Vom Tropfenzählen zum Fundamentalsatz

Einführung des Integralbegriffs mit den TI-CAS-Rechnern

Josef Böhm und Wolfgang Pröpper

Einführung des Integralbegriffs mit den TI-CAS-Rechnern

Vom Tropfenzählen zum Fundamentalsatz

Josef Böhm und **Wolfgang Pröpper**

"*soliche ding sind zu vill sachen nütz*"
Albrecht Dürer, Underweysung der messung
mit dem zirckel und richtscheyt, Nürnberg 1525

Vorwort

Der Begriff des Integrals ist einer der zentralen Aspekte der Analysis, und es gibt eine Fülle von Literatur, die sich diesem Thema widmet. Mit dem vorliegenden Büchlein soll nicht das 1002. Märchen über die Einführung des Integralbegriffs erzählt werden. Sein Ziel ist zu zeigen, wie Riemannsche Summen und damit verwandte Begriffe mit Verwendung eines der CAS-TI (TI-89, TI-92, TI-92PLUS, Voyage™200) sowohl numerisch/symbolisch als auch anschaulich dargestellt und untersucht werden können.
Die Autoren haben die Hoffnung, dass mit Hilfe des Programmpakets `integ()` der Zugang zum Integralbegriff für Lernende erleichtert werden kann. Dies mag auf dem Wege der Demonstration durch den Lehrer[1] im Unterricht erfolgen. Aber es ist ebenso denkbar, dass Schüler durch Experimentieren mit den Programmen (und das kann auch eine Art Spielen sein) eigene Erfahrungen zum Integralbegriff sammeln können.
Zur besseren Orientierung ist das Büchlein in zwei Teile gegliedert:
Der erste Abschnitt beschreibt detailliert, wie das auf der Diskette mitgelieferte Programmpaket auf einem der gängigen CAS-TI installiert und bedient wird (inkl. Anleitung für TI-Connect), so dass auch Anwender, die noch wenig Erfahrung mit diesen Geräten besitzen, damit umgehen lernen. Zusätzlich wird im ersten Teil noch die Programmstruktur dargestellt
Im zweiten Teil werden in einem Workshop 14 Aufgabengruppen angeboten, die zeigen, wie `integ()`eingesetzt werden kann. Es ermöglicht dem Benutzer, Einblicke in den Riemannschen Integralbegriff zu erlangen, und zeigt, wie dieser Begriff aus ganz unterschiedlichen Blickwinkeln gesehen werden kann.
Die Inspiration zur Implementation auf dem TI erhielt Josef Böhm, der Vater der Derive User Group und Herausgeber des Derive Newsletter, von Francisco José Santonja [1] aus Spanien. Er hatte schon früher, anlässlich der „International Spring School on the Didactics of Computer Algebra“ 1992 in Krems ein DERIVE-Paket zur Behandlung des Integralbegriffs entwikkelt [2],[3], zu dem später auch noch Terence Etchells [4] wesentliche Beiträge lieferte. Santonjas Beitrag war ihm Anlass, ein TI-92-Paket `riemann()` zu entwickeln, das wie leider viele Public-Domain-Programme, nur vom Autor selbst gefahrlos verwendet werden konnte. Wolfgang Pröpper ergänzte einige Teile und versah das Ganze mit einer Oberfläche, die auch dem ungeübten Benutzer ein weitgehend sicheres Arbeiten mit den Programmen ermöglicht.

Würmla und Nürnberg im Frühjahr 1999 und Frühjahr 2004
J. Böhm, W. Pröpper

---

[1] Wenn in diesem Text von „Lehrern“ oder „Schülern“ die Rede ist, sind damit selbstverständlich auch Lehrerinnen oder Schülerinnen gemeint. Die Reduzierung nur auf das männliche Geschlecht soll nicht als Diskriminierung aufgefasst werden, sondern dient ausschließlich der möglichst einfachen Sprachgestaltung.

Teil I Programmbeschreibung

1 Installation und Programmstart

Auf der beigefügten Diskette befinden sich mehrere gruppierte Fassungen des Programms `integ()`. Es sind dies:

Integ92.92g	Integ92P.9xg
Integ89.89g	Integv2.v2g

Die Gruppen können jeweils mit der entsprechenden *GraphLink* Software bzw. mit TI Connect auf den entsprechenden Rechner übertragen werden. Die Namen der transferierten TR-Programme sind:

AUS1	ENDE	SIMP	VGL_GRAF
AUS2	FEIN	SM	W1
BASIS	HILF	SUMS	ZUFREG
BEISP	INTEG	TRA	ZUFZER
BOX	INTFUNK	UEBER	
DAT	PUL	VGL	
EINST	SCREEN	VGL_ALLE	

Die extensions dieser Dateien[2] auf dem PC sind *.92P, *.89P, *.9xP oder *.v2P, jeweils für den TI-92, TI-89, TI-92 Plus oder Voyage™200. Entsprechend sind die Programme sind auf dem TI-92, dem TI-89, dem TI-92 Plus und und dem Voyage™200 lauffähig. (Alle im Text abgebildeten screen shots wurden mit dem TI-92 PLUS erstellt.)

Übertragung mit *GraphLink* (hier am Beispiel des TI-92 Plus)

- Verbinden Sie die serielle Schnittstelle Ihres PC und den Taschenrechner mit dem Graph Link Kabel
- Starten Sie *GraphLink* für Ihr Taschenrechnermodell (für den Voyage™200 wählen Sie *GraphLink* für den TI-92 Plus)
- Wählen Sie Link-Senden: Ein Fenster "Sende Dateien an TI-92 Plus" öffnet sich (Abb. 1.1).
- Stellen Sie das Laufwerk, in dem sich die Diskette befindet (z. B. A:) ein.

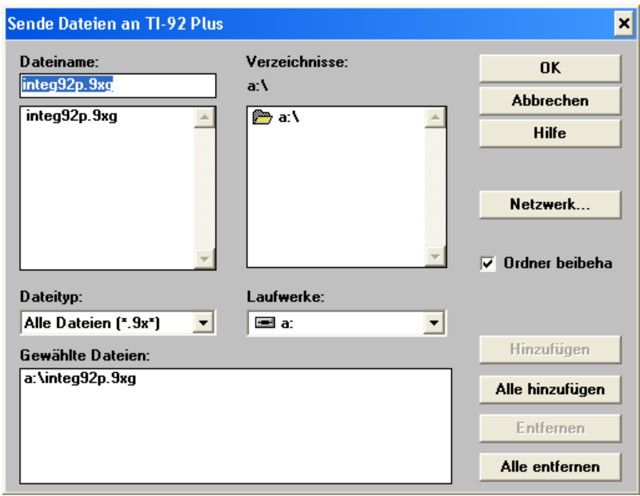


Abb 1.1 Das Sende-Fenster des *GraphLink*

[2] Beim TI-89 ist die Datei AUS2.89P nicht enthalten.

- Integ92P.9XP wird im Feld Dateiname angezeigt. (Wenn nicht, stellen Sie den Dateityp "Alle Dateien" ein und klicken dann auf den angezeigten Dateinamen.)
- Klicken Sie die Schaltfläche "Alle hinzufügen" an: Im Fenster "Gewählte Dateien" wird nochmals der Dateiname aufgeführt.

- Wenn Sie nun die Schaltfläche "OK" anklicken, werden die in der Gruppe enthaltenen Dateien auf Ihren TI-92 Plus übertragen. Dabei können zwei Fälle auftreten:

  - Ist das Kästchen "Ordner beibehalten" angeklickt, wird bei der Datenübertragung auf Ihrem TI-92 Plus ein Verzeichnis `integ` erzeugt, in das alle Dateien kopiert werden.
  - Ist "Ordner beibehalten" nicht angeklickt, öffnet sich ein Fenster zur Auswahl eines Zielordners auf Ihrem TI-92 Plus (Abb. 1.2). Falls Sie diese Alternative bevorzugen, sollten Sie vor Beginn der Übertragungsprozedur auf Ihrem TI-92 Plus ein Verzeichnis mit einem Ihnen geeignet erscheinenden Namen anlegen (z.B. wie in Abb. 1.2 angedeutet: integral) und die Dateien von `integ()` dorthin kopieren.

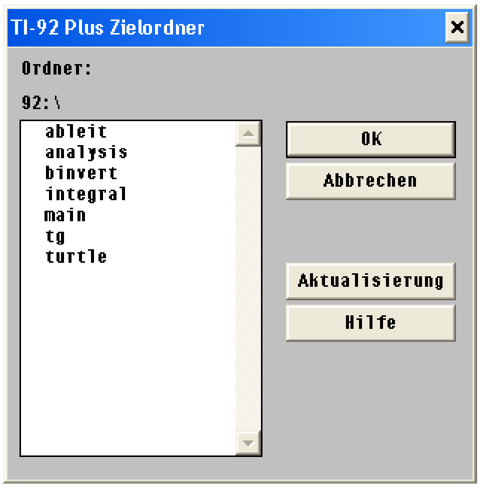


Abb. 1.2 Auswahl des Zielverzeichnisses

Übertragung mit TI Connect (am Beispiel des Voyage™200)

- Aus TI Connect wird der TI Device Explorer gestartet. Wenn der TR über das serielle *GraphLink*- oder das USB_ Kabel an den PC angeschlossen ist, wird nach kurzem Suchen der Verzeichnisbaum des TR angezeigt. Abb. 1.3 zeigt, dass kein Verzeichnis mit Namen integ angelegt ist.
- Ein Klick auf das Icon  öffnet den Windows Explorer. Man macht das Diskettenlaufwerk (hier: A:) zur aktuellen Adresse. Die vier eingangs erwähnten group files werden angezeigt. Wir wählen IntegV2.

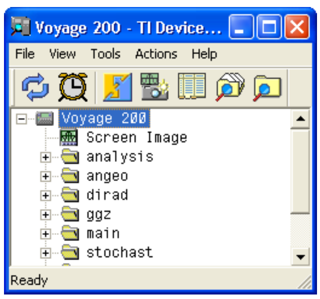


*Abb. 1.3*TI DeviceExplorer vorher

- Wenn man nun mit gedrückter linker Maustaste das group icon auf die Wurzel des Verzeichnisbaums im TI Device Explorers zieht, werden die einzelnen Dateien der Gruppe in der oben angegebenen Reihenfolge übertragen.

- Nach Abschluss der Übertragung zeigt sich das TI Device Explorer Fenster mit dem neuen Verzeichnis (hier: `integ`).

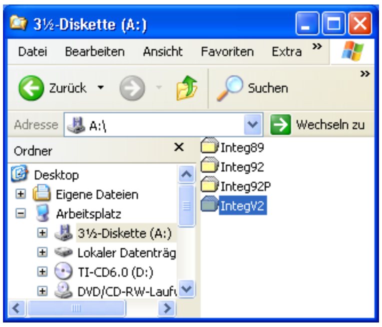


Abb. 1.4 Windows Explorer

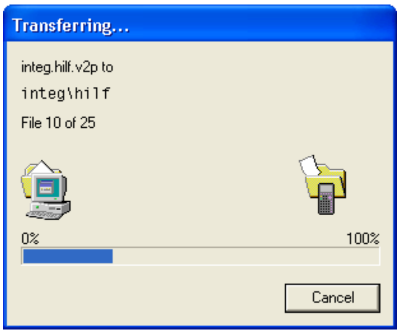


Abb. 1.5 Übertragungsanzeige

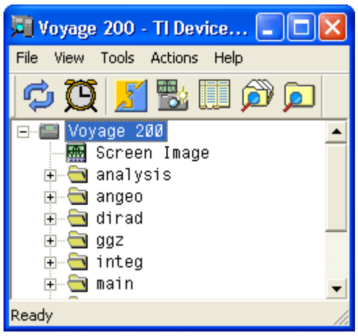


Abb. 1.6 TI Device Explorer nachher

- Das Paket `integ()` belegt auf dem TR ca. 28 kBytes an Speicherplatz. Bei extensivem Betrieb können nochmals bis zu 8 kBytes dazukommen. Vergewissern Sie sich deshalb vor dem Übertragen und dem Programmstart durch einen Blick in die Speicherverwaltung (mit 2nd MEM), ob noch genügend Speicherplatz vorhanden ist. Andernfalls lagern Sie Dateien oder Verzeichnisse auf Ihren PC aus oder verschieben Sie in den Archivspeicher des TR (nicht möglich beim TI-92).

- Lösen Sie nun das *GraphLink* Kabel und wechseln Sie in das Verzeichnis `integ` bzw. das von Ihnen angelegte Zielverzeichnis. (Das aktuelle Verzeichnis erkennen Sie in der Statuszeile, links unten. Um in ein anderes Verzeichnis zu wechseln müssen Sie mit der Taste MODE in das Mode-Fenster gehen und mit ⊙⊙ die Liste der vorhandenen Verzeichnisse öffnen (Abb. 1.6). Nun wählen Sie mit der Cursor-Taste das gewünschte Verzeichnis aus und bestätigen Sie die Wahl durch zweimaliges Drücken der ENTER-Taste.)

- Zum Programmstart schreiben Sie in die Eingabezeile des Hauptbildschirms `integ()`(aber vergessen Sie das Klammerpaar nicht!) und schließen mit ENTER ab. Sie rufen damit das Hauptprogramm auf, welches die Steuerung übernimmt. Der Bildschirm Ihres Taschenrechners sieht nun, wie in Abb. 1.7 gezeigt, aus.

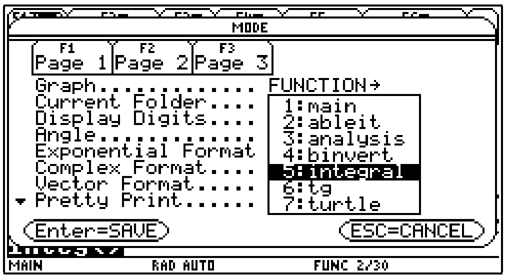


Abb. 1.6 Auswahl des aktiven Verzeichnisses

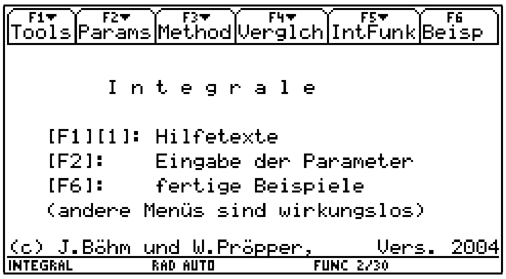


Abb. 1.7 Der Eröffnungsbildschirm

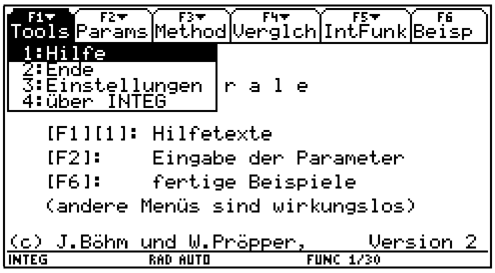
Nach dem Start von `integ()`muss eine Funktion angegeben werden, für die verschiedene Arten von Riemannschen Summen untersucht werden können. Die Eingabe der Funktion und der damit zusammenhängenden Parameter erfolgt über den Menüpunkt F2. Die Untersuchung der Riemannschen Summen wird vom Methodenmenü F3 gesteuert. Im Menü F4 werden Verfahren zum übersichtlichen Vergleich verschiedener Methoden angeboten. Mit den im Menüpunkt F5 vorhandenen Möglichkeiten kann der Begriff der Integralfunktion näher beleuchtet werden. Unter F6 ist eine Reihe von Beispielen verborgen, die von den Autoren als in gewisser Weise elementar betrachtet werden und dem Einsteiger in das Programmpaket einen möglichst bequemen Zugang ermöglichen. Schließlich sind im Menü F1 eine Ende-Routine zum Verlassen des Programms und einige Hilfen enthalten.

Die Beschreibung dieser Menüs und damit die Gebrauchsanweisung des Programms erfolgt in den nächsten Abschnitten.

Wenn in der weiteren Folge vom TI-92 die Rede ist, dann sind gleichermaßen dessen Nachfolger TI-92+, TI-89 und Voyage™200 gemeint.

2.1 Das Menü [F1]: `Tools`

Das Menü wird durch Drücken der Funktionstaste [F1] geöffnet. Es bietet vier Alternativen an. Die Wahl erfolgt entweder durch Verschieben der markierten Zeile mit den Cursor-Tasten und einem abschließenden [ENTER] oder durch Drücken einer der gewünschten Ziffern [1], [2], [3] oder [4]. Bei der zuerst genannten Methode hat man noch die Möglichkeit, das Menü vor dem Drükken der [ENTER]-Taste mit [ESC] ohne eine Aktion zu verlassen. (Diese Anleitung zum Umgang mit Menüs gilt sinngemäß auch für die anderen Menüs.)

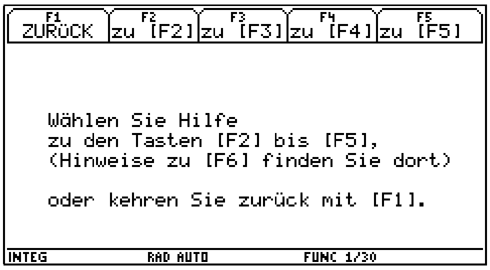


Alternativen des Menüs [F1]

[1]: Hilfe

Hinter diesem Menüpunkt verbirgt sich eine kleine Online-Hilfe. Beim ersten Aufruf werden zwei einleitende Seiten gezeigt. Der Wechsel zur Folgeseite erfolgt mit [ENTER]. Diese beiden Seiten werden bei einem späteren Aufruf der Hilfe in einer Sitzung nicht mehr angeboten.

Die Hilfstexte zu den Menüpunkten [F2] bis [F5] des Hauptprogramms (s. auch Abb. 1.7) erreicht man durch Drücken der jeweiligen Funktionstaste. Dort sind in komprimierter Form auf einigen Seiten Hilfstexte abgelegt, wobei jeweils die Anzeige der Folgeseite mit [ENTER] ausgelöst wird. Nach dem Durchlaufen einer solchen Textfolge kehrt man wieder zu der Verteilerseite zurück. Man kann dann andere Hilfstexte einsehen oder mit der Funktionstaste [F1] die Hilfe verlassen und wieder zum Hauptbildschirm zurückkehren.



Der Verteiler der Hilfe

[2]: Ende

Dieser Menüpunkt ist ausgesprochen wichtig, weil das Programm `integ()` immer über ihn verlassen werden sollte. Nur dadurch ist sichergestellt, dass sich Ihr TI-92 nach Beendigung der Arbeit mit `integ()` in einem definierten Zustand befindet und bei weiterer Verwendung keine unerwünschten oder unerwarteten Nebeneffekte auftreten.

Zum Durchlaufen der Ende-Routine werden zwei Alternativen angeboten:

Alternativen bei 2:`Ende`

- Ein Abschluss mit ENTER beendet die Arbeit mit `integ()`. Das heißt, alle während des Programmlaufs generierten Variablen werden gelöscht und der TI-92 wird wieder in den Zustand versetzt, in dem er vor dem Start von `integ()` war. So werden alle Einstellungen wie Grafik-Modus oder Dezimalanzeige etc. auf ihren ursprünglichen Zustand zurückgesetzt. Der Zweck dieser Maßnahme ist einerseits das Beseitigen nicht mehr benötigter Variablen (was sonst zu Speicherplatzproblemen führen könnte). Andererseits wird dem Anwender die Mühe genommen, den Rechner wieder von Hand in den gewohnten Betriebsmodus bringen zu müssen.
- Schließt man die Arbeit jedoch mit ESC ab, bleiben wesentliche Variable und Einstellungen erhalten. Sie können anschließend im Home Screen (Ausgangsbildschirm, Computer-Algebra--Fenster) des TI-92 angesehen und weiter bearbeitet werden. Diese Möglichkeit ist sinnvoll (und es wird an gegebener Stelle darauf verwiesen), weil bei der Arbeit mit `integ()` Terme oder Matrizen auftreten können, die ohne die Scroll-Möglichkeiten des Home Screen nicht vollständig überblickt werden können.

  Um nach endgültigem Abschluss der Arbeit wieder zu einem definierten Zustand zurückzukehren, muss `integ()` nochmals gestartet und sofort wieder mit F1 2 ENTER beendet werden.

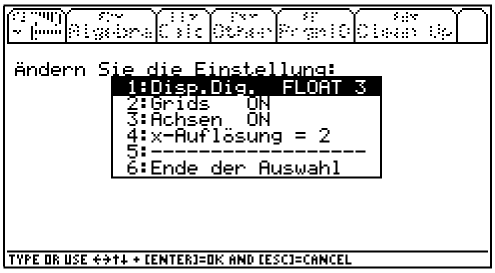
  Ein anderer Grund für das Beenden mit ESC kann sich ergeben, wenn man die augenblickliche Arbeit für andere Aufgaben nur unterbrechen und zu einem späteren Zeitpunkt wieder mit der gleichen Funktion fortsetzen will. Beim neuerlichen Start zeigt der TI-92 nämlich die aktuellen Parameter an und man kann sofort die unterbrochene Arbeit wieder aufnehmen, ohne neuerlich den Funktionsterm und alle anderen Parameter eingeben zu müssen. (In diesem Fall ist es außerdem ratsam, das aktuelle Verzeichnis zu wechseln, weil sonst gültige interne Variable versehentlich geändert werden könnten.)

  Für eine reine Arbeitsunterbrechung läßt man den TI-92 einfach liegen. Die Abschaltautomatik (APD - Automatic Power Down) ist so konstruiert, dass der ausgeschaltete Rechner nach Drücken der ON-Taste wieder dort aufsetzt, wo er über die Automatik geschlossen wurde.

Ob die Aufräumarbeiten beim Schließen des Programms erfolgreich waren, sieht man beim Inspizieren des `INTEG`-Verzeichnisses (= `folders`) mit [2nd] [VAR-LINK]: Dieses Verzeichnis darf nur noch die im Abschnitt 1 aufgelisteten 25 PRGM-Dateien enthalten.

[3]: Einstellungen

`integ()` stellt beim Start die meisten Betriebsparameter auf gewisse Standardwerte ein. Bei einigen Parametern mögen von Fall zu Fall andere Werte sinnvoll sein.



Änderbare Einstellungen

Zum Ändern der Stellenzahl öffnet sich ein weiteres Popupmenü. Die Alternativen [2] ... [4] wirken wie Flip-Flops.
Die Änderungen werden im Popupmenü sofort angezeigt und bleiben für die Dauer einer Sitzung (oder wenn sie wieder variiert werden) aktiv.
Ausnahmen: Die Stellenzahl bei den Vergleichen ist auf Float 6 (bei [F4] [1]) bzw. Fix 4 (bei [F4] [2]) festgelegt. Bei den Integralfunktionen ([F5] [2]) wird grundsätzlich die x-Auflösung 2 verwendet (= `xres` im [WINDOW]-Fenster).

