Semesterarbeit Teil 5: Numerische Integration und Abschluss

Aufgabenstellung

(Die Nummern dienen als Referenz für den Rest des Dokumentes)

1. Implementieren Sie die Trapezregel und die Simpsonsche Regel zur numerischen Bestimmung von bestimmten Integralen.
2. Wenden Sie diese Algorithmen auf die Berechnung des Volumens eines Kreisrings an. (Ein Kreisring kann als Rotationskörper eines Kreises aufgefasst werden, wenn der Kreis sich vollständig oberhalb der x-Achse befindet und wenn er um die x-Achse rotiert wird.
3. Vergleichen Sie die Genauigkeit dieser Approximationen miteinander und vergleichen Sie die Resultate mit den exakten Werten, die sie von Hand und mit Sympy bestimmen.
4. Schließen Sie Ihre Semesterarbeit mit einer Diskussion des Einsatzes von Python für praxisbezogene Anwendungen ab. (Dieser Teil ist in diesem Dokument nicht ersichtlich. Meinen Beitrag an der Diskussion finden Sie hier: https://moodle.ffhs.ch/mod/forum/discuss.php?d=25204)

Theorieteil

Für diese Aufgabe habe ich zuerst den Theorieteil dokumentiert und dann den Code gemäss Theorie implementiert.

Trapezregel

Die Trapezregel dient zur Berechnung einer Fläche zwischen einer Funktion und der x-Achse.

Wie man auf Abbildung 1 gut erkennen kann, wird diese Fläche in Trapeze aufgeteilt mit einer Breite von Delta-X, je kleiner Delta-X ist, desto genauer stimmen die Berechnungen.

Für die Berechnung benötigt man also folgende Parameter: Die Funktion f(x), Punkt a auf der x-Achse und Punkt b auf der x-Achse, sowie die Anzahl der Trapeze n (daraus folgt Delta-X).

Die Berechnung in Pseudocode (die Parameter aus dem letzten Absatz werden übergeben):

```
1. Definiere: total=0
2. Definiere: deltaX=(b-a)/n
3. Definiere: i=0
4. Definiere: xi=a+(i*deltaX)
5. Definiere: xii=a+((i+1)*deltaX)
6. Definiere: total+=((f(xi)+f(xii))*deltaX)/2
7. Falls (i+1<n) Dann (Definiere: i+=1; Definiere
   xi=xii; Gehe zu 5) Sonst fortfahren
8. Fertig: total entspricht der Approximation der Fläche
```

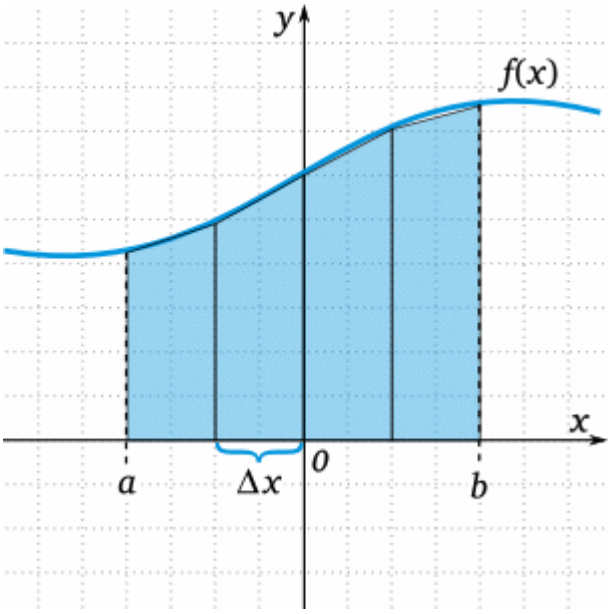

Abbildung 1 http://matheguru.com/images/trapezregel2.png

Dieser Pseudo-Code entspricht der Berechnung jedes einzelnen Trapezes und wird für die Implementation so mit Python umgesetzt als Funktion im «tools.py»-Script.

