Hinweise zur Programmieraufgabe 2

3. November 2010 Anton Prochel

1 Generelle Hinweise

1. Teilen Sie die Erstellung des Programmcodes in möglichst kleine Schritte auf.
2. Testen Sie diese kleinen Schritte nach Möglichkeit schon.

2 Konkrete Anleitung

Hier nun eine konkrete (Trippel-)Schritt für Schritt Anleitung, wie man bei der Erstellung der einzelnen Programmteile der Programmieraufgabe vorgehen könnte. Sie setzt die in dem Einführungskurs zu Matlab erworbenen Kenntnisse voraus. (Diese Anleitung sieht länger aus, als sie ist.)

Öffnen Sie nun das MATLAB-Programm.

1. Erstellen Sie eine Datei namens `f_a2.m`. Darin erstellen Sie eine Funktion
   `function [f] = f_a2(x)`.
   Eingabeparameter `x` und Ausgabeparameter `f` sollen folgende Bedeutung haben: `x` soll die Stelle, an der eine Funktion ausgewertet wird, sein und `f` deren Funktionswert $f(x)$.

   $\Rightarrow$: Statt hier gleich die geforderte Funktion aus Aufgabe 2 berechnen zu lassen, ist es zum Entwickeln und Testen der restlichen Programmteile sinnvoll, als Funktion z.B. $f(x) = 1$, oder $f(x) = x$ oder $f(x) = x^2$ zu nehmen; also etwas, das man selbst noch im Kopf integrieren kann, um damit die Richtigkeit der `qf_...`-Funktionen zu testen. Für $f(x) = x$ würde in der Funktion also stehen

   ```
     f = x
   end
   ```

2. Erstellen Sie in einer Datei namens `qf_rechteck.m` Code, der für genau ein Intervall a, b nach der Rechteckregel die Näherung des Integrals berechnet.
   Lassen Sie N und die Teilintervalle noch unberücksichtigt.
   Machen Sie noch keine Funktion aus dem Code, sondern definieren Sie oberhalb Ihres Codeabschnittes einfache Werte für a und b, z.B.

   ```
   a = 0
   b = 2
   ```

3. Lassen sie ihren Code laufen, d.h. lassen Sie die Datei `qf_rechteck.m` ausführen. Beheben Sie Flüchtigkeitsfehler. Falls Sie $f(x) = 1$ gewählt haben, sollte als Näherungswert des Integrals nun 2 herauskommen, für $f(x) = x$, entsprechend $(a - b) \cdot f(a)$, Null. Wenn nicht, suchen Sie weiter nach dem Fehler. Er kann nicht weit sein.

4. Klappt alles, testen Sie das ganze für $f(x) = x$ und nochmal für $f(x) = x^2$. Bekommen Sie auch da die richtigen Werte, sollten Sie nun von der richtigen Funktionsweise des vorläufigen Codes für die spätere Funktion `qf_rechteck` überzeugt sein.

5. Um diesen Code zu vervollständigen, muss noch das N eingebaut werden. D.h. das Integral muss über N Teilintervallen berechnet werden.

   Überlegen Sie, wie dafür die `for`-Schleife beschaffen sein muss, die nun um den bisherigen Code in `qr_rechteck.m` gelegt werden muss. Denn dieser soll nun ja N mal ausgeführt werden auf äquidistanten Teilintervallen von [a, b].

6. Testen Sie den (bis auf etwaige Fehler) fertigen Code für die Rechteck-Integration, sinnvollerweise mit den bisherigen Einstellungen für die Invervallgrenze a, b und f(x) für $N = 1$, (Nichts sollte

sich ändern) und anschließend für andere N.

7. Von der Richtigkeit Ihres Codes nunmehr überzeugt, müssen Sie aus dem Code in der Datei `qf_rechteck.m`, der bislang ein Skriptcode ist, eine Funktion machen, welcher Sie `a`, `b`, `N` übergeben, und welche `I`, die Näherung des Integrals, ausgibt.

8. Verfahren Sie mit den anderen beiden `qf_...`-Funktionen, `qf_trapez` und `qf_simpson` genauso.

9. Damit sind die Arbeiten an allen Funktionen abgeschlossen.

   Vergessen Sie nicht, nun in der Datei `f_a2.m` die Funktion aus Aufgabe 2 berechnen zu lassen (und nochmals zu testen, dass dies tatsächlich geschieht.)

10. Als letztes ist noch das Skript namens `p02.m` zu erstellen, welches die geforderten Aufrufe der Integrationsfunktionen durchführt und die Plots erstellt:

    Erstellen Sie dazu einen Vektor (z.B. `n_n`), der die verschiedenen N enthält, für die die `qf_...`-Funktionen aufgerufen werden sollen.

11. Rufen Sie nun aus einer Schleife heraus die `qf_...`-Funktionen für die geforderten N auf und speichern Sie die Ergebnisse jeweils wieder in einem Vektor, z.B `n_rechteck`, `n_trapez`, `n_simpson`.

    Somit haben Sie vier Vektoren: einen, in dem die N gespeichert sind, und drei, in denen die den N zugehörigen Näherungswerte für die drei Quadraturformeln gespeichert sind.

12. Zu Plotten sind aber nicht die Näherungswerte, sondern die logarithmierte positive Differenz zwischen den Näherungen und einer „exakten“ Lösung, also die Logarithmen der Integrationsfehler.

    Berechnen Sie zuerst einen exakten Wert für das angegebene N mit der sinnvollsten `qf_...`-Funktion. Bilden Sie mit diesem die Differenz für alle berechneten Näherungswerte. Speichern Sie die Logarithmen der Differenzen in einem Vektor ab. Achtung, sind die Differenzen nicht positiv, ist das Ergebnis vielleicht unerwartet.

    Nun können Sie die berechneten Fehler plotten. Vergessen Sie nicht, dass über den Logarithmus von h zu plotten ist, und vorher h aus N zu berechnen ist.

13. Schauen Sie sich die Ergebnisse an, überlegen Sie ob das alles wirklich so stimmen kann. Verbessern Sie die Fehler.

14. Dies ist ein essentieller Punkt!

    Stellen Sie sicher, dass Ihr Skript durchläuft und sinnvolle, ausführliche, übersichtliche, (richtige) sowie leicht und schnell verständliche Ausgaben erzeugt.

    Wenn alles stimmt, schauen Sie den Code nochmals durch und überlegen Sie, wo standardmäßig Kommentare hingehören (z.B. bei Funktionen), wo Sie sich welche Kommentare (oder mehr Kommentare) wünschen würden, bzw. wo welche für das schnelle Verständnis sinnvoll sind. Schreiben Sie sie alle hin.

15. Schreiben Sie in den Anfang der Skriptdatei die Kommentare für die Abgabe hinein: Name, wie Aufzurufen (Matlab, octave), welche Dateien mitgeschickt werden und wozu sie gut sind, ..., sonstiges Mitteilenswertes.

    Schicken Sie Ihre Lösung ab.

Fertig!