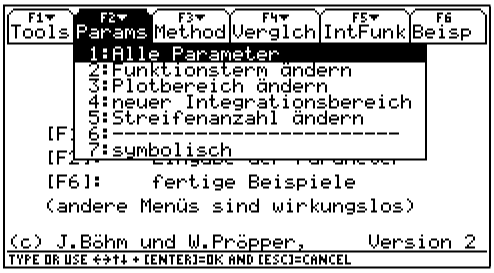
[4]: über INTEG

In diesem Menüpunkt werden einige Geheimnisse zur Entstehungsgeschichte von `integ()` gelüftet und die E-Mail-Adressen der Autoren preisgegeben.

2.2 Das Menü [F2]: `Params`

Dieses Menü dient zur Eingabe der zu untersuchenden Funktion und ihrer relevanten Parameter. Es muss immer zuerst aufgerufen und ausgeführt werden, wenn man eigene Funktionen untersuchen will.

Die einzelnen Unterpunkte:



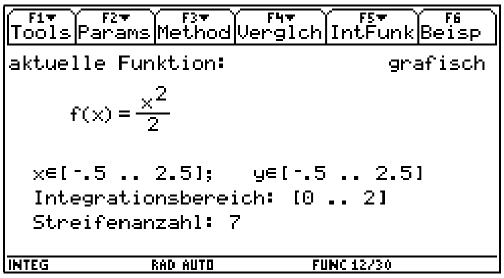
Das `Params`-Menü

[1]: Alle Parameter

Es werden nacheinander Eingabedialoge für den Funktionsterm, den Plotbereich, den Integrationsbereich und die Streifenanzahl geöffnet. Nach der Eingabe aller Werte durchläuft `integ()` eine Plausibilitätsroutine, die einige Sicherheitsprüfungen für einen störungsfreien Ablauf des Programms vornimmt. Gegebenenfalls wird mit einem knappen Hinweis zu einem der Dialoge verzweigt, um Korrekturen vornehmen zu können.

Wenn alle Parameter eingegeben und geprüft sind, zeigt der Bildschirm die aktuellen Werte an.

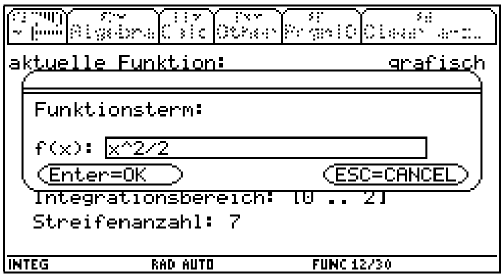
Eine detaillierte Beschreibung der erforderlichen Eingaben erfolgt bei den nachfolgenden Menüpunkten, in denen auf Möglichkeiten zum Ändern der einzelnen Parameter eingegangen wird.



Der aktuelle Bildschirm

$\boxed{2}$: Funktionsterm ändern

Das einzeilige Eingabefeld verlangt die Eingabe eines Funktionsterms wie zB `1/2x^2` oder `sin(x)`. Natürlich verträgt `integ()` auch sehr viel komplexere Terme. Sogar abschnittsweise definierte Funktionen unter Verwendung der `sign(`-Funktion oder der `when`-Klausel können Verwendung finden. (Möglichkeiten dazu werden im Teil II erörtert.)

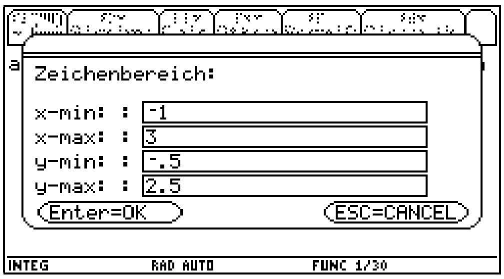


Eingabe des Funktionsterms

Die Plausibilitätsprüfung untersucht, ob ein zulässiger Funktionsterm eingegeben wurde, d.h., ob er der üblichen mathematischen Syntax entspricht. In diesem Sinne zulässig sind auch symbolische Funktionsterme, wie z.B. `h(x)`. Bei Eingabe eines symbolischen Funktionsterms schaltet das Programm gleichzeitig auf den sogenannten *symbolisch*en Modus (s. $\boxed{\text{F2}}$ $\boxed{7}$ bzw. $\boxed{\text{F3}}$) um. Auch im Home Screen vordefinierte Funktionen wie etwa `kost(x)` können verwendet werden.

$\boxed{3}$: Plotbereich ändern

Beim Plotbereich sind jeweils für das Argument x und den Funktionswert y die Unter- und die Obergrenze anzugeben (Abb. 2.2.4). Die Plausibilitätsprüfung kontrolliert, ob die Grenzen numerisch sind und ob die Untergrenze kleiner als die Obergrenze ist.



Kalibrieren des Zeichenbereichs

Beim Festlegen des Plotbereichs sollte man darauf achten, dass die Graphen im größeren Fenster des im Verhältnis 1:2 vertikal geteilten Bildschirms gezeichnet werden. Damit der Betrachter die Orientierung behält, wird als *x*- und als *y*-Einheit die 1 gewählt; mit diesen Einheiten sind auch die Gitterpunkte (*grids*) eingezeichnet.

(Die `grids` können mit [F1] [3] [4] ausgeschaltet werden, was bei einem größeren Zeichenbereich sehr zu empfehlen ist.)

Der Plotbereich ist beim Menü [F4] und in der Betriebsart *symbolisch* eigentlich irrelevant. Er muss dennoch in diesen Fällen mit angegeben werden, d.h., man kann die Eingabe mit [ENTER] einfach überspringen.

[4]: neuer Integrationsbereich

Der Integrationsbereich ist die Menge der Argumente in denen das Integral bzw. eine Riemannsche Summe bestimmt werden soll. Einzugeben sind Unter- und Obergrenze.

Ändern des Integrationsbereichs

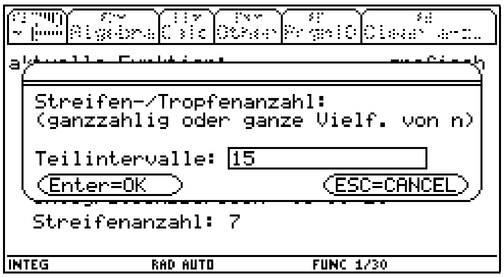
Die Plausibilitätskontrolle prüft nicht, ob der Integrationsbereich innerhalb des Plotbereichs liegt. Jedoch dürfte eine derartige Wahl in den meisten Fällen keine sehr aussagekräftigen grafischen Darstellungen liefern. Ebenso wird eine Untergrenze, die größer als die Obergrenze ist, nicht moniert. Es ist sogar durchaus lehrreich in diesem Fall nicht nur den Vorzeichenwechsel bei den entsprechenden Summen zu beobachten, sondern auch zu sehen, wie der Zeichenvorgang in umgekehrter Richtung abläuft.

Wenn die eingegebenen Grenzen nicht numerisch sind, wird kontrolliert, ob der Untergrenze die Variable *a* und/oder der Obergrenze die Variable *b* zugewiesen wurde. Diese Einschränkung mag auf den ersten Blick als einengend betrachtet werden, ist jedoch für einen fehlerfreien Programmablauf erforderlich (s. 3. Die Programmstruktur).

Wenn eine der Integrationsgrenzen symbolisch ist, wird vom System auf die Betriebsart *symbolisch* umgeschaltet, weil dann ein Plotten von Streifen nicht mehr möglich ist. Das CAS des TI-92 ermöglicht jedoch auch Untersuchungen von symbolischen Termen (s. Menü [F3]) und geht damit weit über die Möglichkeiten eines rein numerischen Taschenrechners hinaus.

[5]: Streifenanzahl ändern

Die Streifenanzahl gibt in den meisten Fällen die Anzahl der Teilintervalle an, in die der Integrationsbereich zerlegt wird. Bei einer Methode (s. [F3] [A]: `Monte Carlo`) wird hier anstelle einer Streifenanzahl eine Tropfenanzahl erfasst.



Eingabe der Streifen-/Tropfenzahl

Die Streifenanzahl kann, ebenso wie der Integrationsbereich, numerisch oder symbolisch sein. Bei einem numerischen Wert wird geprüft, ob er nichtnegativ und ganzzahlig ist. Als symbolischer Wert ist die Variable *n* oder ein ganzzahliges Vielfaches von zugelassen.

6 **-------------------**

Dieser Unterpunkt hat keine Wirkung. Er dient nur der optischen Gliederung des Menüs.

7: symbolisch / grafisch

Mit dem Menüpunkt 7 kann zwischen dem sogenannten *grafisch*en und *symbolisch*en Betriebsmodus umgeschaltet werden. Dabei wirkt F2 7 wie ein Kippschalter. Der eingestellte Modus ist in der rechten oberen Ecke des aktuellen Bildschirms sichtbar.

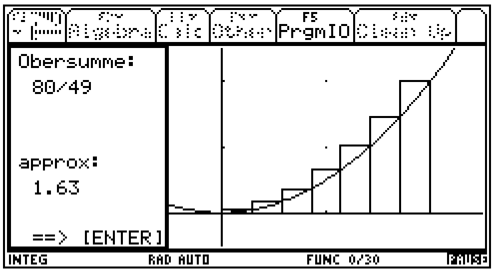
Diese beiden Modi beziehen sich auf die im Menü F3 aufrufbaren Methoden:
Bei der Betriebsart *grafisch* wird die ausgewählte Riemannsche Summe neben ihrem numerischen Wert auch als Streifenmuster veranschaulicht (siehe die Abbildungen auf der nächsten Seite). Damit sollen dem Lernenden die verschiedenen Methoden grafisch vor Augen geführt werden.
Wie oben dargelegt, können aber Integrationsgrenzen und Streifenanzahl auch nichtnumerisch, d.h. symbolisch eingegeben werden. Dann ist eine grafische Darstellung nicht mehr sinnvoll und es werden nur noch die berechneten Terme angezeigt. In manchen Fällen, bei denen eine grafische Behandlung möglich (und durchaus geboten) ist, werden jedoch die numerischen Terme im exakten Modus so umfangreich, dass für die gleichzeitige Darstellung von Graph und Term der Bildschirm des TI-92 nicht ausreichend ist, oder ein scheinbar falsches Resultat anzeigt (s. Trapezsumme auf Seite 13). Für diesen Fall ist die *symbolisch*e Darstellungsart neben der *grafisch*en angebracht.
Beim Programmstart ist der *grafische* Modus voreingestellt. Oben wurde schon darauf hingewiesen, dass das System bei symbolischen Parametern automatisch in den *symbolisch*en Modus umschaltet. In diesen Fällen kann nicht in den *grafisch*en Modus zurückgeschaltet werden. Das Drücken der Tastenfolge F2 7 ist dann also wirkungslos. Wenn jedoch bei *symbolisch*er Betriebsart die Voraussetzungen für den *grafisch*en Modus wiederhergestellt werden (zB die Streifenanzahl wird von *n* auf 6 gestellt), kann bzw. muss der Anwender manuell, d.h. durch F2 7, in den *grafisch*en Modus schalten.
Sie können das Programm im *symbolisch*en Modus zwingen, nicht „exakt" zu rechnen, indem Sie zumindest eine Integrationsgrenze als Dezimalzahl angeben – anstelle von 5 geben Sie 5.0 ein. Das verkürzt in manchen Fällen enorm die Rechenzeit, da der Rechner viel Zeit und Speicherplatz benötigt, ein Ergebnis mit vielen Wurzeln und Brüchen zu formatieren und auf den Bildschirm zu bringen. Sie finden ein Beispiel dazu im Workshop unter der Aufgabe 5.7.

2.3 Das Menü [F3]: `Method`

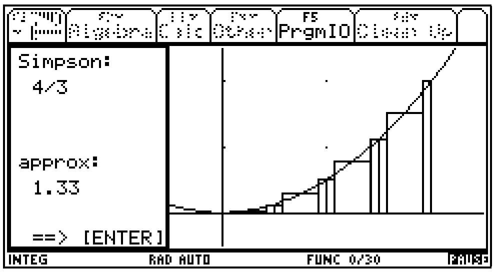
Das Methodenmenü

Mit der Funktionstaste [F3] wird ein Menü geöffnet, welches 11 Methoden zur Auswahl anbietet. Ihre Bezeichnungen sagen dem Mathematiker, welche Inhalte damit verknüpft sind. Der Lernende kann diese Inhalte beim Umgang mit dem Programm erfahren. Deshalb werden diese Inhalte weiter unten in diesem Abschnitt beschrieben. Die Beispiele in diesem Abschnitt beziehen sich weitgehend auf die Funktion $f(x) = x^2/2$.

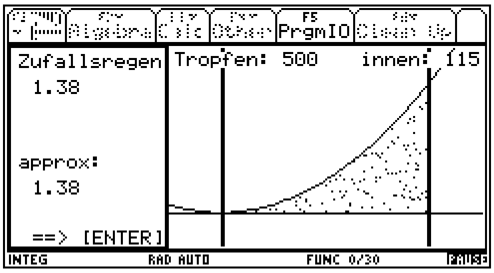
In der Darstellungsart *grafisch* (s. Menü [F2] [7]) wird der Bildschirm vertikal im Verhältnis 1:2 geteilt. Der Graph der aktuellen Funktion wird im rechten Fenster geplottet und die Streifen der gewählten Methode werden eingezeichnet bzw. werden bei der Monte-Carlo-Methode als Punkte für gefallene Tropfen abgebildet. Anschließend zeigt das linke Fenster den Namen der Methode sowie die errechnete Summe in exakter und approximierter Darstellung (Standard: 3 signifikante Ziffern). Durch Drücken der [ENTER]-Taste kommt man zurück zum Hauptbildschirm mit den aktuellen Parametern.



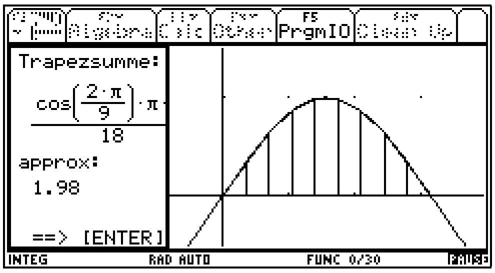
Obersumme zu $y = \frac{1}{2}x^2$ *mit n*=7



Simpson-Verfahren mit n = 4



Monte Carlo mit 500 *Tropfen*

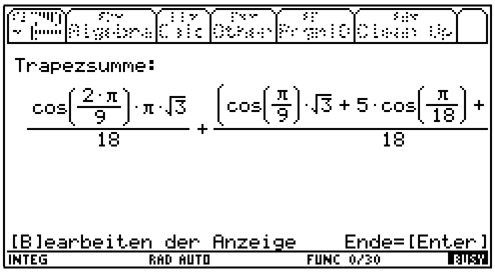


Ein scheinbar falsches Resultat

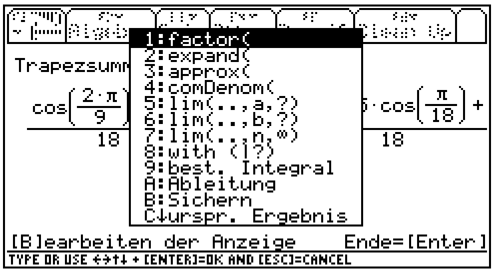
Die letzte Abbildung zeigt ein (scheinbar) fehlerhaftes Resultat: Offensichtlich handelt es sich um die Trapezsumme mit 9 Teilintervallen zur Funktion $f(x) = \sin(x)$ im Bereich von 0 bis π. (Diese Funktion ist in der Beispielsammlung, die mit [F6] aufgerufen werden kann, enthalten.) Der approximierte Wert ist sicher korrekt, weil bekanntlich $\int_0^{\pi} \sin(x)\, dx = 2$ ist.

Für den angeblich exakten Wert $\dfrac{\cos\left(\frac{2\pi}{9}\right)\cdot\pi}{18}$ erhält man jedoch überschlagsweise 0,13, also einen Wert der weit daneben zu liegen scheint. Eine Auflösung des Widerspruchs ergibt sich, wenn man die Trapezsumme im Modus *symbolisch* betrachtet[3]. Der exakte Term ist sehr viel umfangreicher, als im Split-Screen-Modus des TI-92 darstellbar ist, ja er nimmt sogar mehr Platz ein, als der volle Bildschirm anbietet.

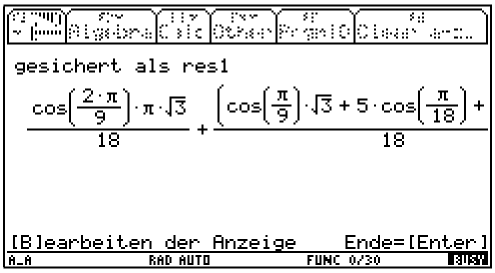
Für weitere Untersuchungen kann man im *symbolisch*en Modus mit der Taste $\boxed{\text{B}}$ ein Popupmenü mit 13 Alternativen zum Bearbeiten des angezeigten Terms öffnen. Die folgenden Screenshots zeigen den Bildschirm zuerst nachdem der unübersichtliche Term unter dem Namen `res1` gesichert wurde[4] und anschließend mit der TI-92-Funktion `factor` vereinfacht wurde.



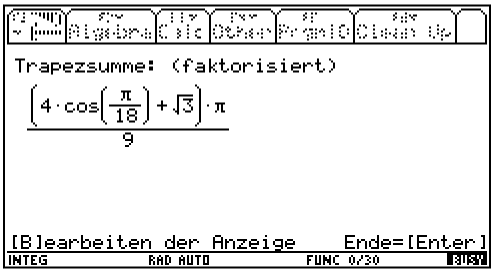
Der (fast) vollständige Term



Das `[B]earbeiten`*-Menü*



Gesichert als `res1`



Das Resultat wurde vereinfacht

Eine sehr viel wichtigere Rolle als im eben gezeigten Fall spielt der *symbolisch*e Mode aber bei Funktionen in allgemeiner Darstellung. Als Beispiel werde die Funktion $f(x) = x^2/2$ im Integrationsbereich $[a\,..\,b]$ mit n Streifen betrachtet. Wie oben dargelegt, schaltet das System in diesem Fall automatisch in die Betriebsart *symbolisch* um. Nach dem Aufruf der Methode

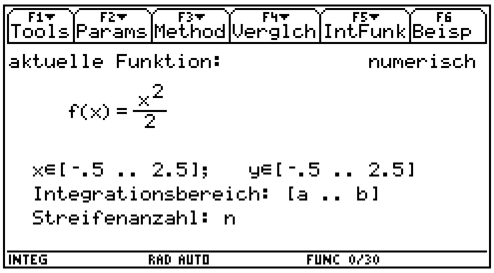
---

[3] Um zur vollständigen symbolischen Termdarstellung zu gelangen, muss man die grafische Darstellung mit $\boxed{\text{ENTER}}$ abschließen, dann mit $\boxed{\text{F2}}$ $\boxed{7}$ in den *symbolischen* Mode schalten und mit $\boxed{\text{F3}}$ $\boxed{5}$ noch einmal die Methode „Trapezsumme" aufrufen.

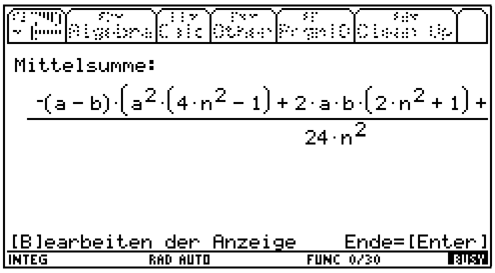
[4] Die Namensvergabe erfolgt automatisch, um keine Konflikte mit den Namen von globalen Variablen entstehen zu lassen.

„Mittelsumme" ([F3] [6]) erhält man zuerst einen einigermaßen unübersichtlichen Ausdruck in a, b und n. Dieser wird, wie hier nicht gezeigt, unter dem Namen `res2` gesichert. Der Term kann nun mit Hilfe des `[B]earbeiten` Menüs weiter untersucht bzw. verändert werden. Zuerst wird der Grenzwert für $n \to \infty$ berechnet und dann `expand`iert man diesen Term. Lässt man schließlich die Ableitung dieses Terms nach b berechnen (Tastenfolge: [B] [A] und Frage „`Ableitung nach:`" mit [B] [ENTER] beantworten), so ist man beim Hauptsatz der Differential- und Integralrechnung angelangt. Ebenso kann man zum Vergleich das bestimmte Integral berechnen lassen.[4]

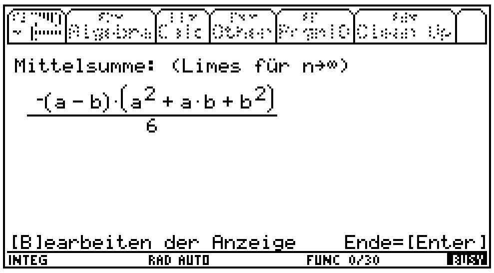
Wenn man das Programm mit [F1] [2] [ESC] verlässt, können die unter den Namen `res1` und `res2` gespeicherten Terme im Home Screen des TI-92 durch horizontales Scrollen genauer betrachtet werden (Abb. auf Seite 16).



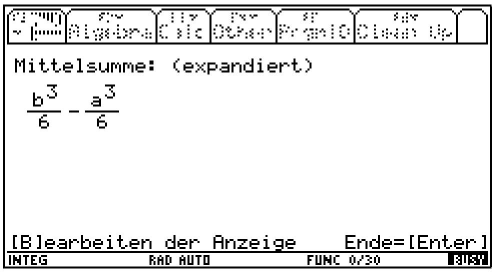
Die aktuelle Funktion



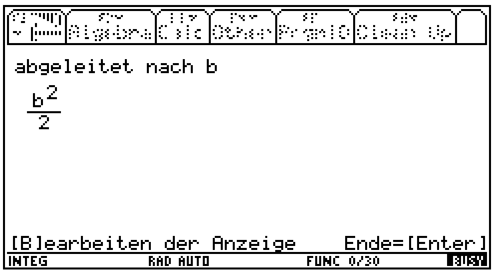
Nach dem Aufruf von [F3] [6]



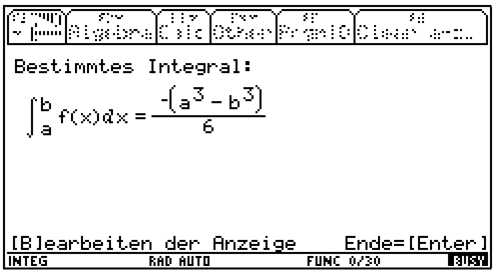
Berechnung des Limes mit [B] [7]



Vereinfachung mit [B] [2]



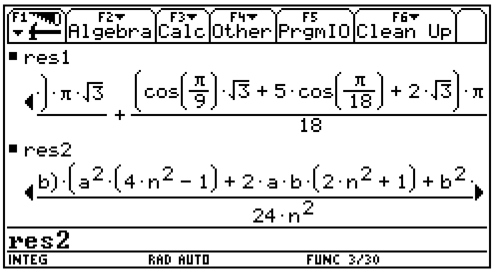
Ableitung nach b



Bestimmtes Integral mit [B] [9]

[4] Es erweist sich oft als günstig, das erste Ergebnis des TI-92 vor der Grenzwertbildung vereinfachen zu lassen. Auch zusätzliche Bedingungen (zB Nichtnegativität) können hilfreich sein.