Simpsonsche Regel

Die Simpsonsche Regel dient zur Berechnung einer Fläche zwischen einer Funktion und der x-Achse.

Wie man auf Abbildung 2 erkennen kann, wird diese Fläche in Parabeln aufgeteilt. Diese Stücke werden dann berechnet. Auf Abbildung 2 werden nur 2 Parabeln erstellt (also n=2), für unsere Berechnungen werden wir n offenlassen.

Für die Berechnung benötigt man also folgende Parameter: Die Funktion f(x), Punkt a auf der x-Achse und Punkt b auf der x-Achse, sowie die Anzahl der Trapeze n (daraus folgt Delta-X).

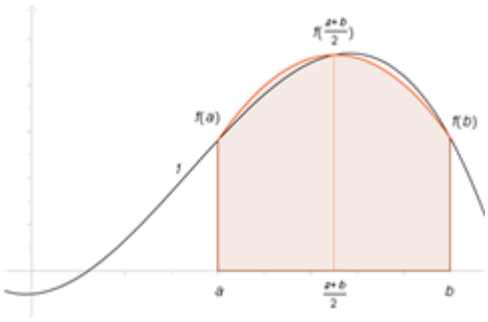

Abbildung 2 http://www.mathepedia.de/html/th_numerik/a_liste/c_numerischeintegration/Simpson_rule.aspx?w=240&h=156

$$\frac{\Delta x}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \cdots + 4y_{n-1} + y_n)$$

Abbildung 3 Visualisierung des Pseudocodes für die Simpsonsche Regel

Die Berechnung in Pseudocode (die Parameter aus dem letzten Absatz werden übergeben):

```
1.  Definiere: total=f(a)
2.  Definiere: deltaX=(b-a)/n
3.  Definiere: lastXValue=b-deltaX
4.  Definiere: x=a+deltaX
5.  Definiere: total+=4*f(x)
6.  Definiere: total+=2*f(x + deltaX)
7.  Definiere: x+=2*deltaX
8.  Falls (x<lastXValue) Dann (Gehe zu 5) Sonst fortfahren
9.  Definiere: total+=4*f(lastXValue)
10. Definiere: total+=f(b)
11. Definiere: total*=deltaX/3
```

Dieser Pseudo-Code entspricht der Berechnung der Simpsonsche Regel und wird für die Implementation so mit Python umgesetzt als Funktion im «tools.py»-Script.

Weitere ausführliche Theorie und Formeln zur Simpsonsche Regel sind hier zu finden: http://www.mathepedia.de/Simpsonsche_Formel.aspx

Oder in diesem Video gut erklärt: https://www.youtube.com/watch?v=N0kFSTDvDcw

Kreisring / Torus

Referenzen zu den folgenden Annahmen und Behauptungen:

- Referenz 1: https://de.wikipedia.org/wiki/Kreisring
- Referenz 2: https://de.wikipedia.org/wiki/Rotationsk%C3%B6rper
- Referenz 3: https://de.wikipedia.org/wiki/Torus
- Referenz 4: http://www.nibis.de/~lbs-gym/AnalysisTeil3pdf/Rotationsvolumen.pdf

Gemäss meinem Verständnis geht es in der Aufgabe 2 um einen Torus. Ein Kreisring ist ein zweidimensionaler Körper und hat entsprechend kein Volumen. Wird jedoch ein Kreis als Rotationskörper um die x-Achse rotiert, wie dies zusätzlich definiert wird, entsteht ein Torus. Ich habe diesbezüglich auch einen Beitrag im Moodle-Forum verfasst, erhielt jedoch bisher keine Bestätigung eines Dozenten. https://moodle.ffhs.ch/mod/forum/discuss.php?d=25487

Für diese Semesterarbeit werde ich also davon ausgehen, dass ein Torus gemeint ist.

