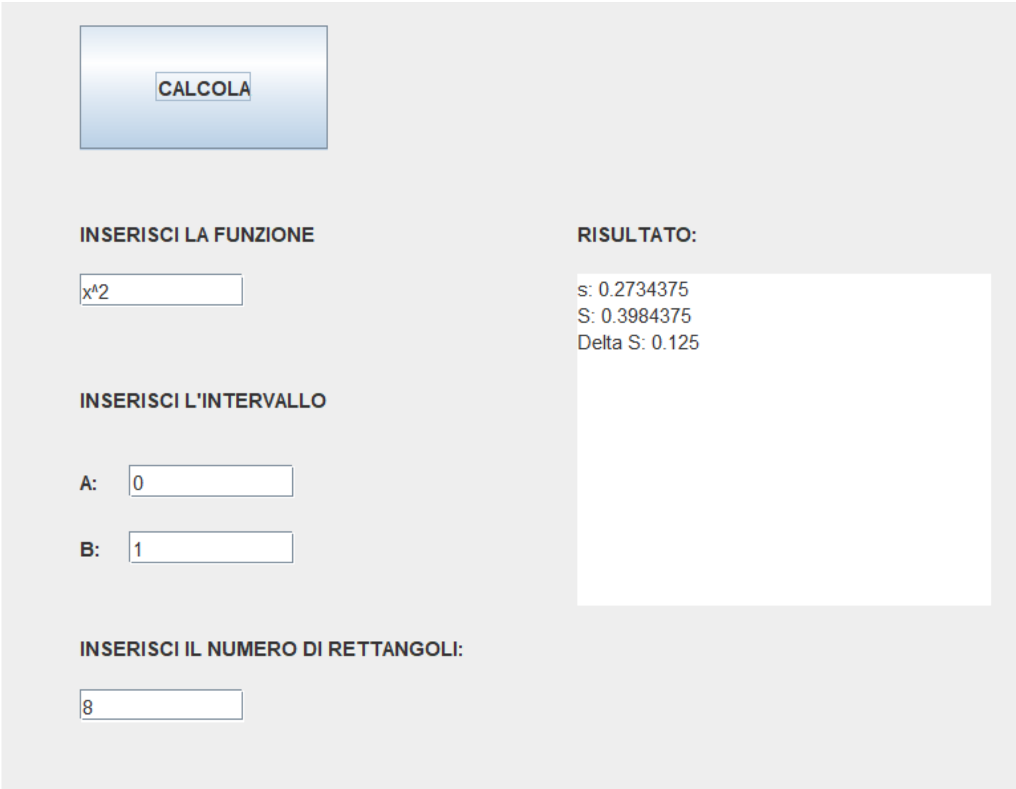
```
for(int i = 1; i <= rettangoli; i++){
     double xi =  a + i * lunghezzaRettangoli;
       double fxi = expr.setVariable("x", xi).evaluate();
       S += fxi * lunghezzaRettangoli;
}
```

Il ciclo è quasi lo stesso, soltanto che in quello per difetto partiamo da i = 0 e si arriva a n-1, mentre in quello per eccesso si parte da 1 e si arriva a n.

Le istruzioni successive servono a calcolare il *deltaS*, facendo la differenza tra *Sn* e *sn*, e stampare il risultato.

Esempi




CALCOLA
INSERISCI LA FUNZIONE
x^2
INSERISCI L'INTERVALLO
A: 0
B: 1
INSERISCI IL NUMERO DI RETTANGOLI:
8
RISULTATO:
s: 0.2734375
S: 0.3984375
Delta S: 0.125


CALCOLA
INSERISCI LA FUNZIONE
2x+1
INSERISCI L'INTERVALLO
A: 0
B: 1
INSERISCI IL NUMERO DI RETTANGOLI:
20
RISULTATO:
s: 1.9500000000000002
S: 2.05
Delta S: 0.09999999999999964