Das nebenstehende Bild zeigt den Home Screen des TI-92 nach Verlassen des Programms mit F1 2 ESC und Ausgabe der Variablen `res1`, ganz nach rechts gescrollt, sowie `res2`, der Sicherung der Mittelsumme von Seite 15, etwa mittig gescrollt. Beim Wiedereinstieg in das Programm `integ()` ist die Funktion, mit der es verlassen wurde, mit allen Parametern weiter aktiv.



`res1` *und* `res2` *im Home Screen*

Der Bildschirm zeigt sich mit allen voreingestellten Parametern.

Bei den Bearbeitungsverfahren, die eigene Eingaben verlangen, kann es, wenn unsinnige Werte eingegeben werden, zu einem Programmabsturz mit einer Fehlermeldung kommen, weil Prüfroutinen, die dies abfangen, zu viel Aufwand erfordert hätten. Falls sich das Programm mit einer Fehlermeldung verabschiedet, muss man mit F5 in den Home Screen schalten. Wenn man `integ()` dann neu startet und es sofort wieder mit der Tastenfolge F1 2 ENTER verlässt, entstehen keine „bleibenden” Schäden, d.h. im aktuellen Verzeichnis sind alle temporären Variablen gelöscht und es enthält nur die 25 PRGM files, die das Programmpaket `integ()` umfasst.

Abschließend werden noch einige Bemerkungen zu den implementierten Methoden gemacht:

1: Untersumme und **2: Obersumme:**

Bei Unter- und Obersumme im Riemannschen Sinn wird bekanntlich die zu integrierende Funktion jeweils durch das Minimum bzw. Maximum der Funktion im betrachteten Teil-intervall ersetzt und die sich so ergebende Summe von Rechtecksflächen berechnet. Insofern sind die Bezeichnungen Unter- bzw. Obersumme etwas übertrieben. Denn um eine effiziente Rechengeschwindigkeit zu erreichen, werden hier für Unter- bzw. Obersumme nur Minimum bzw. Maximum des Funktionswerts an den Rändern des betreffenden Teilintervalls genommen. Für monotone Funktionen ist diese Vereinfachung sogar korrekt. Bei nichtmonotonen Funktionen, die keine zu starken Schwankungen aufweisen, erhält man dennoch gute Ergebnisse (siehe dazu auch [4]).

3: Linkssumme und **4: Rechtssumme:**

Für die Streifenhöhe wird jeweils der Funktionswert am linken bzw. rechten Randpunkt des Teilintervalls genommen. Dadurch sind diese beiden Methoden die schnellsten Verfahren. Beim Vergleich der Methoden (s. Menü F4 1) erkennt man sehr schön, dass Unter- und Linkssumme bzw. Ober- und Rechtssumme bei monoton wachsenden Funktionen gleiche Werte liefern. Bei monoton fallenden Funktionen entsprechen sich Unter- und Rechtssumme bzw. Ober- und Linkssumme.

5: Trapezsumme und **6: Mittelsumme:**

Bei der Trapezsumme werden die Rechteckstreifen durch Trapeze ersetzt. In der Mittelsumme bleiben die Rechtecke erhalten, ihre Höhe ergibt sich jedoch aus dem Funktionswert in der Intervallmitte. Trapez- und Mittelsumme sind erste praktisch angewandte Verfahren der numerischen Mathematik. Ihr Fehler ist von quadratischer Größenordnung, während die Rechtecksverfahren 1 bis 4 nur eine lineare Größenordnung aufweisen. Das heißt, dass eine Verdoppelung der Streifenanzahl den Fehler bei Trapez- und Mittelsumme um den Faktor 4 sinken lassen.

Da die Pixelauflösung des TI-92 Bildschirms nicht gerade berauschend ist, erhält man bei der Trapezsumme häufig Approximationspolygone, die vom Graph der Funktion kaum zu unterscheiden sind.

Der Anzahl der Streifen sind zwar grundsätzlich keine Grenzen nach oben gesetzt. Jedoch bestehen beim *grafisch*en Modus praktische Begrenzungen, die sich aus der Auflösung des Bildschirms ergeben. So sollte für die Methoden 1 bis 6 der Wert für n nicht größer als 20 gewählt werden. Im anderen Modus setzen eigentlich nur Rechenzeit und Speicher Grenzen.

7: Simpson-Verfahren und **8: Pulcherrima[5]:**

Simpson-Verfahren und Pulcherrima waren die bis zur Einführung von Computern mit die am häufigsten verwendeten Näherungsverfahren zum Berechnen von bestimmten Integralen. Bei ihnen wird die Integrandenfunktion in jedem Teilintervall durch eine Parabel bzw. durch ein Polynom 3. Grades approximiert. Dadurch liefern beide Verfahren bei Funktionen bis zum 3.Grad exakte Werte des Integrals.

Für die Darstellung auf dem TI-92 wurden jedoch nicht die Approximationspolynome verwendet, da sie wegen der geringen Auflösung des Bildschirms keine grafische Information geliefert hätten[6]. Der in `integ()` eingeschlagene Weg soll vielmehr die Approximationsformeln dieser beiden Verfahren illustrieren. Diese Formeln lauten in ihrer elementarsten Form, also bei der Einteilung des Integrationsbereichs in nur ein Intervall, bei

Simpson: $$\int_a^b h(x)\,dx = \frac{b-a}{6}\left(h(a) + 4\cdot h(\tfrac{a+b}{2}) + h(b)\right)$$ und bei

Pulcherrima: $$\int_a^b h(x)\,dx = \frac{b-a}{8}\left(h(a) + 3\cdot h(\tfrac{a+2b}{3}) + 3\cdot h(\tfrac{2a+b}{3}) + h(b)\right).$$

Beim Simpson-Verfahren wird deshalb das Teilintervall in drei Abschnitte, deren Breiten sich wie 1 : 4 : 1 verhalten, zerlegt. Bei Pulcherrima erhält man 4 Abschnitte mit Breiten von 1 : 3 : 3 : 1 (siehe die Abbildungen auf Seite 17)[7].

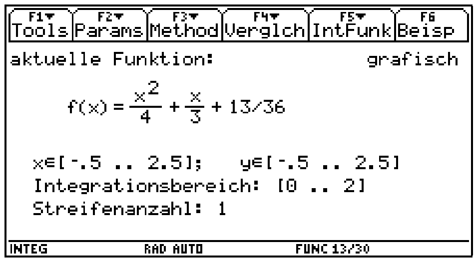
---

5 auch unter dem Namen „Gauß-Quadratur" oder „Gaußsche Regel" bekannt

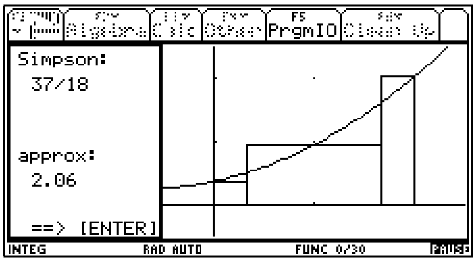
6 siehe im Workshop unter Aufgabe 5.13

7 siehe 6. Bemerkungen zur „Pulcherrima"

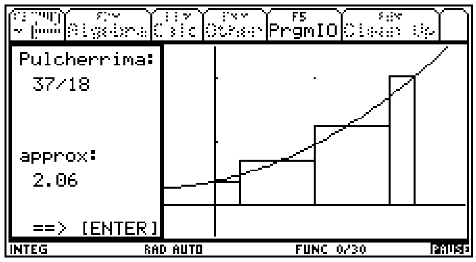
In der *grafisch*en Darstellung wird die vierfache Gewichtung des Funktionswerts in der Intervallmitte beim Simpson-Verfahren bzw. die dreifache Gewichtung an den beiden inneren Stellen in der Pulcherima durch die entsprechende Breite der Streifen ausgedrückt. (Der Funktionsterm wurde aus Gründen der Sichtbarkeit geändert!)



Zugrunde liegende Funktion



Simpson mit n = 1



Pulcherrima mit n = 1

Die obigen Formeln lassen sich im *symbolisch*en Modus erhalten, wenn man als Funktionsterm *h(x)* angibt, den Integrationsbereich von *a* bis *b* laufen lässt und $n = 1$ setzt. Allerdings wird der Ergebnisterm bei der Pulcherrima schon zu umfangreich, so dass er nicht mehr ganz überblickt werden kann.

Bei diesen Methoden erhält man im *grafisch*en Modus auf Grund der Auflösung des Bildschirms die besten Darstellungen, wenn die Streifenzahl *n* den Wert 5 nicht übersteigt. Sonst erscheinen besonders die schmalen Streifen unterschiedlich breit, weil sie beispielsweise einmal 4 ein anderes Mal nur 3 Pixel breit sind.

9: geometrische Folge:

Der Integrationsbereich wird so zerlegt, dass die Längen der Teilintervalle eine geometrische Folge bilden. Um den Fall einfach realisieren zu können, ist er auf Integrationsbereiche, die ganz im Positiven liegen bzw. symbolische Grenzen haben beschränkt. Dies bedeutet keine echte Einschränkung, weil man durch Ersetzen von x im Funktionsterm durch $x - a$, mit geeignetem a, immer zu einem positiven Bereich kommen kann. Bei symbolischen Grenzen ist diese Verschiebung nicht erforderlich. Aber man kann mit dieser Methode auch Funktionen behandeln, bei denen die vorher besprochenen Verfahren nicht zum Ziel führen. Als Beispiel sei die Funktion $f(x) = x^{-1}$ genannt, die bekanntlich auf den natürlichen Logarithmus führt.

A: Monte Carlo:

Mit dieser Methode kann der bekannte Zufallsregen veranschaulicht werden. In ein Rechteck, das durch die Grenzen des Integrationsbereichs sowie durch `ymin` und `ymax` begrenzt ist, fallen zufällig Tropfen. Die Tropfen, die in das Innere der vom Graphen und der x-Achse begrenzten Fläche fallen, werden als Pixel angezeigt und extra gezählt. Das relevante Rechteck ist dick eingerahmt. Als Ergebnis des Verfahrens wird der Wert von Fläche des Rechtecks × $\text{Tropfen}_{\text{innen}}$ / $\text{Tropfen}_{\text{gesamt}}$ ausgegeben. Die Monte-Carlo-Methode liefert nur für positive Funktionen eine Approximation des bestimmten Integrals. In den anderen Fällen wird der vom Graphen und der x-Achse eingeschlossene Flächeninhalt approximiert.

Um bei dieser Methode zu einem einigermaßen anschaulichen Resultat zu kommen, sollte die Tropfenzahl, die hier anstelle der Streifenzahl (s. Menü F2 5) tritt, mindestens 50 sein. Ein entsprechender Hinweis erscheint, wenn diese Zahl unterschritten ist und bietet die Möglichkeit abzubrechen, um die Tropfenanzahl zu erhöhen.

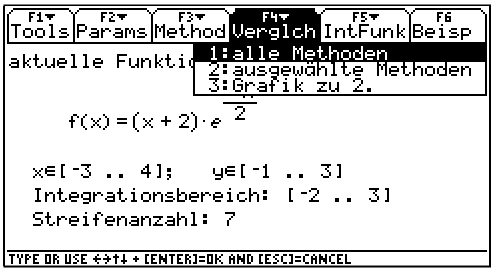
B: Zufallszerlegung:

Das Integrationsintervall wird in n Streifen mit zufälligen Breiten zerlegt. Als Streifenhöhe wird der Funktionswert in der Streifenmitte genommen. Natürlich spiegelt dieses Verfahren keine Riemannsche Summe wieder, weil bei einer Zufallszerlegung niemals sicher-gestellt sein kann, dass die Intervallbreite gegen Null geht. Doch mag gerade dieses Beispiel verwendet werden, um den Fall deutlich zu machen. Andererseits führt die Zufallszerlegung häufig sehr schnell zu erstaunlich guten Approximationen.

Die beiden zuletzt genannten Methoden können nur mit nicht-symbolischen Parametern arbeiten und liefern als Wert immer nur eine dezimale Approximation. Für große Streifen- bzw. Tropfenanzahlen ist der *symbolisch*e Modus zu empfehlen, da die Grafiken aus den oben genannten Gründen nicht sehr schön sind und die Rechenzeit steigt.

2.4 Das Menü F4: `Verglch`

In diesem Bereich werden Verfahren zum Vergleich der im Menü F3 vorhandenen Methoden dargestellt. Aus diesem Grund arbeiten diese Methoden hier, unabhängig von der Einstellung *grafisch* oder *symbolisch*, ausschließlich im *symbolisch*en Mode. Deshalb sollten nur konkrete Funktionen mit numerischen Parametern verwendet werden.

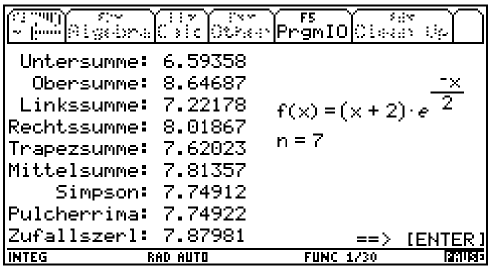


Die Vergleichs-Möglichkeiten

Es ist zwar grundsätzlich auch hier möglich, symbolische Parameter anzugeben, jedoch liefern diese im allgemeinen keine sinnvollen Ergebnisse. Beim Aufruf dieser Verfahren mit symbolischen Parametern wird ein Hinweis gegeben, der den Abbruch der Untersuchung zulässt.

1: alle Methoden:

Die Werte der bei allen Methoden berechneten Summen mit dem aktuellen Integrationsbereich und der aktuellen Streifenzahl werden mit 6 signifikanten Ziffern[8] (im Approximate Mode) übersichtlich auf einer Bildschirmseite ausgegeben. Damit können die im Menü F3 unabhängig voneinander errechneten Werte in einer Zusammenschau verglichen werden.



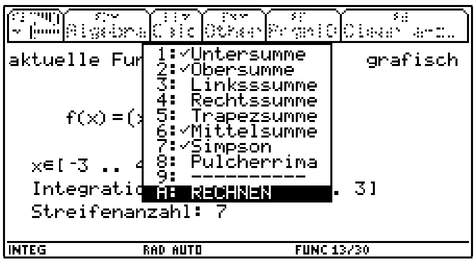
(fast) alle Methoden

Die Monte-Carlo-Methode wird allerdings ausgenommen, weil sie wegen der oben erwähnten unterschiedlichen Bedeutung für Streifen- und Tropfenanzahl nur schlecht in die Systematik passen würde. Die Methode 9 (geometrische Folge) wird nur berücksichtigt, wenn das Integrationsintervall ganz im Positiven liegt.

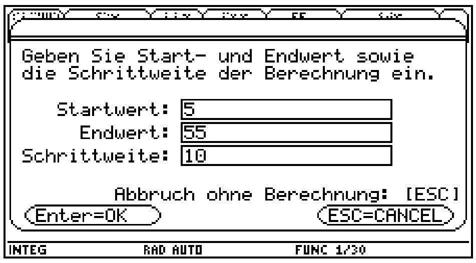
2: ausgewählte Methoden:

Für einzelne Methoden werden die Summen im aktuellen Integrationsbereich, aber mit verschiedenen Streifenzahlen in einem Durchlauf berechnet. Die Ausgabe erfolgt in Form einer Matrix (zugrundegelegt ist die Funktion $f(x) = (x+2)\cdot e^{-\frac{x}{2}}$).

Beim Aufruf öffnet sich zuerst ein Fenster mit den Namen der ersten acht Methoden aus Menü F3. Aus dieser Liste können mit Hilfe der Zifferntasten (oder auch mit



Auswahl der Methoden

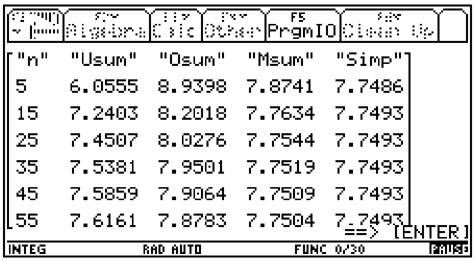


Eingabe des Bereichs

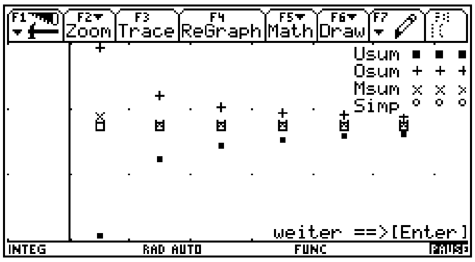
den Cursortasten und ENTER) die gewünschten Methoden ausgewählt werden. Eine gewählte Methode ist jeweils durch ein Häkchen gekennzeichnet. Die Wahl kann durch nochmaliges Drücken der Auswahltaste aufgehoben werden. Zum Abschluss der Auswahl ist die Taste A: RECHNEN zu drücken. Als nächstes wird zur Eingabe der Streifen-

[8] Die 6 Ziffern sind unabhängig von einer mit F1 3 getroffenen Änderung der Standardeinstellung.

anzahl (Anfangs- und Endwert sowie Schrittweite) aufgefordert. An dieser Stelle kann das Verfahren auch noch abgebrochen werden. Anschließend sagt der TI-92, dass er fleißig am Rechnen ist und quittiert jede abgeschlossene Berechnung mit einem Punkt. Damit weiß der Benutzer, dass das Gerät noch aktiv ist. Erst wenn alle Berechnungen „durch” sind, wird die Ergebnismatrix ausgegeben. Die Ausgabe erfolgt mit 4 Nachkommastellen (fix).



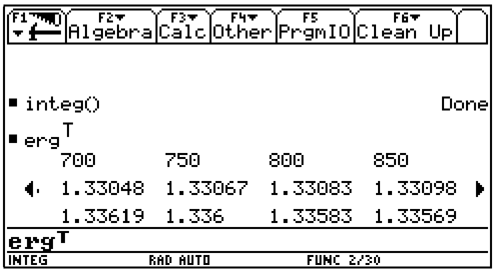
Die Matrix



Grafik zur nebenstehenden Matrix

Bei der Auswahl der Methoden und Festlegung der Streifenzahlen sollte man einerseits beachten, dass dieses Verfahren möglicherweise sehr zeitaufwendig ist, und andererseits daran denken, dass die Ausgabematrix sehr viel Platz beanspruchen kann. Im letzteren Fall werden entsprechende Hinweise gegeben, die auch Korrekturmöglichkeiten offenlassen. Wenn man nicht mehr als 5 Methoden gleichzeitig berechnen lässt und dabei nicht mehr als 6 verschiedene Streifenzahlen vorsieht, reicht der Platz zur Darstellung der Matrix aus. Es kann aber durchaus interessant sein, eine oder zwei Methoden berechnen zu lassen, wobei *n* in einem größeren Bereich, etwa von [50 .. 1000] mit einer Schrittweite von 50 liegt, um beispielsweise das Monotonieverhalten von Unter- und Obersumme zu vergleichen. Ein Inspizieren dieser Matrix wird aber nur möglich sein, wenn man `integ()` mit $\boxed{\text{F1}}$ $\boxed{2}$ $\boxed{\text{ESC}}$ verlässt und dann im Home Screen die Variable `erg`, am besten transponiert, ansieht. Mit Hilfe der Cursortasten kann man durch diese Matrix scrollen. Nötigenfalls setzt man die Stellenzahl des Ausgabeformats höher, um noch die Veränderungen bei großen Streifenzahlen zu erkennen.

Beim Berechnen einer sehr großen Tabelle kann es vorkommen, dass die Automatic-Power-Down-Funktion des TI-92 den Rechner unmittelbar nach der Ausgabe der Tabelle automatisch abgeschaltet. (Der Fall kann natürlich auch an anderer Stelle eintreten, wenn über längere Zeit keine Taste gedrückt wurde.) Dadurch gehen jedoch keine Daten verloren. Man muss nur den Rechner durch Drücken der $\boxed{\text{ON}}$-Taste wieder einschalten und kommt automatisch an die ursprüngliche Stelle zurück.



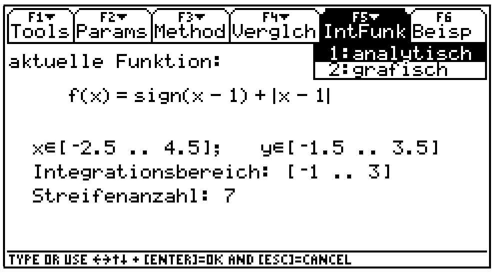
`erg` *für die Funktion* $f(x)=\frac{1}{2}x^2$

$\boxed{3}$: **Grafik zu 2.:**

Eine mit $\boxed{2}$ errechnete Tabelle kann auch in einer grafischen Darstellung ausgegeben werden. Voraussetzung ist, dass die Tabelle mindestens 2 und höchstens 4 Methoden umfasst. Nach rechts werden die Streifenanzahlen, nach oben die Summenwerte aufgetragen. Eine Kalibrierung des Koordinatensystems wird vom Programm vorgenommen, ebenso die Kennzeichnung der Methoden.
Falls die Funktion $\boxed{\text{F4}}$ $\boxed{3}$ aufgerufen wird, ohne dass unmittelbar vorher mit $\boxed{\text{F4}}$ $\boxed{2}$ eine Tabelle errechnet worden war, wird zuerst in den Programmpunkt $\boxed{\text{F4}}$ $\boxed{2}$ verzweigt. Der Ablauf ist dann ebenso wie dort geschildert.

2.5 Das Menü $\boxed{\text{F5}}$: `IntFunk`

Aufgrund der CAS-Fähigkeiten des TI-92 kann auch der Begriff der Integralfunktion, der mit dem ursprünglichen Anliegen des Programmpakets (Riemannsche Summen) wenig zu tun hat, behandelt werden. Integralfunktionen können sowohl in analytischer Form dargestellt als auch gezeichnet werden. Vorausgesetzt ist natürlich, dass eine konkrete Funktion vorliegt und (wegen der grafischen Darstellung) der Zeichenbereich geeignet gewählt ist. Der mit $\boxed{\text{F2}}$ $\boxed{7}$ eingestellte Modus (*grafisch / symbolisch*) ist in diesem Fall nicht relevant.

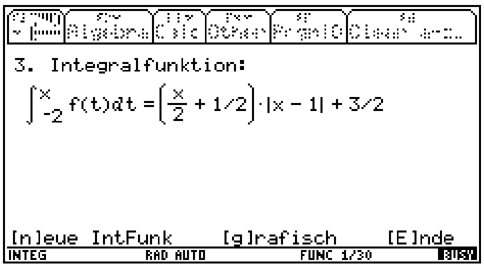


Das Menü $\boxed{\text{F5}}$

Bei beiden Methoden wird zuerst nach einer Untergrenze für die Integralfunktion gefragt. Sie muss numerisch sein.

