ComputerMathematik
5. Computerpraktikum zum Abgabetermin 2.12.2024, 23:59

21. Nutzen Sie `cylinder` um die folgenden Aufgaben zu lösen:

   (a) Rotation der Kurve $r(x) = x\sin(\pi x)$, $x \in [0.05, 0.8]$ um die **x-Achse**.

   (b) Die Rotation der Kurve $r(h) = \sqrt{1-h^2}$ für $h \in [-1, 0.5]$.

   (c) Kegelstumpf der Höhe 4 mit unterem Radius 3 und oberem Radius 1.

   Hinweis: Ist die Höhe in Richtung der Rotationsachse korrekt? Haben Sie `axis equal` eingeschaltet?

22. **Fläche zwischen zwei Funktionen**:
   Bestimmen Sie den Inhalt der von zwei Funktionen $f_1(x)$ und $f_2(x)$ eingeschlossenen Fläche und visualisieren Sie sowohl die Funktionen als auch diese Fläche.

   (a) Beschränken Sie sich auf auf polynomiale Funktionen mit maximalem Grad 4, z.B. $f_1(x) = x^3 - 7*x^2 + (59*x)/4 - 27/4$ und $f_2(x) = -x^4 + (21*x^3)/2 - 36*x^2 + (91*x)/2 - 13$

   (b) Sie können analog zum MatheGuru[1] vorgehen. Hierbei sollen die Schnittpunkte automatisch von Matlab berechnet werden. Probieren Sie `solve(......,'MaxDegree',4)` aus.

   (c) Testen Sie Ihr Programm mit mehreren Funktionspaaren.

   (d) Visualisieren Sie die Funktionen, die Schnittpunkte und die Fläche (`fill`).

   (e) Matlab2019: Ermöglichen Sie eine Eingabe der beiden Funktionen als String von der Tastatur (`input`), mit anschließender Umwandlung in eine symbolische Funktion (`str2sym`).

   Hinweise: `syms`, `int`, `solve`, `vpasolve`, `real`, `imag`,`vpa`, `double`, `matlabFunction`, `max`, `min`, `plot`, `fplot`, `fill`, `linspace`, `legend`, `title`, `input`, `str2sym`

23. **Roulette-Simulation**:
   Schreiben Sie eine Matlabfunktion `[gewinn] = bsp_23_fkt(wahl,einsatz)`, die in Abhängigkeit der Eingabeparameter (`wahl,einsatz`) ein Roulettespiel simuliert.
   Folgende vereinfachte Regeln sind zu beachten:

   - Es besteht nur die Wahl zwischen `gerade/ungerade` oder einer Zahl zwischen 0 und 36.
   - Setzt der Spieler erfolgreich auf `gerade/ungerade` erhält er das Doppelte seines Einsatzes zurück. Fällt die Zahl 0, so erhält er die Hälfte seines Einsatzes zurück. Andernfalls verliert er seinen Einsatz.
   - Setzt der Spieler erfolgreich auf eine Zahl, so erhält er das 36-fache seines Einsatzes zurück. Andernfalls verliert er seinen Einsatz.

   Vergleichen Sie das Setzverhalten zweier Spieler A und B in einem scriptfile bsp_23.m.

   - Beide Spieler besitzen ein Guthaben von 50000 Euro.
   - Spieler A setzt immer einen Euro auf `gerade`.
   - Spieler B setzt immer einen Euro auf eine Zahl (z.B. 2).
   - Lassen Sie beide Spieler so lange spielen, bis die jeweiligen Guthaben 1 Euro unterschritten haben.
   - Wer macht mehr Spiele? Erklären Sie Ihre Beobachtung mathematisch.

   Visualisieren Sie den Verlauf der beiden Guthaben in einer Grafik.

   Hinweise: `randi`, `switch`, `case`, `otherwise`

[1] `https://matheguru.com/integralrechnung/flache-zwischen-zwei-funktionen.html`

24. **Koch-Schneeflocke**:

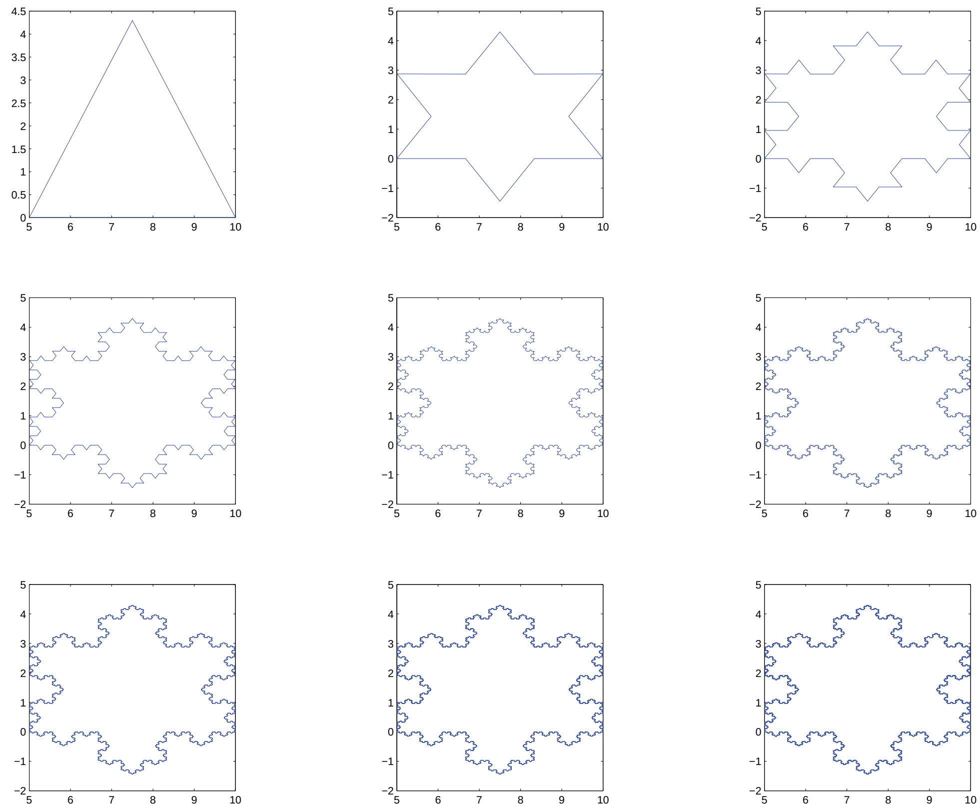
Schreiben Sie eine Matlabfunktion `[p] = bsp_24_fkt(n,A,B,C)` die in Abängigkeit eines vorgegebenen gleichseitigen Dreiecks mit den Punkten A,B,C die dazugehörige Koch-Schneeflocke (siehe Koch-Kurve[2]) berechnet.
Folgende Hinweise sind zu beachten:

- Der Eingabeparameter n bezeichnet die Anzahl der auszuführenden Iterationen.
- Der Ausgabeparameter p soll die Punkte des entstandenen Polygons enthalten.
- Zeichnen Sie die ersten neun Iterierten in unterschiedliche Grafiken.
- Berechnen Sie den Flächeninhalt des jeweiligen Polygons.

Testen Sie Ihr Programm für die Punkte

$$A = (5, 0), B = (10, 0), C = (7.5, \sqrt{18.75})$$

in einem scriptfile bsp_24.m. Das Resultat sollte ähnlich zu nachfolgender Grafik sein.



Hinweise: `polyarea`, `subplot`, `axis square`

[2] `https://de.wikipedia.org/wiki/Koch-Kurve`

25. **Goldbachsche Vermutung**[3]:

Jede gerade Zahl größer 3 kann als Summe von zwei Primzahlen geschrieben werden, d.h. es gilt für alle geraden Zahlen $k \in [4, n]$ ($n >= 4$), dass zumindest zwei Primzahlen p_1, p_2 existieren so dass $k = p_1 + p_2$. (Meist gibt es mehr als ein Paar dessen Summe eine gerade Zahl k ergibt.)

(a) Schreiben Sie eine Funktion `single_goldbach(k)` welche die Anzahl der Primzahlpaare für k zurückgibt (z.B., $k = 694$ besitzt 19 Zerlegungen).

(b) Schreiben Sie eine Funktion `count_goldbach(n)` welche die Anzahl der Primzahlpaare für alle geraden Zahlen $k \in [4, n]$ bestimmt und diese Daten (als Vektor) zurückgibt.

(c) Bestimmen sie nach dem Aufruf der Funktion `count_goldbach(n)` diejenige gerade Zahl mit den meisten Primzahlzerlegungen ($n = 100.000 \Longrightarrow k = 99.330$) und zeigen Sie die Anzahl der Zerlegungen graphisch an (z.B. `bar` oder `histogram`).

(d) Messen sie die Laufzeit Ihrer Funktion `count_goldbach(n)` mit `tic` und `toc`[4] für verschiedene n, z.B. $n \in$ `10.^[3,4,5,6]`.

Hinweise: `primes`, `isprime`, `tic`, `toc`,`bar`, `hold on`, `legend`, `title`

Abgabe der Lösungen:

Die Abgabe der Lösungen (`*.m`-Files und Grafiken) muß über Moodle[5]. erfolgen.

Die Filenamen **müssen** dem Schema `bsp_nummer`, gefolgt von der Filextension, entsprechen. Andere Filebezeichner zählen nicht als abgegebene Files.

Abzugebende Files (auch als ein zip-File möglich):

`bsp_21.m`, `bsp_21.jpg`
`bsp_22.m`, `bsp_22.jpg`
`bsp_23.m`, `bsp_23.jpg`
`bsp_24.m`, `bsp_24_fkt.m`, `bsp_24.jpg`
`bsp_25.m`, `single_goldbach.m`, `count_goldbach.m`, `bsp_25.jpg`

[3] https://en.wikipedia.org/wiki/Goldbach's_conjecture
[4] https://de.mathworks.com/help/matlab/ref/tic.html
[5] http://moodle.uni-graz.at