Implementation

Struktur

Für diese Arbeit verwende ich folgende Scripts:

- «test.py»: Dieses Script dient zur Überprüfung der Implementationen.
- «view.py»: Dieses Script enthält eine einfache Rechner-Funktion für die Benutzereingabe.
- «tools.py»: Dieses Script enthält die Funktionen für die Trapezregel und für die Simpsonsche Regel.

Scripts

Für diese Semesterarbeit habe ich den Python-Code ausführlich kommentiert, damit ich mehr Platz für den Theorieteil der Aufgabe habe.

Resultat

Für die Aufgaben 2 & 3 hatte ich Probleme, welche ich auch auf Moodle angesprochen habe: https://moodle.ffhs.ch/mod/forum/discuss.php?d=25763 Leider habe ich diesbezüglich bisher keine Antwort erhalten, weder vom Dozenten noch von meinen Mitstudierenden. Zum Glück habe ich einen Lösungsansatz in einem Dokument aus dem Internet «Referenz 4» gefunden. Ich habe meine Arbeit auf diesen Erkenntnissen aufgebaut.

Das Script berechnet einerseits das Volumen (wie durch die Aufgabenstellung vorgesehen) und andererseits hat der Benutzer die Möglichkeit, selbst eine Funktion anzugeben, für welche dann die Fläche zur X-Achse durch die beiden Algorithmen berechnet wird.

Für das Volumen des Torus gilt:

$$V_{Torus} = 4\pi R \int_{-r}^{r} \sqrt{r^2 - x^2}\, dx$$

Abbildung 5 http://www.nibis.de/~lbs-gym/AnalysisTeil3pdf/Rotationsvolumen.pdf

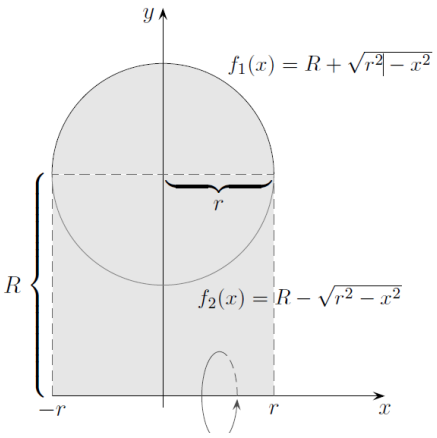

Abbildung 4 http://www.nibis.de/~lbs-gym/AnalysisTeil3pdf/Rotationsvolumen.pdf

Diese Formel habe ich im «tools.py»-Script als «volume_torus_integral(...)» eingebaut.

Vergleich der Werte gemäss Aufgabe 3

Statt dass ich diese Werte wirklich von Hand bestimme (was bei dreidimensionalen Figuren sehr schwierig werden dürfte), habe ich die Funktion «volume_torus_simple(…)» implementiert, welche die einfache Berechnung, also von Hand darstellen soll.

```
Berechnetes Volumen für einen Torus mit r=`5` und R=`20`:
- Einfache Berechnung (vonHand):          `9869.604401089358`
- Integrale Berechnung:                   `9869.60440108937`
- Differenz (Approximation):              `1.0913936421275139e-11`
```

Wie man gut erkennen kann, ist die Abweichung nur sehr klein, in diesem Fall nur ungefähr `0.00000000001`.

Vergleich für `f(x)=2`, `start_x=-3`, `end_x=3`:

Methode:	Von Hand	Trapezregel	Simpsonsche Regel	Abweichung
n=1000	12	12.00000000000001	12.0	1.0658141036401503e-14
n=100	12	11.99999999999998	12.0	1.9539925233402755e-14
n=10	12	11.999999999999998	11.999999999999998	0.0

Ich gehe davon aus, dass es sich bei diesen Werten um Rundungsfehler und Floating-Point-Fehler handelt. Für einen besseren Vergleich müsste man komplexere Funktionen berechnen, dafür reicht jedoch der Platz nicht mehr in dieser Semesterarbeit.