$\boxed{1}$: **analytisch:**

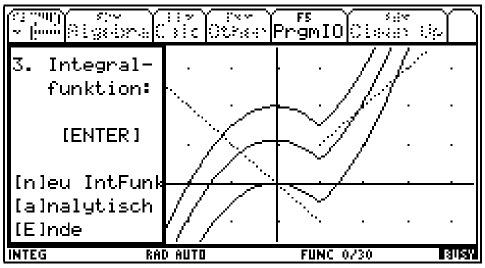
Sofern der TI-92 eine Stammfunktion berechnen kann, wird die Integralfunktion mit der aktuellen Untergrenze als Term in *x* ausgegeben. Anschließend kann eine andere Integralfunktion berechnet, in die grafische Darstellung umgeschaltet oder dieses Verfahren beendet werden. Die Steuerung erfolgt über die entsprechenden Buchstabentasten.



Integralfunktion zu obiger Funktion

$\boxed{2}$: **grafisch:**

Das Fenster wird vertikal im Verhältnis 1:2 geteilt. Im rechten Teil wird zuerst die aktuelle Funktion punktiert gezeichnet. Dann folgen alle bereits ermittelten Integralfunktionen. Die Steuerung erfolgt ebenso wie vorher mit Buchstabentasten. Beim Umschalten in den [a]nalytischen Modus wird dort die zuletzt gezeichnete Integralfunktion angezeigt.



Grafische Darstellung

Die grafische Darstellung der Integralfunktion erfolgt auch dann, wenn keine geschlossene Darstellung der Stammfunktion möglich ist, wenn also die Funktion nicht elementar integrierbar ist. Jedoch läuft dann der Plotvorgang sehr langsam ab, weil der TI-92 für jedes zu plottende Pixel den Wert des bestimmten Integrals neu approximieren muss.

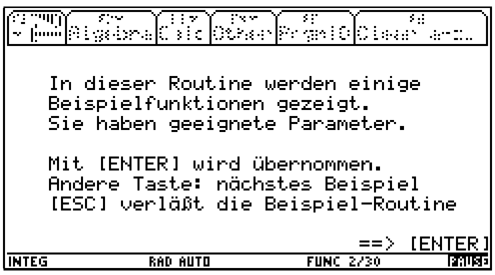
Wie die Screenshots zu diesem Abschnitt zeigen, lassen sich hiermit auch Integralfunktionen von unstetigen Funktionen zeigen. Vor allem in der grafischen Darstellung sind zwei Dinge sehr schön zu sehen:

- Durch Integration werden aus unstetigen Funktionen stetige (wenn auch nicht notwendig differenzierbare) Funktionen. Entsprechend kann man zeigen, dass die Integralfunktionen zu stetigen, aber nicht differenzierbaren Funktionen keine Knickstellen mehr aufweisen, oder vereinfacht ausgedrückt: Integrieren glättet.
- Verschiedene Integralfunktionen gehen durch Parallelverschiebung auseinander hervor. Ihre analytische Darstellung unterscheidet sich jeweils nur durch eine additive Konstante.

2.6 Das Menü $\boxed{F6}$: `Beisp`

In diesem Bereich ist eine Sammlung von acht besonders typischen Beispielfunktionen vorgegeben. Damit kann man sich an die Materie herantasten und erste eigene Erfahrungen mit `integ()` sammeln.

Dem ersten Aufruf der Beispiele in einer Sitzung wird eine kurze Gebrauchsanleitung vorgeschaltet, die bei späteren Aufrufen nicht mehr erscheint. Der Weg durch die Beispiele ist denkbar einfach:



Einleitungsseite zu $\boxed{F6}$

- Drücken einer beliebigen Taste lässt das nächste Beispiel erscheinen. Die 8 Beispiele werden zyklisch durchlaufen.

- Mit [ESC] kann man das Menü [F6] verlassen, ohne die aktuelle Funktion zu ändern.
- Zur Übernahme eines Beispiels als aktuelle Funktion muss man [ENTER] drücken. Der dort vorgeschlagene Zeichen- und Integrationsbereich sowie die Streifenzahl werden zu den aktuellen Werten, die, wie oben beschrieben, geändert werden können, oder die Basis für Untersuchungen darstellen. Da die Beispiele keine symbolischen Parameter aufweisen, wird auch der Betriebsmodus auf *grafisch* eingestellt.
  Der Bildschirmaufbau der Beispielsammlung ist dem aktuellen Bildschirm sehr ähnlich. Deshalb ist eine gewisse Aufmerksamkeit geboten, um zu erkennen, in welchem Programmteil man sich befindet.

Übersicht über die Beispiele:

Term	Plotbereich	Integ.Ber	Streifenzahl	Bemerkungen
$\frac{1}{2}x^2$	$x \in$ [-0,5 .. 2,5] $y \in$ [-0,5 .. 2,5]	[0 .. 2]	7	monoton wachsend, positiv
$-\frac{1}{4}x^3+2$	$x \in$ [-0,5 .. 2,5] $y \in$ [-0,5 .. 2,5]	[0 .. 2]	7	monoton fallend, positiv
$\frac{x^2}{2}-2$	$x \in$ [-0,5 .. 2,5] $y \in$ [-0,5 .. 2,5]	[0 .. 2]	7	monoton wachsend, negativ
$-2x^3+4x^2$	$x \in$ [-0,5 .. 2,5] $y \in$ [-0,5 .. 2,5]	[0 .. 2]	7	nicht monoton; positiv
$\frac{x^2}{2}-1$	$x \in$ [-0,5 .. 2,5] $y \in$ [-1,5 .. 2,5]	[0 .. 2]	7	mon. wachsend, Vorzeichenwechsel
sin(x)	$x \in$ [-π/4..5π/4] $y \in$ [-0,5 .. 1,5]	[0 .. π]	9	Winkelfunktion
sign(x-1)+|x-1|	$x \in$ [-2,5 .. 4,5] $y \in$ [-1,5 .. 3,5]	[-1 .. 3]	7	unstetige Funktion
$\begin{cases} -x;\ \text{falls } x<1 \\ \begin{cases} undef \text{ für } x=1 \\ x \text{ sonst} \end{cases} \end{cases}$	$x \in$ [-2,5 .. 4,5] $y \in$ [-1,5 .. 3,5]	[-1 .. 3]	6	gleiche Funktion wie vorher, jedoch mit `when`-Anweisung deklariert; (s. Kap. 4)

3 Die Programmstruktur

In diesem Abschnitt wird zuerst gezeigt, wie die 25 Module des Programmpakets zusammenwirken. Dann werden verschiedene Methoden zum Betrachten und Studieren des Quellcodes erläutert. Schließlich folgen Hinweise darauf, wie Anwender das Programm verändern und nach ihren speziellen Bedürfnissen modifizieren können.
Wer, insbesondere beim ersten Lesen, mit diesen drei Punkten nichts im Sinn hat, kann diesen Abschnitt ohne Probleme für das Verständnis des übrigen Texts übergehen.

Die Programme:

Das Hauptprogramm `integ()` erfüllt zwei Aufgaben:
Auf der einen Seite stellt es in einem Konstrukt `Toolbar .. EndTBar` das Hauptmenü von `integ()` bereit, auf der anderen Seite wird von hier aus der Aufruf der einzelnen Module gesteuert. Ein kleiner Ausschnitt aus den ersten Zeilen des Programms zeigt diese Struktur:

```
Quellcode                                    {ggf. Kommentare}
integ()
Prgm
basis()              {initialisiert und sichert den Status vor Programmstart}
ClrDraw:ClrGraph
If nn Then:2»dar:Else:1»dar:EndIf            {vorerst unwichtig}
Lbl e0               {bei diesem Label wird immer eingesprungen}
screen()             {dient der Aufbereitung des Bildschirms}
 ToolBar
  Title "Tools"      {Das Menü F1:Tools; verzweigt zu Labels t1 ... t4}
   Item "Hilfe",t1
   Item "Ende",t2
   Item "Einstellungen",t3
   Item "÷ber INTEG",t4
  .....
  Title "Method"     {Das Menü F3:Method; verzweigt zu Labels m1 ... m11}
   Item "Untersumme",m1
   Item "Obersumme",m2
   Item "Linkssumme",m3
   Item "Rechtssumme",m4
  .....
  Title "Beisp",b    {Das Menü F6:Beisp; verzweigt zum Label b}
 EndTBar
Goto e0              {Sprung zum Label e0}
  ....
  ....
```

Von vielen Labels aus wird nur eines der Module aufgerufen und nach dessen Ausführung wieder zum zentralen Label `e0` zurückverzweigt:

`Quellcode` (Labels zu Menü F1:Tools)	{ggf. Kommentare}
`Lbl t1:hilf():Goto e0`	
`Lbl t2:ende():Return`	{nach `ende()` schließt `integ()`}
`Lbl t3:einst():Goto e0`	
`Lbl t4:ueber():Goto e0`	

Bei anderen Labels (insbesondere denen aus dem Methoden-menü - s. folgendes Listing) laufen mehrere Prozesse ab:
Wenn schon eine Funktion deklariert ist, wird die Prozedur `dat()` durchlaufen, welche die Parameter für die gewählte Methode bereitstellt. Sonst wird zum schon bekannten Label `e0` zurück verzweigt.
Wenn `dar=1`, d.h. die Darstellungsart `grafisch` ist, wird das Plotten in der Prozedur `box(...)` erledigt. Auf jeden Fall wird aber in `sums(p_)` die relevante Summe berechnet und in `aus1(p_)` die Ausgabe gesteuert.

`Quellcode` (zum Label `m3`, d.h. F3 3) {ggf. Kommentare}

```
Lbl m3
If getType(f_)="FUNC" Then:dat()
Else:Goto e0:EndIf
If dar=1:box(xl,xr,limit(f(t),t,xl,1)
sums(3):aus1("Linkssumme:"):Goto e0
```

Nun folgen wissenswerte Details zu den einzelnen Modulen in alphabetischer Reihenfolge:

`aus1(p_)` erledigt die Ausgabe bei den Methoden. In `p_` wird der Name der Methode übergeben. In der Darstellungsart `grafisch` (`dar=1`) werden nur die 4 Zeilen in der linken Hälfte des Split Screen mit entsprechenden `output` Anweisungen geschrieben.
Im anderen Fall wird der exakte Term im Full Screen ausgegeben und die Möglichkeit zum Bearbeiten eröffnet. Der Verteiler für die 13 Alternativen (von `factor(` bis `Abbruch`) liegt in dem auf eine Warteschleife folgenden `PopUp`-Menü. Die einzelnen Teilroutinen sind einfach zu durchschauen.

`aus2(p_)` steuert die Ausgabe beim Vergleich aller Methoden (F4 1). In `p_` wird der Name der Methode übergeben. `zl` reguliert den Zeilenabstand, je nachdem die Ausgabe mit der oder ohne die Methode „geometrische Folge“ kommt.

`basis()` sichert die aktuellen `Mode`-Einstellungen und setzt die für den einwandfreien Ablauf des Programms erforderlichen Modi. (Z.B. als Standard für `Display Digits`: `Float 3`.) Weiters werden einige Variable gelöscht, andere hingegen initialisiert. Dies alles erfolgt nur, wenn die Variable `θ_w`, die anzeigt, ob das Programm über F1 2 ESC verlassen wurde, nicht `true` ist.

`beisp()` liefert acht Beispielfunktionen. Für jedes Beispiel werden zuerst alle Parameter ter vom Funktionsterm bis zur Streifenzahl als Strings dargestellt. Dann wird die lokale Routine `aus(p,t)` aufgerufen, die diese Parameter in entsprechender Form auf dem Bildschirm plaziert. In `p` wird die Nummer des Beispiels, in `t` ein knapper erläuternder Text übergeben. Je nach Reaktion des Anwenders (ENTER, ESC oder sonstige Taste) wird zu den Labels `uebn` (für Übernehmen), zu `ende` oder zum nächsten Beispiel verzweigt. Im Block `uebn` werden die als Strings dargestellten Parameter mit der `expr(`-Funktion an Variable zugewiesen. Im Block `ende` werden überflüssige (quasilokale) Variable entfernt.

`box(x1,x2,h1)` zeichnet die Rechteckstreifen bei den meisten Methoden. Zuerst wird die Prozedur `sm()` aufgerufen (s. dort). Dann werden in einer Laufanweisung mit der Laufvariablen `k` zwei senkrechte Strecken und die horizontale Verbindung ihrer Endpunkte gezeichnet. Die Variable `k` wird mit den Parametern `x1` und `x2` (s. auch Prozedur `dat()`) übergeben und in der Laufanweisung mit konkreten Werten versehen. Dem Parameter `h1` wird beim Aufruf der Prozedur immer ein (rechtsseitiger) Grenzwert übergeben. Damit wird erreicht, dass die Rechteckstreifen auch dann gezeichnet werden, wenn Teilpunkte auf Definitionslücken der Funktion fallen.[9]

Das Konstrukt `Try .. EndTry` dient dazu, einen Programmabbruch zu verhindern, wenn `x1`, `x2` oder `h1` nicht vom Typ `"NUM"` sind.

Die Prozedur `box(...)` wird nur vom Hauptprogramm aus aufgerufen.

`dat()` ist eine für alle Methoden, außer den beiden Zufallsverfahren zentrale Prozedur. In ihr werden die Streifenbreite (`d_`) und linker bzw. rechter Intervallrand (`xl` und `xr` bzw. `xgl` und `xgr`) als Ausdrücke mit der Variablen `k` dargestellt. Auf sie wird dann beim Aufruf der Prozeduren wie `box(x1,x2,h1)`, `simp(x1,x2)`, `sums(p_)`, etc. zugegriffen

`einst()` bietet die Möglichkeit, einige Standardeinstellungen zu ändern. Dieses kurze Programm gibt eine schöne Gelegenheit zu studieren, wie man sich selbst modifizierende Menüs „basteln” kann.

`ende()` ist die Ende-Routine, mit der `integ()` verlassen werden soll (bzw. muss!!). Wenn sie über F1 2 ESC verlassen wird, wird die Variable θ_w auf `true` gesetzt. Andernfalls werden die ursprünglichen `Mode`-Einstellungen restauriert und alle denkbaren Variablen werden gelöscht.

`fein(p_)` wird über das `Params`-Menü F2 mit Werten 1 bis 5 für `p_` aufgerufen. Im ersten Teil werden die verlangten Parameter mit der `request`-Anweisung in String-Variable eingelesen. Damit wird unterschieden, ob alle Parameter (F2 1) oder z.B. nur die Streifenzahl (F2 5) geändert werden sollen.

[9] Man kann anstelle des Grenzwerts auch den entsprechenden Funktionswert übergeben. In den meisten Fällen, d.h. bei „anständigen“ Funktionen, arbeitet `integ()` auch dann korrekt.

Ab dem Label `ende` erfolgt die Eingabeprüfung. Diese Prüfungen sind ziemlich aufwendig und teilweise auch trickreich: Um abzufragen, ob der Funktionsterm symbolisch ist, wird ein Funktionswert an einer völlig willkürlich gewählten, „krummen" Stelle - bei $x = 1{,}3719$ - untersucht. Dadurch werden reelle Funktionen mit einer Definitionslücke an dieser Stelle als symbolisch betrachtet. Wer damit nicht leben kann, muss das Programm an dieser Stelle ändern. (Den Autoren ist jedenfalls noch keine Funktion mit einer Definitionslücke bei 1,3719 unter die Finger gekommen.)

Weiter oben wurde dargelegt, dass symbolische Parameter nur gewisse Namen haben dürfen (*a*, *b* oder *n*). Beim Betrachten dieser Prüfroutine wird klar, welchen Aufwand es bedeutet hätte, einerseits diese Namen weitgehend freizugeben und andererseits dafür zu sorgen, dass keine Kollision mit globalen Variablen auftreten kann.

`hilf()` liefert die Hilfetexte, die mit [F1] [1] aufgerufen werden. Die Prozedur greift auf 5 lokale Unterprogramme `h1()` bis `h5()` zu, in denen die eigentlichen Texte enthalten sind. Die Steuerung übernimmt eine Hautptroutine (ganz am Ende des Listings), die ein eigenes Menü aufbaut (s. Abb. auf Seite 7) und die Rückkehr zu `integ()` organisiert.

`intfunk(p_)` wird vom Hauptprogramm direkt aufgerufen und erledigt den Menüpunkt [F5]:`IntFunk` vollkommen selbständig. Der Parameter `p_` steuert, ob die Integralfunktion analytisch oder grafisch dargestellt wird.

Eingangs wird der Funktionsterm `f_(x)` in eine Funktion `g_(t)` umgewandelt und `f_(x)` mit dem Zeichenstil „punktiert" an `y1(x)` zugewiesen. Hochsetzen des Zählers `i` um 1 (von Null beginnend) und die Eingabe der Untergrenze erfolgen vom Label `unt` an. Dann berechnet das Programm die Integralfunktion $\int_u^x g\_(t)\,\mathrm{d}t$ und weist den Wert einem Term `fi(x)` zu. Durch die Anweisung

```
expr("define y"&string(i+1)&"(x)="&string(fi(x)))
```

wird sie als *i*+1-te Funktion in den Funktionen-Editor geschrieben und kann bei der grafischen Betrachtungsweise leicht gezeichnet werden. (Wegen dieser Zuweisung sollten Sie, falls Sie auf Integralfunktionen aus sind, vordefinierte Funktionen im [Y=]-Editor nur mit höheren `yi(x)`-Namen belegen.)

Der Block, der mit `If p_=1 Then` beginnt, liefert dann die Ausgabe der Integralfunktion in analytischer Form im Full Screen (`Lbl num`) oder als Graph im geteilten (= Split-) Screen (`Lbl gra`). In beiden Fällen erfolgt eine Verzweigung zum Label `ausw`, einer Warteschleife, in der die Reaktionen des Anwenders ausgewertet werden. Beim Beenden des Moduls `intfunk(p_)` (`Lbl ende`) werden die im [Y=]-Editor angelegten Funktionen sowie weitere Variable gelöscht.

`pul(x1,x2)` erledigt das Zeichnen bei der Pulcherrima-Methode. Das Prinzip dieser Prozedur wurde schon bei `box(x1,x2,h1)` erläutert. Die lokalen Variablen `h1` bis `h4` und `d1` sind selbsterklärend. Sie wurden aus Gründen der Beschleunigung eingeführt.

`screen()` baut zwei verschiedene Arten von Bildschirmen auf: Wenn noch keine Parameter erfasst sind, wird der Eröffnungsbildschirm, sonst der aktuelle Bildschirm erzeugt Die Prozedur `screen()` steht im Hauptprogramm zwischen den Anweisungen `Lbl e0` und `Toolbar`. Sie ist damit die am häufigsten aufgerufene Prozedur des gesamten Programmpakets.

`simp(x1,x2)` ist für das Zeichnen des Simpson - Verfahrens zuständig. Die relevanten Bemerkungen stehen bei `pul(x1,x2)`.

`sm()` wird von den Prozeduren, die im *grafisch*en Modus Zeichenarbeit verrichten (wie z.B. `box` oder `pul`) gerufen, um den Split Screen einzurichten und den Graph der Funktion zu plotten.

`sums(p_)` berechnet für alle Methoden, außer der Monte-Carlo-Methode und der Zufallszerlegung, die relevante Summe exakt und legt sie in der Variablen `erg` ab. Dabei wird ebenso wie in den Zeichenprozeduren `box(...)` etc. anstelle mit Funktionswerten mit Grenzwerten gearbeitet. Das dort in der Fußnote Gesagte gilt hier entsprechend.

`tra(x1,x2)` zeichnet das Trapez-Verfahren. Das Prinzip wurde bei `box(x1,x2,h1)` beschrieben.

`ueber()` ist eine linear durchlaufene Routine, die von [F1] [4] aufgerufen wird.

`vgl()` ist das Modul, in dem der Vergleich ausgewählter Methoden bewerkstelligt wird ([F4] [2]). In den ersten Zeilen erfolgen Vorbelegungen. In dem nach `Lbl v` folgenden `PopUp`-menü werden die Methoden zur Auswahl präsentiert und anschließend wird die Wahl verarbeitet. Erst wenn [A]:`RECHNEN` gewählt wird, – `v_` hat dann den Wert 10 – wird bei `Lbl vv` fortgesetzt. Nun wird der Bereich der Streifenzahlen abgefragt und ggf. wird auf Dimensionierungsprobleme bei der Ausgabe verwiesen. Anschließend werden in der Ausgabematrix `θmat` die 1. Zeile und 1. Spalte mit Beschriftungen versehen und in einer geschachtelten Laufanweisung unter Verwendung der Prozedur `sums(p_)` die Werte berechnet. Die Ausgabe erfolgt im Format `FIX 4`, d.h. mit 4 Nachkommastellen. Eine Bemerkung zur Variablen `fl`, die in der 6. Zeile von unten abgefragt wird: Sie wird in `vgl_graf()` verwaltet und unterdrückt die Ausgabe der Matrix, wenn `vgl()` von dort nur zu ihrer Berechnung aufgerufen wurde.

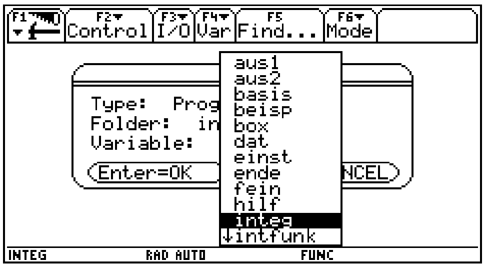
`vgl_alle()` zeigt die Summen (fast) aller Methoden bei gegebener Streifenzahl in einer übersichtlichen Zusammenschau. Für jede Methode wird mit `sums(p_)` der Wert errechnet und mit `aus2(p_)` ausgegeben.

`vgl_graf()` veranschaulicht die in `vgl()` errechnete Matrix grafisch. Wenn die Variable

`erg` (sie enthält immer das Resultat der letzten Berechnung) keine Matrix ist, muss zuerst das Modul `vgl()` durchlaufen werden. Dann werden aus dieser Matrix die Werte für die Streifenanzahl in Liste `l1` und je Methode die Summenwerte in Listen `l2`, `l3`, ... extrahiert und abgelegt. Gleichzeitig werden Scatterplots (Streudiagramme) generiert:

```
NewPlot i-1,1,l1,#("l"&string(i)),,,,6-i
```

Sie verwenden in *x*-Richtung die Liste `l1`, in *y*-Richtung jeweils die Listen `l2`, `l3`, ..., und für die Markierung der Punkte finden nacheinander die Zeichen ■, +, × und □ (s. TI-92 Handbuch, S. 420) Anwendung. In den Variablen `t1`, `t2`, ... werden die Legenden für die Plots gespeichert. Anschließend wird das Koordinatensystem geeignet dimensioniert (und natürlich die aktuelle Einstellung gesichert). Falls nur ein Wert der Ergebnismatrix nicht numerisch ist, wird die Prozedur mit einem entsprechenden Kommentar beendet.



Fenster zur Programmauswahl

`w1(p_)` setzt die Zeichenfolge "⟹ [ENTER]" im Split bzw. Full Screen an die richtige Stelle und schaltet die `pause`-Prozedur ein, die nur mit [ENTER] verlassen werden kann.

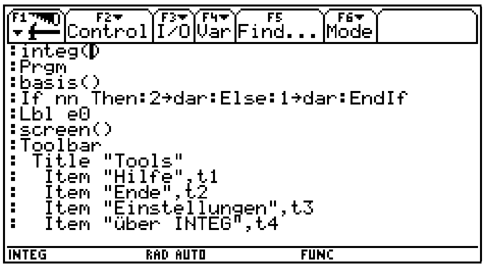
`zufreg()` stellt die Monte-Carlo-Methode dar. Eingangs wird nötigenfalls auf eine zu geringe Tropfenzahl verwiesen. Dann werden, wie bei den anderen Methoden, die Prozeduren `dat()` und `sm()` durchlaufen und der fragliche Bereich wird mit einer doppelt dicken Linie umrandet. Der „Regen” fällt in der Laufanweisung mit der Laufvariablen `k`. Die Approximation der Fläche erfolgt in der 5. Zeile von unten. (Wenn die Untergrenze des Integrationsbereichs größer als die Obergrenze ist, werden diese Grenzen temporär vertauscht, um dennoch einen positiven Flächeninhalt zu erreichen.)

`zufzer(p_)` setzt die Methode der Zufallszerlegung um. Eine Liste `lran` mit zufällig gewählten Werten aus dem (abgeschlossenen) Integrationsintervall wird gebildet und geeignet sortiert. Wenn die Prozedur mit `p_=true` aufgerufen wurde (d.h. aus dem Menü [F3] – beim Aufruf aus [F4] [1] ist `p_=false`) und die *grafisch*e Darstellungsart aktuell ist, wird mit diesen Teilpunkten das Streifenmuster als eine Art Mittelsumme gezeichnet. Die Summe der Streifen wird in `erg` abgeliefert.

Inspizieren des Quellcodes

Um einen Einblick in die Programmstruktur zu erhalten, muss man sich den Quellcode der einzelnen Module ansehen. Dies lässt sich auf drei verschiedenen Wegen verwirklichen:

- Mit dem TI-92 vor sich kann man den Quelltext im Programmeditor einsehen. Dazu öffnet man zuerst mit [APPS] [7] [2] das `OPEN`-Fenster des Program Editors. Mit ⊙ ⊙ ⊙ zeigt sich eine Liste der Programme, aus der man das gewünschte mit Cursorbewegung aussuchen kann. Zweimaliges [ENTER] holt den Quellcode in den Programmeditor. Hier kann er eingesehen, aber auch geändert werden. (Siehe dazu die Abbildungen hier und auf der nächsten Seite.)

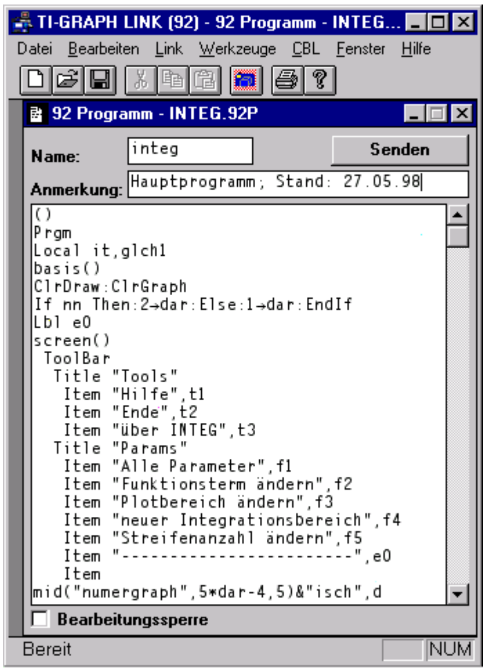


Die ersten Zeilen von `integ()`

Durch den Quelltext kann man mit den Cursortasten ↓ und ↑ scrollen. Will man seitenweise blättern, muss jeweils ein [2nd] vorgeschaltet werden (also [2nd] ↓ bzw. [2nd] ↑). Der Cursor kann mit → bzw. ← durch die Zeilen geschoben werden, wobei [2nd] → zum Zeilenende und [2nd] ← an den Zeilenanfang führt.

Aber **seien Sie <u>vorsichtig</u>**: Der TI-92 akzeptiert jeden Tastendruck im Editor und verändert damit den Quellcode. Wenn Sie den Programmeditor verlassen (zB indem Sie mit [◆] [HOME] in den Home Screen schalten) oder wenn Sie einen anderen Quellcode einsehen, wird die Änderung **ohne Rückfrage** gespeichert. Ob das Programm nach einer versehentlichen Änderung noch die gewünschten Resultate zeigt oder ob es überhaupt noch zum Laufen kommt, hängt dann von Ihrem Tagesglück ab. (Aber selbstverständlich kann ein versehentlich geänderter Modul mit dem *GraphLink* wieder hergestellt werden.)

- Eine Methode mit größerer Sicherheit bietet das TI-*GraphLink*. Nachdem die Programme auf der beigelegten Diskette verfügbar sind, können sie in das Editierfenster des *GraphLink* geladen werden. Dort kann man mit den üblichen *Windows*-Methoden durch den Text scrollen. Änderungen werden nur gespeichert, wenn man dies ausdrücklich verlangt. (Für die Eingabe von Sonderzeichen, Umlauten etc. muss zusätzlich das Fenster Funktionsliste geöffnet werden.)
  Ein Mangel von *GraphLink* ist, dass die Größe des Editierfensters nicht veränderbar ist. Man hat nur maximal 22 Zeilen im Blickfeld. Das ist aber doch deutlich mehr als die 12 Zeilen im Editorfenster des TI-92, außerdem ist das Editieren einfacher.



Editierfenster des GraphLink

- Die sicherste Methode zum Einsehen des Quellcodes ist - auch wenn das Zeitalter des papierlosen Büros begonnen haben sollte – ein Listing auf Papier.

Denn ein Blatt Papier – und die Listings der meisten Module überschreiten eine Seite nicht – ist übersichtlich und erträgt Programmänderungen mit Bleistift ohne Murren. Wenn man eine Sicherungskopie des ursprünglichen Moduls angelegt hat, lässt sich die Änderung ohne größeres Risiko durchführen. Den Ausdruck eines Listings erhält man am einfachsten aus dem Editierfenster des *GraphLink* heraus, wenn man das Drucker-Icon anklickt.

Programmänderungen

Wer sich an das Vorhaben wagt, das Programm zu ändern, sollte auf jeden Fall zuvor eine Sicherungskopie des ursprünglichen Programms angelegt haben. Ein probates Mittel bietet *GraphLink* mit dem Gruppieren. Im Menü „Werkzeuge" steht als erster Punkt: „TI-92-Dateien gruppieren..." Wenn man ihn auswählt, öffnet sich ein Fenster, ähnlich dem zum Senden von Dateien. Tauft man beispielsweise die Originalgruppe `integ00.92g`, so können folgende Versionen entsprechend durchnummeriert werden, und man behält die Übersicht.

Seitens der Autoren werden im Folgenden drei Vorschläge zu Programmänderungen gemacht. Da der Phantasie keine Grenzen gesetzt sind und die Neugier der Autoren groß ist, bitten sie jedoch darum, dass Anwender, die größere Änderungen vornehmen, davon berichten. Sicherlich kann so das gesamte Programm ein gewisses Eigenleben entwickeln und im Lauf der Zeit optimiert werden.

- Einzelne Module entfernen

  Wer eine gewisse Erfahrung mit dem Programm hat, wird auf die Online-Hilfe (F1 1) verzichten können. Auch das Modul `ueber()` (F1 3) enthält nichts, was für den Betrieb des Programms wichtig ist. Man kann also die Module `hilf()` und `ueber()` einfach löschen. Allerdings muss auch im Hauptprogramm `integ()`der Aufruf dieser Module unmöglich gemacht werden. Im nachfolgenden Ausschnitt sind die zu löschenden Aufrufe unterstrichen dargestellt.

`Quellcode` (Labels zu Menü F1:Tools in `integ()`)	{ggf. Kommentare}
`Lbl t1:`<u>*`hilf():`*</u>`Goto e0`	{neu: `Lbl t1:Goto e0` }
`Lbl t2:ende():Return`	
`Lbl t3:einst():Goto e0`	
`Lbl t4:`<u>*`ueber():`*</u>`Goto e0`	{neu: `Lbl t4:Goto e0` }

  Nach dem Öffnen des Menüs F1 sind zwar die Unterpunkte `1:Hilfe` und `4:über INTEG` noch vorhanden, bei ihrer Wahl wird jedoch keine (nach außen sichtbare) Aktion ausgelöst. Diese Maßnahme spart ca. 5 KB Speicherplatz des TI-92.

- Beispiele ändern/einbauen

  Das folgende Listing zeigt das 3. Beispiel aus dem Modul `beisp()`:

`Quellcode` (3. Beispiel aus `beisp()`)	{ggf. Kommentare}

```
ClrIO ¨ 3.Beisp.
 "1/2x^2-2"»fu
 "ª.5"»xl:"2.5"»xr:"ª2.5"»yu:"0.5"»yo
 "0"»gru:"2"»gro:"7"»nri
 aus(3,"(mon. wachs., negativ)"):If k=13:Goto uebn
If k=264:Goto ende
```

  Man erkennt sofort, dass allen Variablen Strings zugewiesen werden. Der Funktionsterm an `fu`, die Grenzen des Plotbereichs an `xl`, `xr`, `yu` und `yo`, die Integrationsgrenzen an `gru` bzw. `gro` sowie die Streifenzahl an `nri`. Wenn man ein Beispiel ändern oder ein neues Beispiel einfügen will, muss man nur diese acht Strings entsprechend ändern. Doch Vorsicht: Das Programm prüft an keiner Stelle, ob diese Werte einwandfrei sind, sondern stürzt möglicherweise an einer Stelle ab, an der die Ursache nur schwer zu finden ist.

  Anschließend erfolgt der Aufruf der lokalen Prozedur `aus(p,t)`, an die als 1. Parameter die Nummer des Beispiels und als 2. Parameter eine knappe Charakterisierung des Beispiels (max. 24 Zeichen) übergeben wird. `aus(p,t)` bereitet dann den Bildschirm auf und verweilt in einer Warteschleife, die den Parameter `k` zurückgibt, der den weiteren Ablauf bestimmt.

- Direktes Aufsuchen von Beispielen

  Es mag auf die Dauer langweilig sein, beim Aussuchen eines Beispiels immer die ganze Liste aller Beispiele „durchzuradeln”. Da ohnehin nach der Präsentation jedes Beispiels über das Programm eine Tastaturabfrage erfolgt, kann man dies ausnützen, um Beispiele direkt anzusteuern.

  Dazu müssen bei jedem Beispiel nur zwei Anweisungen ergänzt werden (wobei auf obiges Listing Bezug genommen wird):

  - Vor jedes Beispiel muss ein Label gesetzt werden, das die Nummer des Beispiels enthält. ZB vor dem `ClrIO` des 3. Beispiels müsste stehen: `Lbl bsp3`
  - Nach der Zeile `If k=264:Goto ende` muss eine weitere Zeile eingefügt werden:

```
If k>48 and k<57:Goto #("bsp"&string(k-48))
```

  Der Programmablauf geht nun wie folgt:

In der lokalen Prozedur `aus(p,t)` wird ein Beispiel präsentiert und dann die Tastatur abgefragt. Nach Drücken einer Taste wird geprüft:

- [ENTER] (d.h. `k=13`): Sprung zum Label `uebn`
- [ESC] (d.h. `k=264`): Sprung zum Label `ende`
- [1] ... [8] (d.h. `k`∈`{49` ... `56}`): Mit der „Umleitungs"-Anweisung # wird ein Label der Form `bsp1` ... `bsp8` gebildet und dorthin gesprungen.
- Bei allen anderen Tasten erfolgt die Anzeige des nächsten Beispiels.

Natürlich müsste die Hinweiszeile (`Output 90,60,"[ESC] [ENTER] [sonst]")` in `aus(p,t)` und der Text beim 1. Aufruf von `beisp()` entsprechend modifiziert werden. Aber wer sich an diese Änderung macht, wird ohnehin nicht ruhen, bis das Programm „schön" ist.

Teil II Workshop

In diesem Workshop werden Ihnen einige Aufgabenstellungen vorgestellt, die mit Hilfe von `integ()` behandelt werden können. Sie stammen von Workshops, die im Rahmen von Lehrerfortbildungsveranstaltungen in vielen Ländern veranstaltet wurden, aber auch aus der schulischen Praxis der beiden Autoren. Viele Aufgaben eignen sich nach Meinung der Autoren vorzüglich als mehr oder weniger offene Themen oder Arbeitsaufträge für Schüler oder Schülergruppen. Die angebotenen Bearbeitungsvorschläge sollen nur als mögliche Lösungsansätze verstanden werden und zu eigenen Aufgabenstellungen inspirieren. Die Autoren würden sich über Rückmeldungen und Erfahrungsberichte sehr freuen.

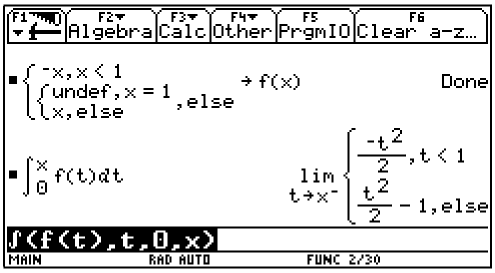
Hinweis: Der „Smiley“ ☺ steht vor einer Aufgabe oder einem Arbeitsauftrag, während das ❑-Zeichen einen zusätzlichen Kommentar hervorheben soll.

4 Abschnittsweise definierte Funktionen

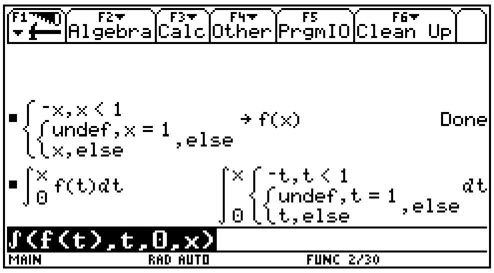
Mit den CAS-TI können abschnittsweise definierte Funktionen auf verschiedenen Wegen deklariert werden. Das 7. Beispiel zeigt den Fall einer unstetigen Funktion, die sich mit Hilfe der Funktionen `sign(` und der `abs(` in einem geschlossenen Ausdruck darstellen lässt. Auf den ersten Blick scheinen alle Methoden wie bei anderen „ordentlichen” Funktionen zu arbeiten. Allerdings fehlt bei der Mittelsumme ein Streifen in der Zeichnung und der Wert der Summe enthält `sign(0)`. Ändert man die Streifenzahl auf einen geraden Wert (zB 6), so entdeckt man den umgekehrten Effekt: Bei allen Methoden außer der Mittelsumme fehlen Streifen und die Summe wird nicht korrekt ausgewertet. (Am besten sieht man diesen Effekt, wenn man vorher die Koordinatenachsen mit $\boxed{\text{F1}}$ $\boxed{3}$ $\boxed{3}$ ausgeschaltet hat.)
Ursache für dieses Verhalten ist die (etwas eigenartige) Definition der `sign(`-Funktion bei den CAS-TI: Sie hat bei $x = 0$ nicht, wie in der üblichen Literatur beschrieben, den Wert 0, sondern ist dort nicht definiert.
Eine andere Lösung bietet die `when`-Anweisung (s. Beispiel 8). Obwohl auch hier die Funktion an der Stelle $x = 1$ eine Definitionslücke aufweist[10], werden die Grenzwerte in den Prozeduren `sums`, `box`, `tra`, etc. anders ausgewertet. Die mit $\boxed{\text{F3}}$ und $\boxed{\text{F4}}$ aufrufbaren Verfahren arbeiten hier also einwandfrei.

Leider hat die Deklaration von abschnittsweise definierten Funktionen mit der `when`-Anweisung beim TI-92 PLUS einen gravierenden Nachteil: Offensichtlich werden die Integrale solcher Funktionen unterschiedlich bzw. überhaupt nicht ausgewertet. Die beiden folgenden Screenshots zeigen diesen Unterschied:

[10] Die Lücke bei $x = 1$ ist eigentlich nur erforderlich, um bei der Zeichnung getrennte Geradenstücke zu erhalten.



TI-92 bzw. TI-92 II



TI-92 PLUS

Die Folge dieses Unterschieds ist, dass die Funktion F5 `:IntFunk` beim PLUS-Modul mit der `when`-Klausel nicht zufriedenstellend arbeitet. Da die Integralfunktionen nicht berechnet werden, ist auch ihre grafische Darstellung alles andere als befriedigend.

5 Ausgewählte Aufgaben

5.1 Von den Grenzkosten zu den Gesamtkosten

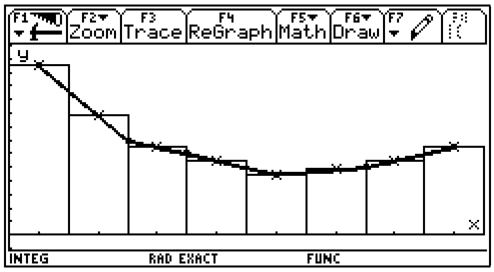
Die Grenzkosten – das sind die i.a. veränderlichen Produktionskosten für die jeweils nächste Produktionseinheit – werden von der Kostenrechnungsstelle eines Betriebes für unterschiedliche Produktionsmengen ermittelt.

Menge x	5	20	30	45	60	75
Grenzkosten *GK*	125	70	60	45	50	65

☺ Übertragen Sie die Punkte in ein geeignetes Koordinatensystem.

☺ Lässt sich an der Lage der Punkte eine Gesetzmäßigkeit erkennen?
Verbinde die Punkte zu einem geschlossenen Graphen (Kurve oder Polygonzug!)

Berechnen Sie näherungsweise die variablen Gesamtkosten für die ersten 80 Mengeneinheiten, indem Sie in Abständen von je 10 Mengeneinheiten die jeweils mittleren Grenzkosten näherungsweise als Stückkosten für diesen ganzen Abschnitt annehmen. Tragen Sie diese mittleren (= durchschnittlichen) Grenzkosten in die Zeichnung ein. Die Gesamtkosten K ergeben sich natürlich aus *Menge* × *Stückkosten* – vorerst bleiben Fix- oder Stillstandskosten unberücksichtigt.



Ihre Handzeichnung könnte dann ungefähr so aussehen wie am TI-92 simuliert. (Hier wurde mit kubischen Splines „geschwindelt“!)

Die Gesamtkosten der ersten 80 Mengeneinheiten lassen sich als die Summe von insgesamt 8 Rechtecksflächen deuten:

$K = 10 \times GK(5) + 10 \times GK(15) + 10 \times GK(25) + \ldots\ldots\ldots\ldots + 10 \times GK(75) \approx 5467$

Die Werte für die Grenzkosten können näherungsweise vom Graphen abgelesen werden.

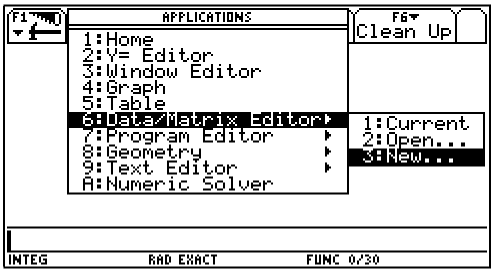
Diese näherungsweise Berechnung wird noch ziemlich grob sein. Eine Verbesserung des mathematischen Modells wäre in zweifacher Hinsicht möglich:

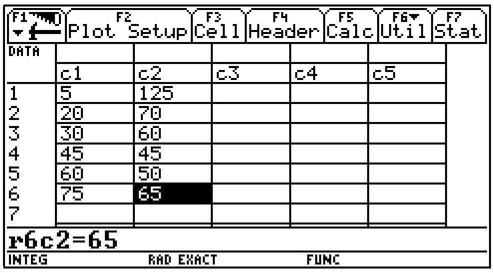
- Ein geeignetes Modell der Grenzkostenfunktion macht das Zeichnen des Graphen und das Ablesen der Funktionswerte überflüssig.
- Kleinere – aber dafür mehr – Intervalle gleichen die Unterschiede am Anfang und Ende der Abschnitte wesentlich besser aus.

Beide Ideen sind durchführbar, erfordern aber einige Rechenarbeit, die man getrost dem CAS überlassen kann.

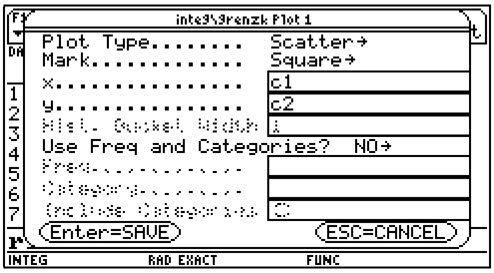
Stellen Sie die Datenpunkte auf dem TI-92 dar und verwenden Sie den Rechner, um eine geeignete Näherungsfunktion für die Grenzkostenfunktion zu finden.

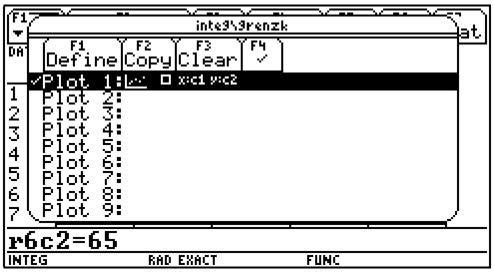
❑ Öffnen Sie mit [APPS] `6:Data/Matrix Editor` eine neues Datenblatt mit dem Namen `grenzk`:



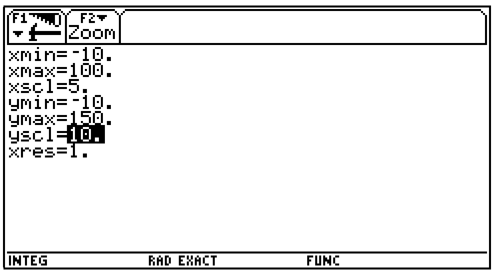


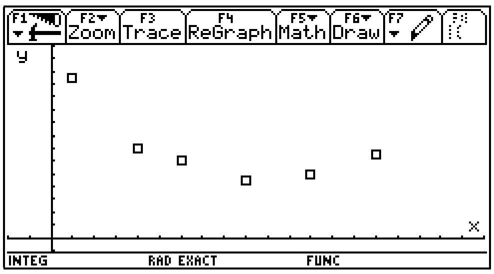
[F2] `Plot Setup` und anschließend [F1] `Define`:



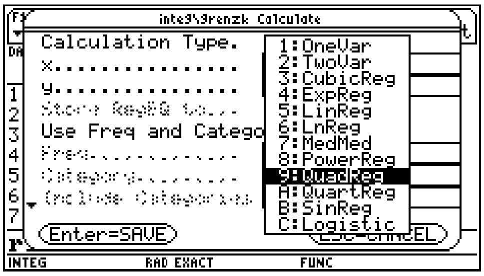


Für eine ordentliche Darstellung müssen noch die [WINDOW]-Werte angepasst werden:

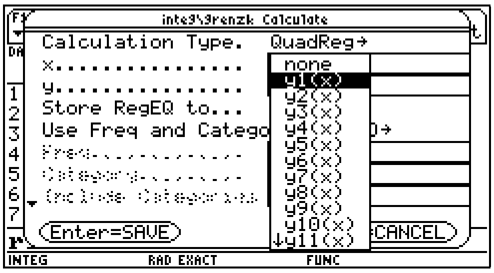


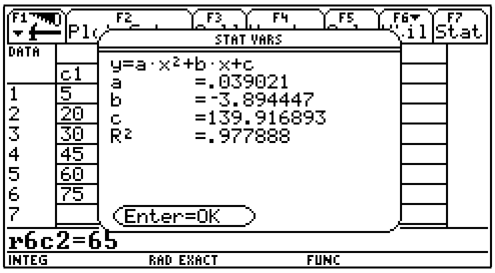


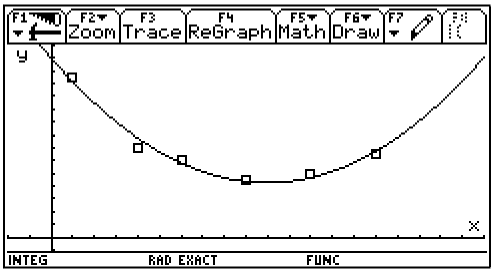
Im Datenblatt gelangen Sie über den Menüpunkt F5 `Calc` zu statistischen Auswertungsmöglichkeiten Ihrer Daten. Nachdem in den Zeilen `x`.... und `y`.... die Spalten `c1` und `c2` als *x*-, bzw. *y*-Werte festgelegt wurden, bietet `Calculation Type` eine Fülle von Bearbeitungsmöglichkeiten:



Hier werden sogenannte „Ausgleichs”- oder „Regressionslinien” angeboten. Die Lage der Punkte erinnert am ehesten an eine Parabel, daher wählen Sie `QuadReg` und lassen die entstehende Regressionsgleichung unter einer geeigneten Bezeichnung im Funktionen-Editor ablegen.







Die angenäherte Grenzkostenfunktion hat daher die Funktionsgleichung

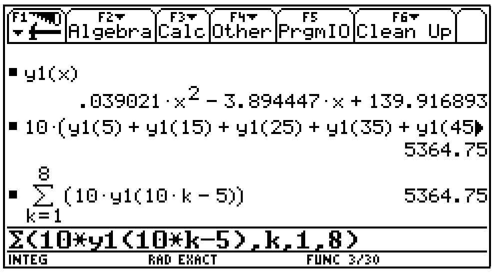
$$GK(x) \;=\; 0{,}039\,x^2 - 3{,}89\,x + 139{,}92.$$

Sie wird ab nun als `y1(x)` angesprochen.

Diese Funktion übernimmt die Rolle der sonst mit der Hand gezogenen willkürlichen Verbindungslinie zwischen den Datenpunkten.

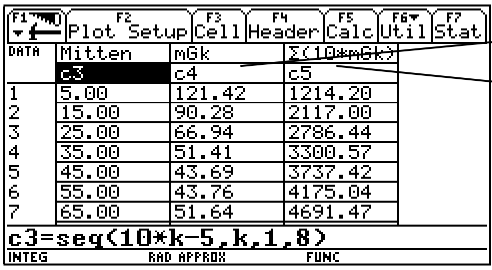
Die Berechung der Gesamtkosten für die ersten 80 Mengeneinheiten lässt sich im Home Screen sofort durchführen.

Mit dem Summenzeichen ist das aber viel eleganter möglich (siehe Abbildung).



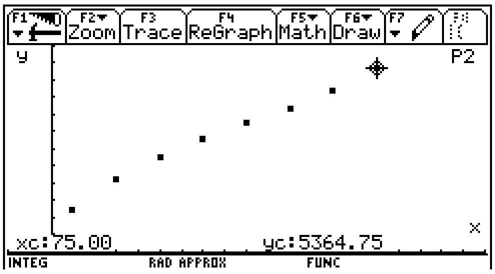
Die Berechnung können Sie auch im Datenblatt durchführen.

☺ Stellen Sie eine Tabelle für die jeweiligen Gesamtkosten für $0 \le x \le 80$ auf, oder entnehmen Sie diese Werte einer geschickt angelegten Spalte in der Datentabelle `grenzk`. Stellen Sie dann auch diese Gesamtkostenfunktion grafisch dar.



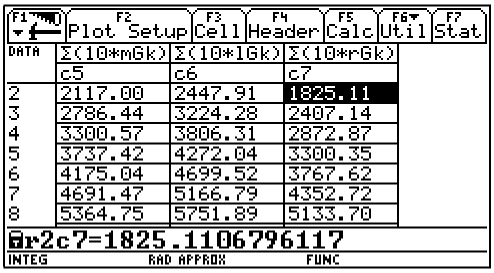
c4 = y1(c3)

c5 = cumSum(10∗c4)



Die Lage der Punkte ist typisch für eine Kostenfunktion. Sie zeigt einen S-förmigen und damit den für den Graphen einer Polynomfunktion 3. Grades charakteristischen Verlauf.

Die Summe dieser so berechneten Rechtecksflächen nennt man **Mittel(punkts)summe**. Sie können das Datenblatt für weitere näherungsweise Berechnungen verwenden: Nehmen Sie für jeden Abschnitt von je 10 Mengeneinheiten den jeweils ersten oder den jeweils letzten Funktionswert als Stückkosten für das ganze Intervall an. Diese Summen werden **Links-** und **Rechtssumme** genannt.



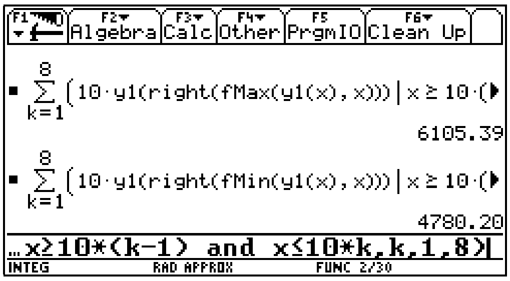
C6 = cumSum(10∗y1(c3-5))

C7 = cumSum(10∗y1(c3+5))

☺ Zeichnen Sie die entsprechenden Rechtecke auch zum Graphen der Funktion. Es fehlt noch eine sehr wichtige Abschätzung. Suchen Sie für jeden „Streifen" die größten Grenzkosten als „Stellvertreter" für das ganze Intervall und berechnen Sie die Gesamtkosten, um damit zur **Obersumme** zu gelangen. Wenn Sie dagegen den jeweils kleinsten Wert verwenden, erhalten Sie die **Untersumme**. Vergleichen Sie dann Ihre „händisch" ermittelten Werte mit den Ergebnissen des TI-92. Schlagen Sie bitte die verwendeten TI-92-Funktionen im Handbuch nach.

❑ Leider lässt sich `fmax(`.... nicht ins Datenblatt übertragen!

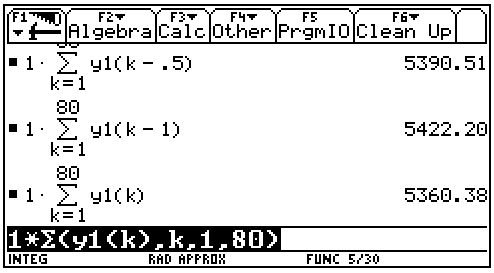
Der gesuchte Wert für die Gesamtkosten liegt sicher zwischen Unter- und Obersumme!!



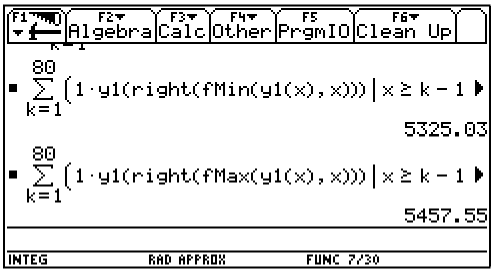
Mit einem CAS können Sie nun ohne viel Aufwand die Rechnung auch im zweiten Sinn genauer machen, indem Sie die Breite der Rechtecke verkleinern und gleichzeitig natürlich deren Anzahl erhöhen.

Machen Sie nun einen gewaltigen Schritt und wählen Sie als Streifenbreite 1 Mengeneinheit. Damit erhalten Sie gleich 80 Streifen. Als Näherungswert für die tatsächlichen Stückkosten verwenden Sie die „theoretischen" Grenzkosten in der Mitte des entsprechenden Intervalls, wobei Sie annehmen können, dass sich diese stetig – und nicht sprunghaft – verändern. Eine derartige Kontinuisierung ist ein häufig angewandtes Hilfsmittel der Mathematik, um zu bequemeren analytischen Modellen zu gelangen. Entsprechend dem vorher Gesagten gibt es neben dieser neuen Mittelsumme natürlich auch Links- und Rechts-, aber auch Unter- und Obersumme.

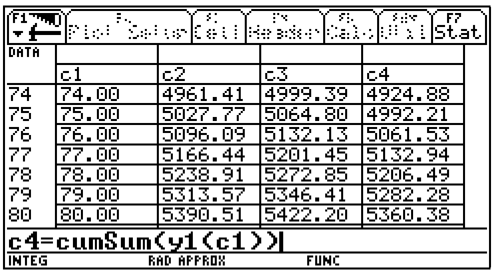
☺ Verändern Sie die Tabelle bzw. arbeiten Sie auch mit den entsprechenden Summen im Home Screen.



Mittel-, Links- und Rechtssumme



Ober- und Untersumme



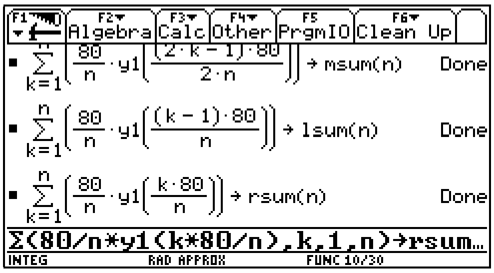
DATA	c1	c2	c3	c4
74	74.00	4961.41	4999.39	4924.88
75	75.00	5027.77	5064.80	4992.21
76	76.00	5096.09	5132.13	5061.53
77	77.00	5166.44	5201.45	5132.94
78	78.00	5238.91	5272.85	5206.49
79	79.00	5313.57	5346.41	5282.28
80	80.00	5390.51	5422.20	5360.38

c4=cumSum(y1(c1))

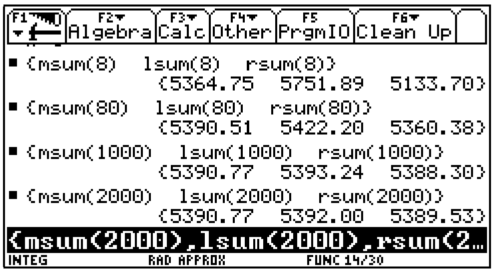
Aus der grafischen Interpretation des Problems erkennen Sie, dass die Suche nach den Gesamtkosten sich als **Berechnung der Fläche** unter der Grenzkostenkurve zwischen zwei Werten $x_1 = 0$ und $x_2 = 80$ interpretieren lässt.

☺ Für eine weitere Verallgemeinerung versuchen Sie nun drei Funktionen `msum(n)`, `lsum(n)` und `rsum(n)` zu erzeugen, wobei *n* die Anzahl der Streifen bedeutet. Überprüfen Sie zuerst die gefundenen Funktionen für *n* = 8 und *n* = 80. Erhöhen Sie dann die Anzahl der Streifen mehr oder weniger systematisch und beobachten Sie das Verhalten der Funktionen. Mit `osum(n)` und `usum(n)` ist das natürlich auch möglich, ihre Anwendung erfordert aber bereits längere Rechenzeiten. Nach diesen Vorbereitungen können Sie das alles – und einiges mehr - getrost dem Programmpaket `integ()` überlassen.

Beachten Sie die syntaktisch richtige Eingabe der Summen!



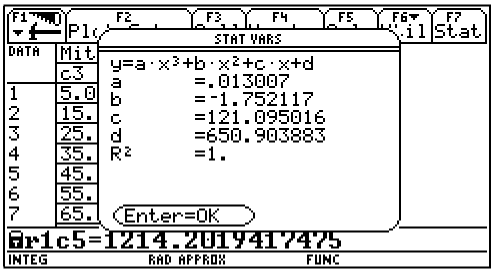
Die Formeln

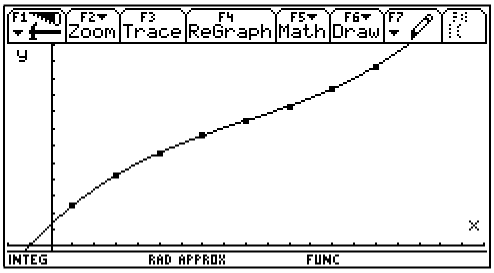


Für n = 2000 *gibt's längere Rechenzeiten!*

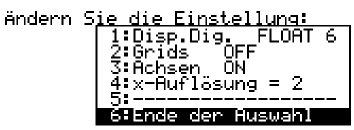
☺ Nützen Sie noch die Möglichkeiten des TI-92, um eine Gesamtkostenfunktion zu ermitteln. Verwenden Sie dazu die Werte in den Spalten `c3` und `c5` aus der Tabelle auf Seite 39. Als Regressionslinie wählen Sie nach dem Gesagten eine Funktion 3. Grades.

Es stellt sich die Frage, wieso der Graph dieser Funktion nicht durch den Koordinatenursprung geht, obwohl noch keine Fixkosten berücksichtigt wurden. Wenn Sie die gleiche Untersuchung mit der Linkssumme durchführen, werden Sie eine Antwort finden.



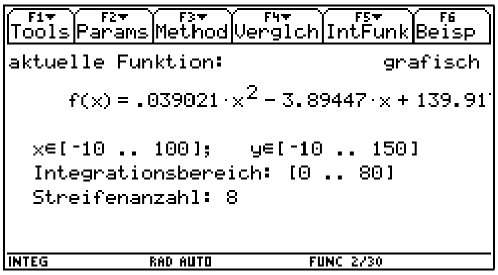


Laden Sie bitte `integ()` und treffen Sie die [F1] `3:Einstellungen` wie angegeben:



Wählen Sie [F2] `1:Alle Parameter` und geben Sie die dem Problem entsprechenden Werte ein. Als Funktionsterm genügt `y1(x)`, sofern Sie die angenäherte Grenzkostenfunktion noch immer unter diesem Namen im [Y=]-Editor gespeichert haben. Der Zeichenbereich ist derselbe, der auf

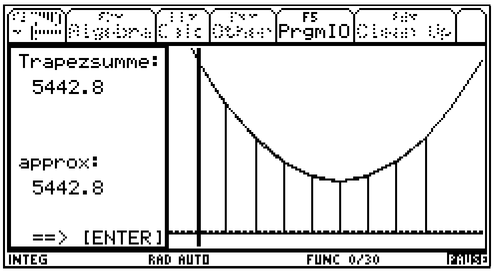
Seite 38 in [WINDOW] angegeben wurde, die Grenzen sind 0 und 80. Wählen Sie zuerst einmal 8 Teilintervalle. Der Bildschirm sollte sich nun so zeigen:



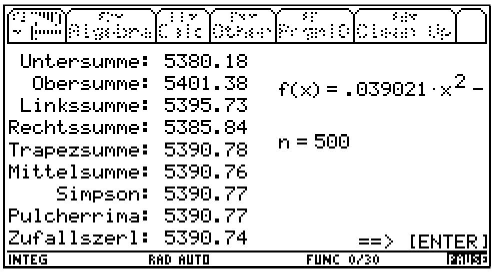
Wählen Sie eine der angebotenen Methoden. (Da eine Grenze 0 ist, lässt sich die geometrische Zerlegung nicht anwenden!)

Vergleichen Sie die Ergebnisse mit den bisher erworbenen.

Im Menüpunkt [F4] können Sie alle (sinnvollen) Methoden numerisch vergleichen.

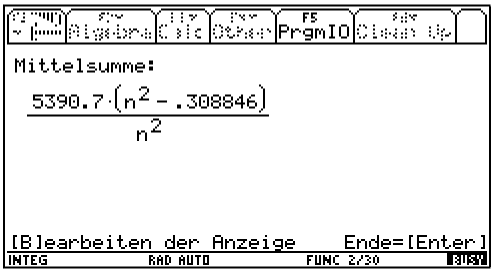


Anstelle der Rechtecke treten Trapeze



Vergleich mit n = 500

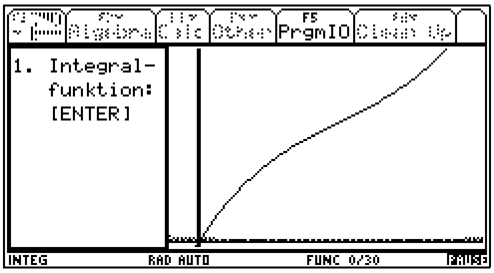
Treffen Sie jetzt die erste wichtige Verallgemeinerung, indem Sie die Streifenanzahl mit *n* nicht mehr numerisch, sondern allgemein angeben. Wählen Sie verschiedene Methoden und bilden Sie jeweils den Grenzwert für $n \to \infty$.



Beispiel Mittelsumme

Der Grenzwert führt zum Wert 5390.7

Lassen Sie die Integralfunktion zeichnen und vergleichen Sie das Ergebnis mit dem Bild des Funktionsgraphen auf Seite 41.



Natürlich entsteht die Frage, warum der Graph dieser Funktion – obwohl ebenfalls S-förmig – offensichtlich durch den Koordinatenursprung verläuft.

Schülerdiskussionen sollten darüber ebenso geführt werden, wie über die Frage, welche Bedeutung die Änderung der Untergrenze hat.

Im Schulgebrauch können Sie nun weiter verallgemeinern, indem Sie auch anstelle der Grenzen 0 und 80 die Werte a bzw. b eingeben und somit direkt zur Integrationsregel gelangen. Der Weg zur Stammfunktion ist im Home Screen durch Substitution von a und b durch 0 und x leicht möglich. Wenn Sie die Schüler diese neue Funktion differenzieren lassen, könnte sich der vom Lehrer so sehr gewünschte Aha-Effekt einstellen. Es stellt sich aber die Frage, ob es sinnvoll ist, die ganze Tragweite des Integralkonzepts bereits an einem ersten einführenden Beispiel bis zur letzten Konsequenz durchzuspielen.

5.2 Was kostet die Haltung eines Fahrzeugs?[11]

Ein Automobilhersteller schätzt, dass sich die jährlichen Instandhaltungskosten eines bestimmten Fahrzeugtyps ungefähr mit der Formel

$$K(t) = 105t + 9{,}5t^2 \quad \text{(EURO)}$$

beschreiben lassen. Dabei bedeutet t das Alter des Fahrzeugs in Jahren.

☺ Wie hoch sind $K(1)$ und $K(5)$ und was bedeuten diese Zahlen?

☺ Welche Gesamtinstandhaltungskosten sind während der durchschnittlichen Lebensdauer eines Fahrzeugs von 10 Jahren zu erwarten?

☺ Verwenden Sie `integ()` und arbeiten Sie mit

a) Jahresintervallen und einer beliebigen Methode,

b) Halbjahresintervallen,

c) Monatsintervallen,

d) Tagesintervallen (365 Tage/Jahr),

e) n Intervallen.

f) Wie hoch sind die „exakten" Gesamtkosten nach dem gewählten Modell?

g) Suchen Sie eine „Formel" $GK(x)$ für die Gesamtkosten der ersten x Jahre.

h) Wie lautet die Formel, wenn Sie das Fahrzeug nach a Jahren kaufen und nach b Jahren wieder verkaufen?

i) Welcher Zusammenhang besteht zwischen der jährlichen Zunahme der Gesamtkos-
ten und den bisher aufgelaufenen Gesamtkosten?

---

[11] Die Anregung zu dieser Aufgabe stammt aus [6].

5.3 $f(x) = 2(x+2)\,e^{-\frac{x}{3}}$

Gesucht ist die Fläche zwischen dem Kurvengraphen und der x-Achse für $-1 \le x \le 6$.

☺ Fertigen Sie eine Skizze des Funktionsgraphen an und suchen Sie einen Näherungswert für die gesuchte Fläche unter Verwendung von 7 Streifen. Benützen Sie den TI-92 zur Bestimmung der Funktionswerte. Zeichnen Sie auch die gewählten ein- oder umschriebenen Rechtecke bzw. Trapeze ein.

☺ Verwenden Sie `integ()` und arbeiten Sie mit der Monte-Carlo-Methode (mit mindestens 200 „Regentropfen"). Addieren Sie alle Ergebnisse in Ihrer Arbeitsgruppe oder lassen Sie die Berechnung mehrmals durchführen und ermitteln Sie den Durchschnittswert.

☺ Erzeugen Sie eine grafische Darstellung der kumulierten Durchschnittswerte für $n = 200, 400, 600, ...$

☺ Wenden Sie andere Methoden (n = 20 Streifen) an und notieren Sie die Ergebnisse. Vergleichen Sie mit den Ergebnissen aus dem „Zufallsregen". Beachten Sie bitte, dass die Zerlegung nach einer geometrischen Folge ohne Transformation (Verschiebung in den positiven Bereich) nicht funktioniert.

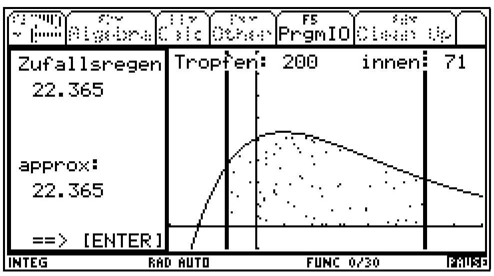
☺ Benützen Sie die Option F4, um die Konvergenz der Methoden zu vergleichen. (Wählen Sie maximal vier Methoden.)

☺ Erste Verallgemeinerung: Wählen Sie drei Methoden und verwenden Sie n Streifen. Bilden Sie jeweils den Grenzwert $n \to \infty$. Können Sie herausfinden, warum Unter- und Obersumme versagen?

☺ Zweite Verallgemeinerung: Wählen Sie drei Methoden und geben Sie als Unter- und Obergrenzen a und b an. Bilden Sie wiederum die Grenzwerte. Speichern Sie ein Ergebnis als `res1`.

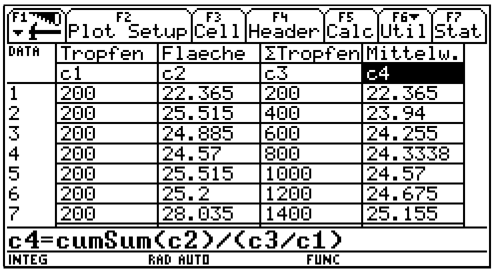
☺ Verlassen Sie `integ()`. Ersetzen Sie im Home Screen b durch x und definieren Sie eine Flächenfunktion `flaeche(x)` mit x als variabler oberer Grenze. Bestimmen Sie den Differentialquotienten dieser Flächenfunktion.

❑ Hier finden Sie einige Screenshots als Anregung:



`ymin` = -1 und `ymax` = 8; daher ist die Fläche des umschriebenen Rechtecks $7 \times 9 = 63$.

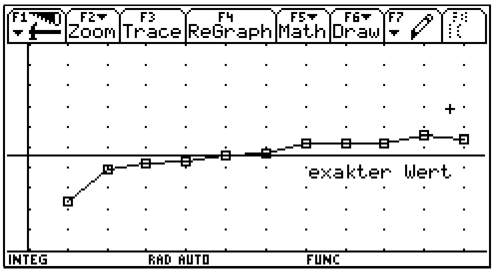
$63 : AppFl = 200 : 71 \to AppFl = 22{,}365$



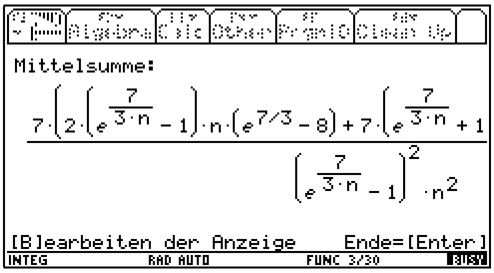
DATA	Tropfen	Flaeche	ΣTropfen	Mittelw.
	c1	c2	c3	c4
1	200	22.365	200	22.365
2	200	25.515	400	23.94
3	200	24.885	600	24.255
4	200	24.57	800	24.3338
5	200	25.515	1000	24.57
6	200	25.2	1200	24.675
7	200	28.035	1400	25.155

c4=cumSum(c2)/(c3/c1)

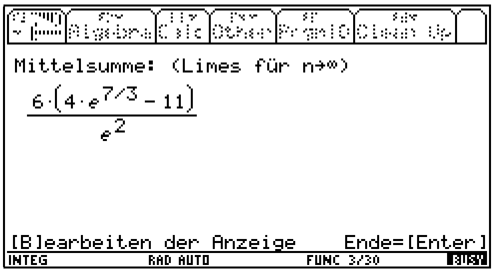
Die Tabelle mit den gesammelten Werten →



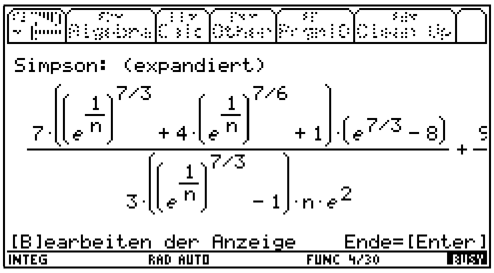
Als `xyline` *dargestellt*



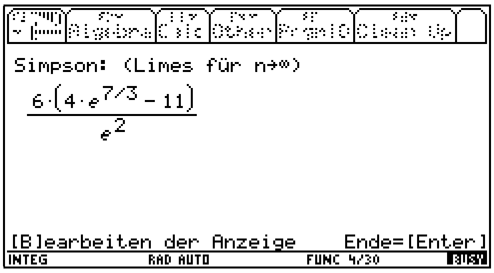
Für n Streifen (TI-92 und TI-92 PLUS können unterschiedliche Ausgabeformen aufweisen!)



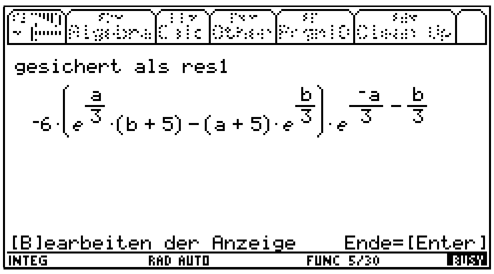
Der zugehörige Grenzwert ≈ **24,5626**



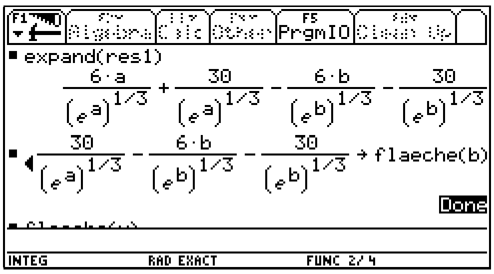
Das Ergebnis für die Simpsonsche Regel →



Mit dem zugehörigen Grenzwert!!

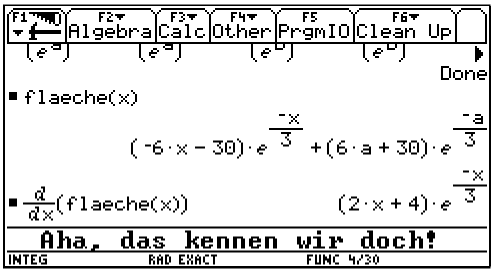


Trapezregel mit allgemeinen Grenzen a und b →

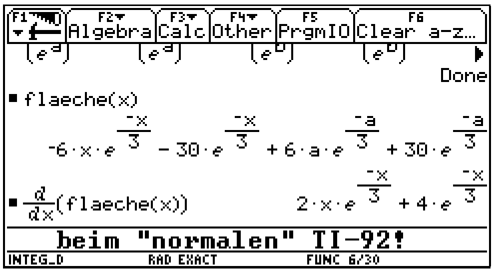


Und im Home Screen geht es weiter

Verlassen Sie `integ()`, ohne alles im Speicher zu löschen, mit [ESC] und arbeiten Sie mit `res1` im Home Screen weiter.



So sieht es aus beim TI-92 PLUS



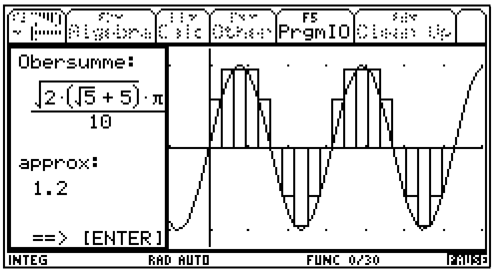
. . . und so beim „normalen” TI-92

5.4 Die Fläche unter der Sinusschwingung

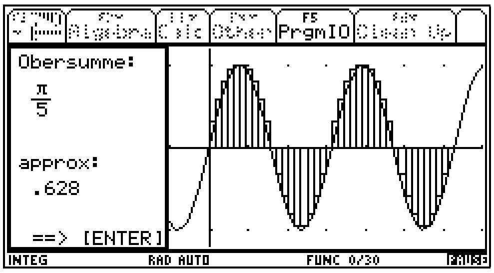
Bestimmen Sie die Fläche zwischen Graph von $f(x) = \sin(2x)$ und x-Achse für $0 \leq x \leq 2\pi$.

☺ Erzeugen Sie Tabellen für verschiedene Methoden für n = 10, 20, 30, ... (inkl. deren grafischer Darstellung). Erklären Sie die „merkwürdigen” Ergebnisse.
Verwenden Sie auch die Monte-Carlo-Methode mit n = 100, 200, 300, ... Tropfen.

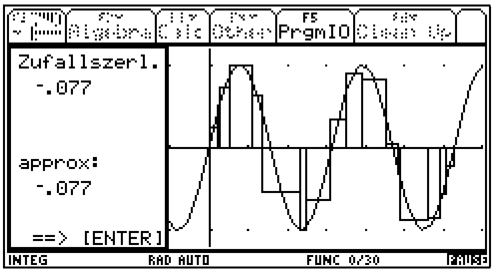
☺ Ändern Sie die Funktion auf $f(x) = |\sin(2x)|$



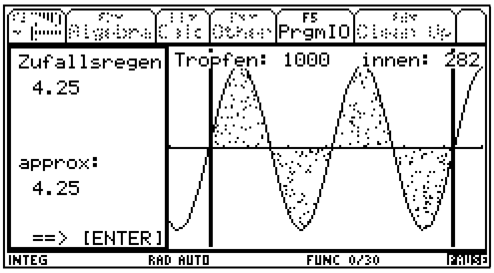
Das kann doch nicht stimmen!



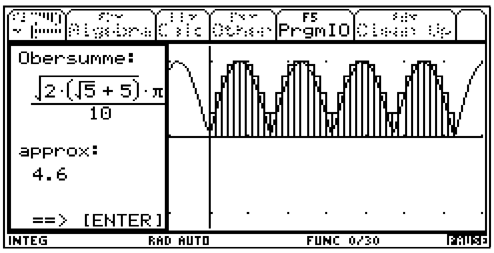
Der Flächeninhalt wird ja immer kleiner!!



Eine Zufallszerlegung mit n = 20



In Monte Carlo haben Sie schon mehr Glück!



Wenn Sie die Betragsfunktion verwenden, liefern natürlich auch die üblichen Methoden sinnvolle Ergebnisse bei der Bestimmung des Flächeninhalts.

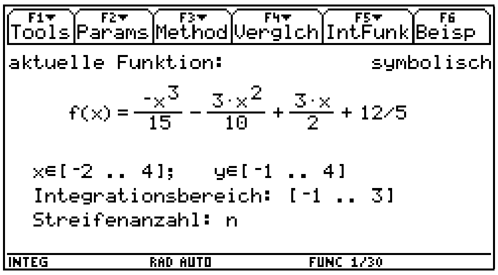
Wie könnte man noch vorgehen?

5.5 Simpson und Pulcherrima bei einer kubischen Funktion

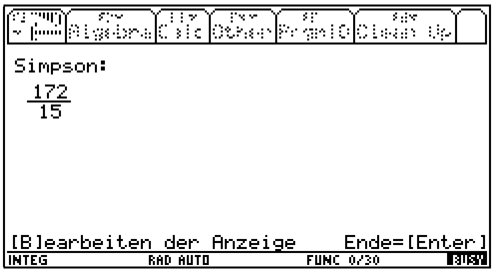
$$f(x) = -\frac{x^3}{15} - \frac{3x^2}{10} + \frac{3x}{2} + \frac{12}{5}\;;\; -1 \le x \le 3.$$

☺ Bestimmen Sie näherungsweise die Fläche zwischen Graph und *x*-Achse unter Verwendung der Simpsonschen Regel und der Pulcherrima.

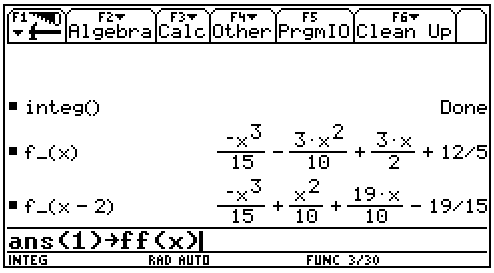
☺ Machen Sie es möglich, auch die Zerlegung nach einer geometrischen Folge anzuwenden, obwohl hier eine negative untere Grenze vorliegt.



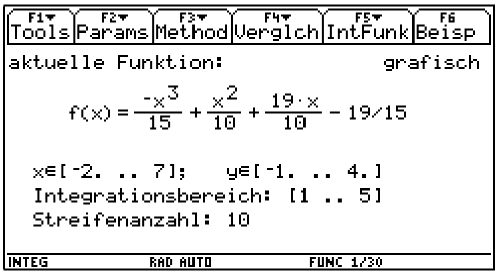
Simpson und Pulcherrima liefern unabhängig von der gewählten Streifenanzahl immer dasselbe Ergebnis. Verallgemeinern Sie mit *n* Streifen.



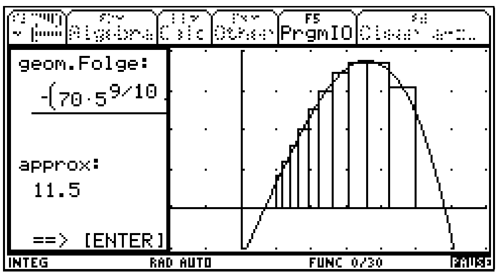
Überraschenderweise ergibt sich weder bei Simpson noch bei Pulcherrima die Notwendigkeit der Grenzwertbildung.



Wechseln Sie über den [ESC]-Ausgang in den Home Screen und verschieben Sie den Graphen so weit nach rechts, dass er ganz in der rechten Halbebene zu liegen kommt. Sie können als Funktionsterm dann `ff(x)` eingeben.



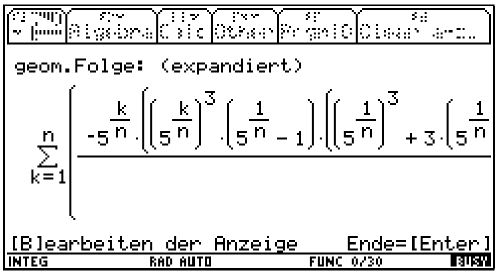
Vergessen Sie nicht, auch die Grenzen zu ändern. Beim „gewöhnlichen" TI-92 ist der Übergang f_(x-2) abzuraten[12], da es wegen des `Circular-Definition`-Effekts bis zum totalen Absturz des Rechners mit Verlust des Speicherinhalts kommen kann.



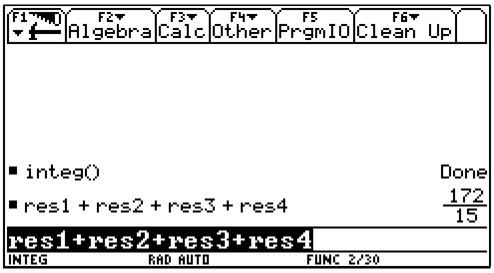
Die Generalisierung mit *n* Streifen und anschließender Grenzwertbildung ist nicht möglich, da der TI-92 die Summen nicht geschlossen darstellen kann. Es gibt aber einen Ausweg: Zerlegen Sie das Polynom in seine vier Summanden ...

---

[12] Dem Circular Definition Error kann man wir folgt „entkommen": f_(x–2) mit einer anderen Variablen aufrufen: zB `f_(u-2)`[ENTER], den Term in u mit `ans(1)|u=x` [ENTER] in einen Term in `x` transformieren und dann an der Funktion `ff(x)` mit `ans(1)`[STO▸]`ff(x)`[ENTER] zuweisen.



So sieht die Summe aus



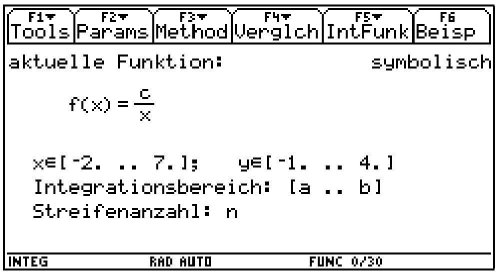
Na, es stimmt ja doch!

5.6 Die Suche nach weiteren elementaren Integrationsregeln

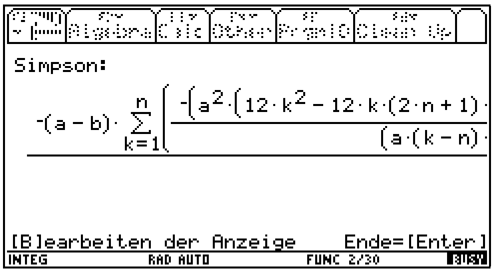
☺ Suchen Sie Integrationsregeln für:

$$\int_a^b c\,x^{\frac{p}{q}}\,\mathrm{d}x\,; \qquad \int_a^b c\,e^{px}\,\mathrm{d}x\,; \qquad \int_a^b c\sin x\,\mathrm{d}x\,; \qquad \int_a^b \frac{c}{x}\,\mathrm{d}x$$

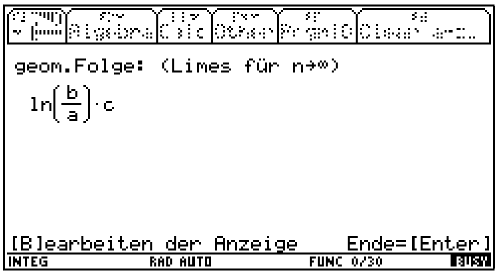
❑ Hier ist es interessant, mit einigen Methoden zu spielen. Aber nur die geometrische Zerlegungsfolge führt zu brauchbaren Ergebnissen.



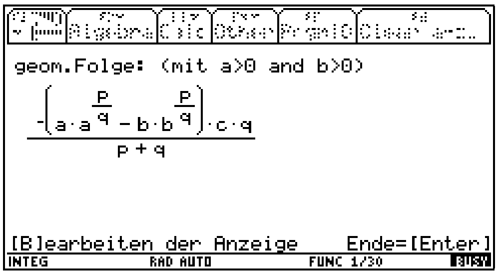
Das Integral von c x⁻¹



Mit den üblichen Methoden wird's nichts



Aber mit der geometrischen Folge!



Das Integral für die allgemeine Potenzfunktion

Leider kann der TI-92 die Summen der trigonometrischen Funktionen nicht bilden. So lassen sich die Integrationsregeln für die Winkelfunktionen nicht ableiten. Mit CAS-Programmen wie etwa *DERIVE* lässt sich das am PC aber leicht durchführen [2].

#54: $\left[F(x) := \cos\left(-\frac{x}{4}\right),\ a := c,\ b := x\right]$

#62: $$\text{MSUM\_VAL}(n) = \frac{\frac{(x-c)\cdot \sin\left(\frac{x}{4}\right)}{2\cdot n} + \frac{(c-x)\cdot \sin\left(\frac{c}{4}\right)}{2\cdot n}}{\sin\left(\frac{x}{8\cdot n} - \frac{c}{8\cdot n}\right)}$$

#63: $$\lim_{n\to\infty} \frac{\frac{(x-c)\cdot \sin\left(\frac{x}{4}\right)}{2\cdot n} + \frac{(c-x)\cdot \sin\left(\frac{c}{4}\right)}{2\cdot n}}{\sin\left(\frac{x}{8\cdot n} - \frac{c}{8\cdot n}\right)} = 4\cdot \sin\left(\frac{x}{4}\right) - 4\cdot \sin\left(\frac{c}{4}\right)$$

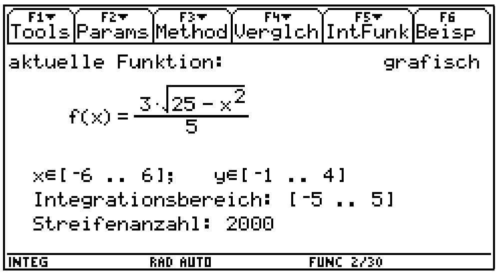
Falls es einem CAS-TI-Anwender gelingen sollte, auch die Integration der Winkelfunktionen auf diese Weise durchzuführen, ersuchen die Autoren um Rückmeldung.

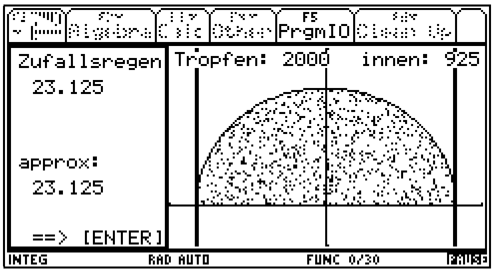
5.7 Welchen Flächeninhalt hat eine Ellipse?

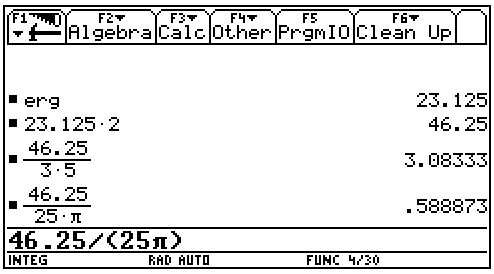
☺ Wählen Sie eine geeignete Ellipse in Mittelpunktslage, zB $a = 5$, $b = 3$.

$$\frac{x^2}{25} + \frac{y^2}{9} = 1 \quad \rightarrow \quad y = \pm\frac{3}{5}\sqrt{25 - x^2}$$; nur die obere Halbellipse!

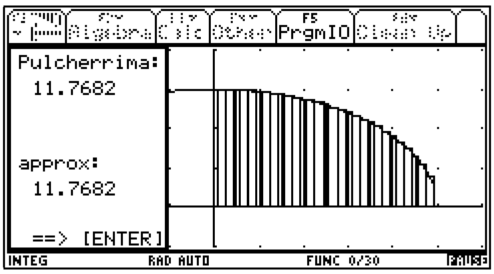
In Anlehnung an die berühmte Monte-Carlo-Methode, den Flächeninhalt eines Kreises zu bestimmen, könnten Sie auch hier den Zufallsregen verwenden.





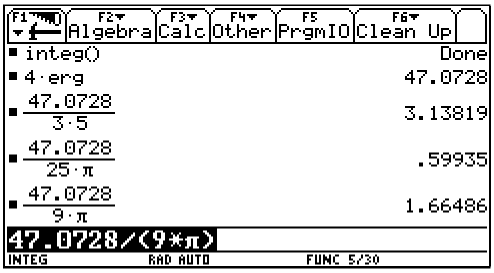


Das Verhältnis Ellipsenfläche : Kreisfläche



Mit der Pulcherrima wird es noch deutlicher!

Hier mit nur 10 Streifen für eine Viertelellipse.
Als Grenze wurde 5.0 eingegeben.



Der Wurzelterm beschreibt den umschriebenen Kreis mit dem Radius $r = 5$. Das Verhältnis der Funktionswerte von Ellipse und Kreis ist offensichtlich $a/b = 3/5$.

Bestimmen Sie das Verhältnis $Fl_{ell} : Fl_{kreis}$!

Eine weitere Überlegung kann sein: Beim Kreis ist das Verhältnis von Kreisfläche zum Quadrat des Radius konstant = π. Lasst uns daher bei der Ellipse das Verhältnis des Flächeninhalts zum Produkt der „beiden Radien a und b" untersuchen.

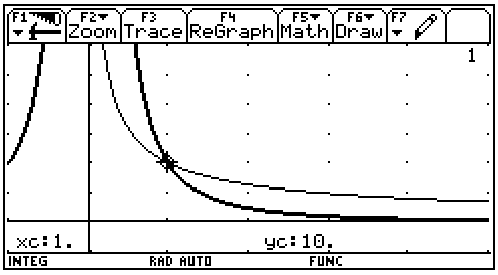
Damit kann die Flächenformel $A = ab\pi$ zumindest einmal begründet werden.

5.8 Was sind eigentlich „uneigentliche" Integrale?

☺ Vergleichen Sie ein mögliches Konvergenzverhalten der beiden Integrale

$$\int_0^4 \frac{10}{\sqrt[3]{x^2}}\,dx \quad \text{und} \quad \int_0^4 \frac{10}{x^2}\,dx$$

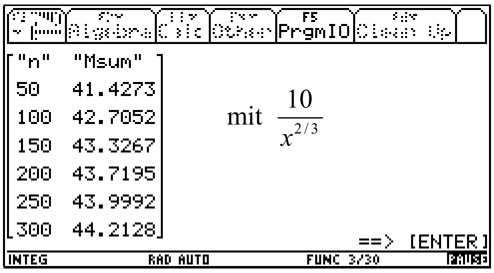
Bei einem groben Vergleich der beiden Funktionsgraphen ist es für die Schüler oft nicht einsichtig, dass die ins Unendliche reichende Fläche das eine Mal existiert und das andere Mal nicht. Reine Grenzwertbildung hat vielleicht nur manipulativen Charakter, so dass es sinnvoll sein könnte, das unterschiedliche Konvergenzverhalten der Flächeninhalte auch zu veranschaulichen.



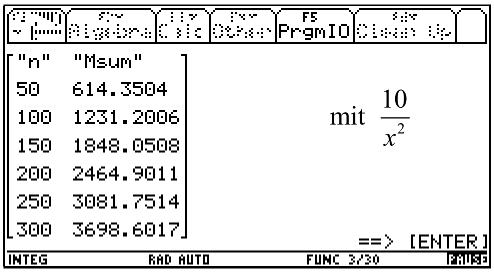
Beachten Sie die [WINDOW]*-Einstellungen*

☺ Wählen Sie für beide Aufgaben eine – geeignete – Methode und lassen Sie die Streifenanzahl n eine Folge 10 .. 100 (Schrittweite 10) durchlaufen.

❑ Natürlich ist nur eine Methode geeignet, die die Stelle $x = 0$ nicht verwendet. Die Linkssumme scheidet damit ebenso aus wie die Trapezsumme und andere.



Flächenzuwachs wird immer kleiner



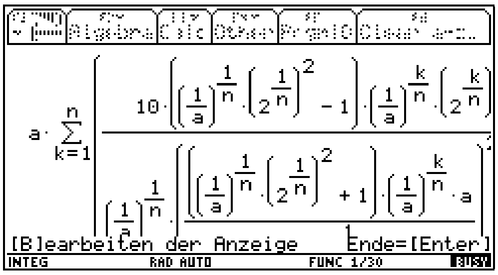
Flächenzuwachs ist streng monoton steigend!

Im Menü [F4] `Verglch` wird nur eine Methode gewählt! (Als Grenze 4.0 eingeben!)

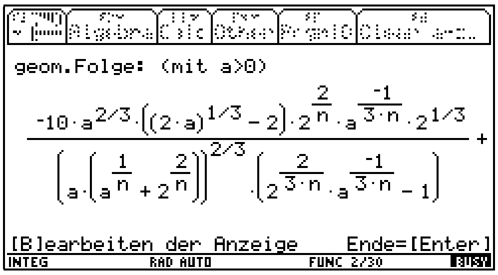
Es muss natürlich darauf hingewiesen werden, dass der Vergleich der Flächenzunahmen auch zu einem Fehlschluss führen kann. Die Tatsache, dass die Zunahmen eine Nullfolge bilden, ist ja nicht hinreichend für die Konvergenz der Summe. Daher jetzt:

☺ Verallgemeinern Sie für n Streifen mit der variablen Untergrenze a. Führen Sie die Grenzübergänge erst $n \to \infty$ und dann $a \to 0$ durch.

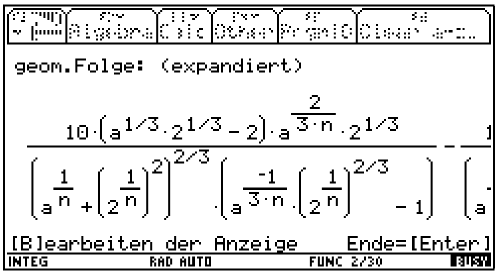
❑ Verfolgen Sie die nächste TI-92-Sequenz für das erste Integral:



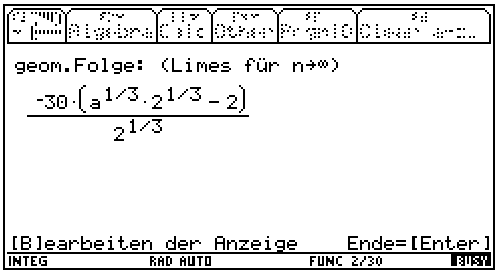
Ein ungeheurer Term ...



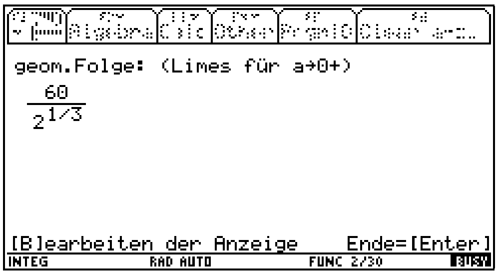
Für $a > 0$ gibt es eine geschlossene Summe



Vor dem Grenzübergang vereinfachen!



Das sieht schon recht ordentlich aus!



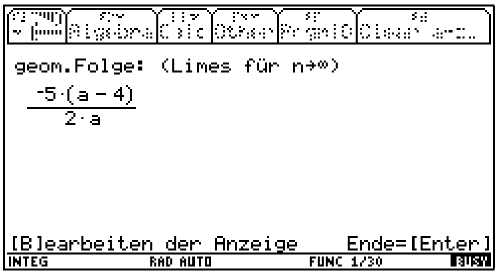
Dieser Grenzübergang ist wohl auch ohne Unterstützung durch den TI-92 möglich!?

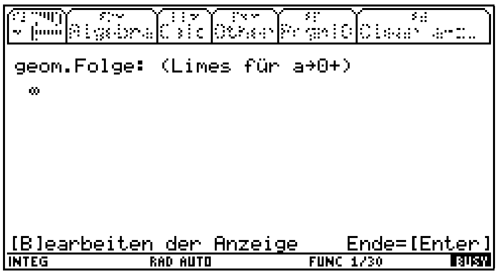
Das approximierte Ergebnis lautet:

47,6220

Vergleichen Sie mit der Folge auf der vorigen Seite!

Gehen Sie mit dem zweiten Integral entsprechend vor:



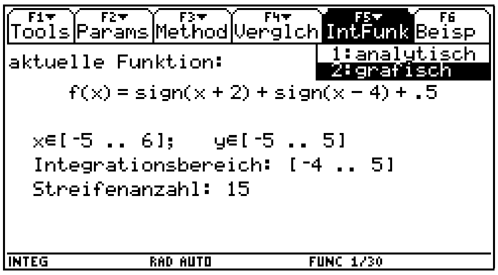


❑ TI-92, TI-92 PLUS und Voyage 200 können unterschiedliche Ausgaben erzeugen.

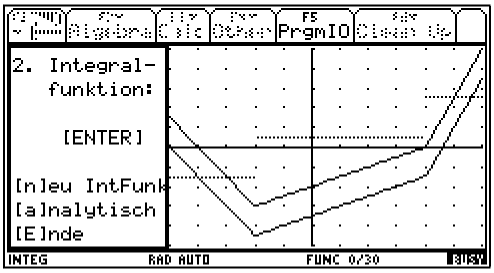
☺ Ändern Sie die das Integrationsintervall auf [4, ∞) und führen Sie ähnliche Untersuchungen durch, um auch die zweite Art der uneigentlichen Integrale zu veranschaulichen. (Das könnte eine schöne Schüleraufgabe sein, nachdem man die Integrale mit einer Polstelle an einer Grenze gemeinsam bearbeitet hat.)

5.9 Wie springt man mit Sprungstellen um?

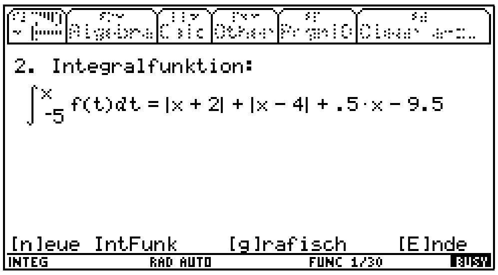
❑ Mit der Signumfunktion `sign()` lassen sich leicht Sprungstellen erzeugen.



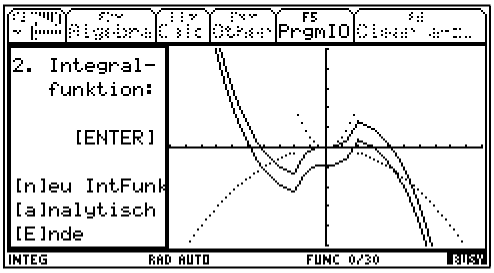
Die „Angabeseite”:



Grafische Darstellung: Untergrenzen – 4 *und* –5



Eine Integralfunktion in analytischer Form



Hier wurde eine andere Funktion mit `when()` definiert. Zwei Integralfunktionen sind dargestellt. Leider lassen sie sich nicht mit dem TI-92 PLUS-Modul ermitteln (siehe S. 36).

5.10 Beziehungen zwischen den Approximationsverfahren

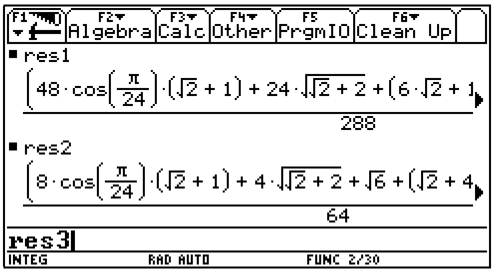
Zwischen den verschiedenen Approximationsmethoden gibt es zahlreiche Beziehungen:

- bekannt und leicht bewiesen: 2 × TRAPSUM = LSUM + RSUM
- auch bekannt? 3 × SIMPSON = TRAPSUM + 2 × MITTSUM
- das auch? 3 × SIMPSON(n) = 4 × TRAPSUM($2n$) – TRAPSUM(n)

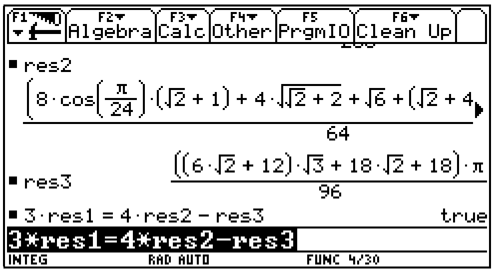
☺ Verifizieren Sie zumindest eine der aufgestellten Beziehungen an Hand einer frei gewählten Funktion. Vielleicht können Sie diese auch unterstützt durch CAS beweisen?

☺ Suchen Sie in der einschlägigen Literatur nach weiteren Beziehungen.

❑ Die dritte Behauptung soll mit der Funktion $y(x) = \frac{3}{2}\sin\left(\frac{x}{2}\right)$ für $\frac{\pi}{3} \le x \le \pi$ verifiziert werden ($n = 4$):
Lassen Sie Simpson ($n = 4$) unter `res1`, Trapezsumme ($n = 8$) unter `res2` und Trapezsumme ($n = 4$) unter `res3` sichern und wechseln Sie in den Home Screen.

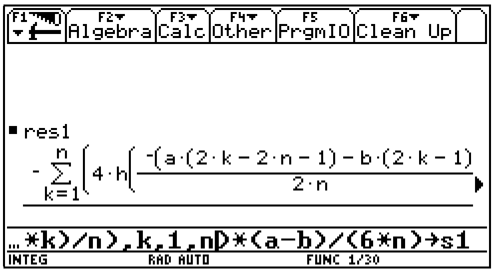


Arbeiten Sie mit exakten Ergebnissen!

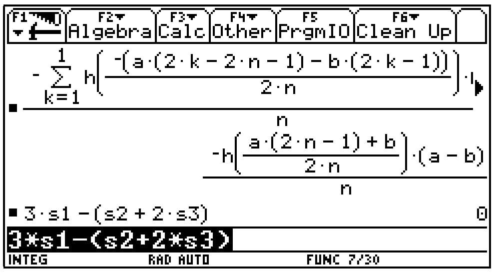


Die Antwort ist zwar kein Beweis, aber doch recht eindrucksvoll!

❑ Die zweite Beziehung soll mit der allgemeinen Funktion $h(x)$ bewiesen werden. Berechnen Sie die drei Summen mit allgemeinem n und den Grenzen a und b und speichern Sie die – recht stattlichen – Ergebnisse unter `res1 ,res2` und `res3`. Wenn Sie die Summen betrachten, werden Sie feststellen müssen, dass diese leider nicht in geschlossener Form angeboten werden. Da aber alle von $k = 1$ bis $k = n$ summiert werden, führt es vielleicht zum Ziel, wenn man zuerst nur die Summandenterme betrachtet und diese als `s1, s2` und `s3` definiert. (Bilden Sie nur die „Summe" des ersten Gliedes.)



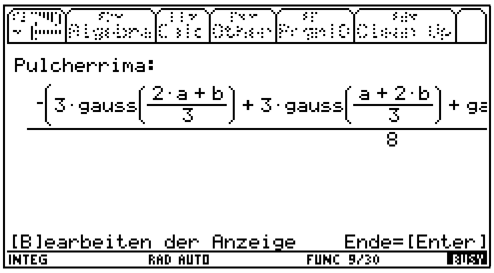
Das `n` vor der Schreibmarke in der Eingabezeile ist durch 1 zu ersetzen.



Die Identität der beiden Seiten der Gleichung liegt auf der Hand.

5.11 Integrieren Sie sich selbst!!

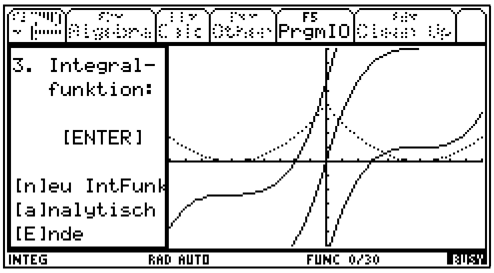
☺ Wählen Sie eine beliebige symbolische Funktion wie zB $g(x)$, $test(x)$ oder $fritz(x)$, nehmen Sie a und b als Grenzen und arbeiten Sie nur mit $n = 1$ Streifen. Wenden Sie eine beliebige Methode an und interpretieren Sie das Ergebnis.



Speichern Sie das Ergebnis und untersuchen Sie im Home Screen die Vorgehensweise bei der Pulcherrima-Methode (Gauß-Quadratur).

5.12 Diese Funktion ist nicht differenzierbar!

Erzeugen Sie eine Funktion mit zumindest einer nicht differenzierbaren Stelle im Integrationsintervall und lassen Sie die Integralfunktion(en) zeichnen. Interpretieren Sie das Ergebnis.



Die Funktion $f(x) = \left(2 - \frac{|x|}{3}\right)^2$ ist an der Stelle $x = 0$ nicht differenzierbar. Ihre Integralfunktion anscheinend schon.
Bestimmen Sie diese analytisch und untersuchen Sie deren Differenzierbarkeit für $x = 0$.

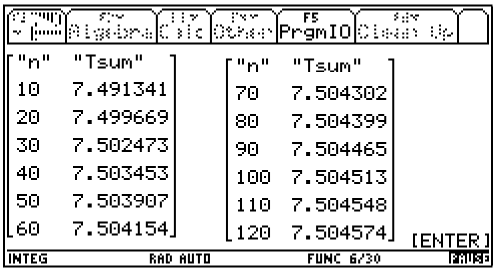
5.13 Dafür gibt es sicher kein geschlossenes Integral!

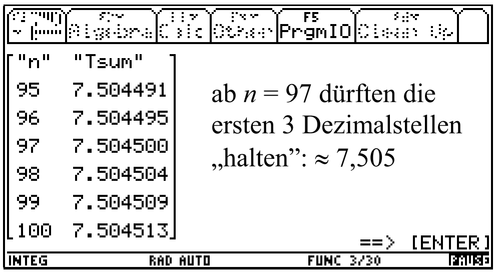
Suchen Sie eine numerische Approximation für $\displaystyle\int_{-1}^{3} \frac{x^2 + e^x}{1 + x^2 + \sin 2x}\,\mathrm{d}\,x$

☺ Wie viele Streifen sind bei den verschiedenen Verfahren notwendig, um eine Genauigkeit auf 3 Dezimalstellen zu erreichen? Vergleichen Sie unter mehreren Methoden!

☺ Vergleichen Sie Ihre Ergebnisse mit der `nInt()`-Funktion des TI-92.

❑ Zur Abschätzung der benötigten Streifen lässt sich die im `verglch`-Werkzeug angebotene Tabelle mit schrittweiser Verfeinerung anwenden. In der Literatur finden sich genaue Abschätzungen [5, 13].





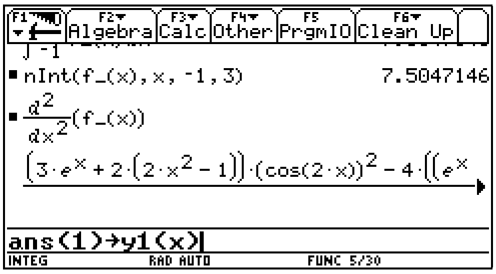
Sowohl das Integral als auch `nInt` liefern den Wert `7.5047146`.

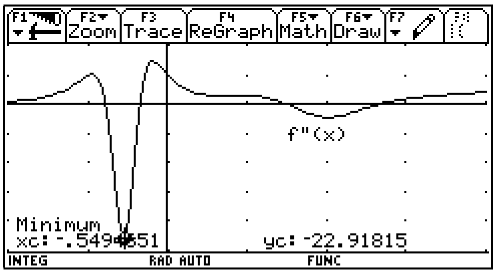
Hier wurde das Programm im Programmbaustein `vgl` auf 6 Fixkommastellen abgeändert.

In [5] und in [13] finden Sie die folgende Abschätzung für die Anzahl von Streifen bei Anwendung der Trapezregel, wenn eine Genauigkeit von ε erreicht werden soll. ε ist der Unterschied zwischen tatsächlichem und approximiertem Wert des Integrals.

$$n > \sqrt{\frac{M_2\,(b-a)^3}{6\,\varepsilon}}\,.\ \text{Dabei ist}\quad M_2 = \max|f''|\ \text{auf}\ [a, b] \qquad (*)$$

Mit dem TI-92 können Sie leicht das Maximum des Absolutbetrags der zweiten Ableitung der gegebenen Funktion im Intervall [-1, 3] aufsuchen.





Wenn Sie nun in die Formel (*) einsetzen, ergibt sich:

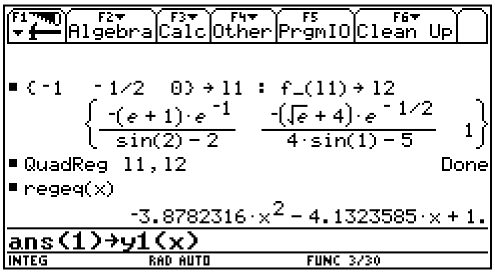
$$n > \sqrt{\frac{22{,}91815 \cdot 4^3}{6 \cdot 0{,}001}} = 494{,}4\,.$$

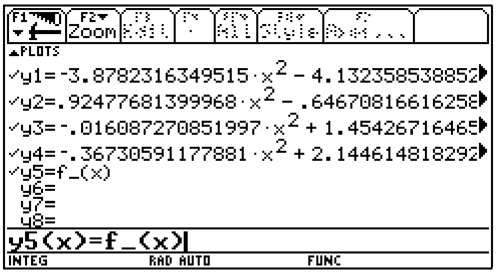
Diese Abschätzung ist viel strenger, als die mit der Tabelle ermittelte.

☺ Die Simpson-Regel entspricht einer quadratischen Approximation der zu integrierenden Kurve. Verifizieren Sie diesen Sachverhalt, indem Sie $n = 4$ annehmen. In jedem entstehenden Intervall wird die Kurve durch eine quadratische Funktion approximiert, die mit der gegebenen Funktion die Funktionswerte an den Intervallenden und in der Intervallmitte gemeinsam hat. Erzeugen Sie auch die „quadratische Approximationskurve”.

❑ Verwenden Sie die quadratische Regression, um durch drei Punkte eine Parabel zu legen. Die entstehenden Funktionen werden im Funktioneneditor abgelegt. Verfolgen Sie bitte dazu die nächsten Abbildungen:

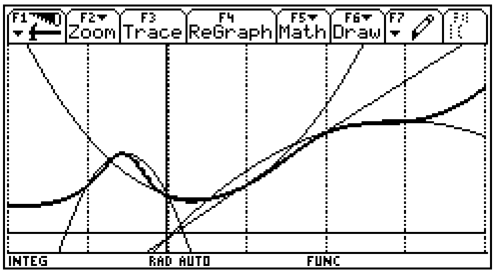
Die Funktion sei unter dem Namen `f_(x)` gespeichert. (Wenn Sie `integ()` über [ESC] verlassen, dann finden Sie die Funktion unter diesem Namen!)

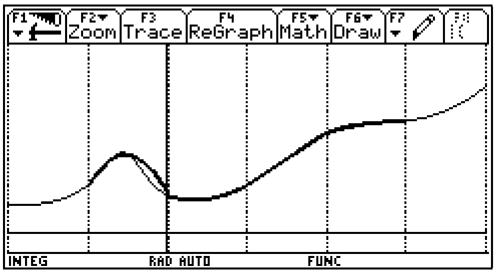




Hier entsteht die erste Parabel für [-1,0]. `regeq(x)` ist eine Systemvariable unter der die jeweils letzte berechnete Regressionslinie abgelegt wird. Gehen Sie entsprechend für die nächsten drei Intervalle vor.

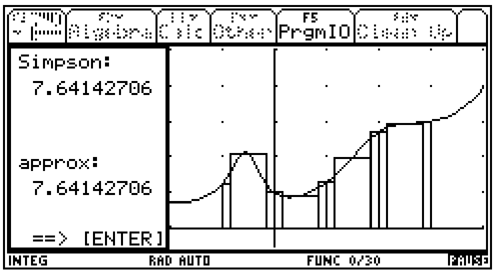
Im [Y=]-Editor finden Sie alle vier approximierenden Parabeln. Im nächsten Bild sehen Sie die Funktion gemeinsam mit den Parabeln.

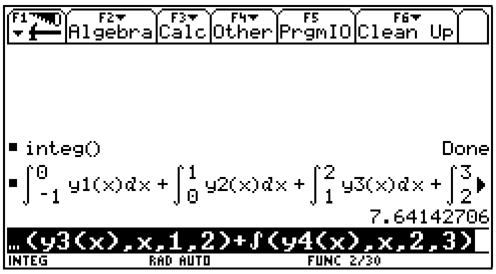




Mit einer `when()`-Konstruktion können Sie die Näherungskurve stückweise definieren und ganz stark ausgezogen über die Grundfunktion legen.

Der rechnerische Vergleich zeigt das erwartete Resultat.





Die Simpson-Approximation und die Fläche unter den vier Parabeln sind gleich.

Es könnte reizvoll für interessierte Leser sein, die Prozedur für die Entwicklung der Näherungsparabeln in einem Script oder gar in einem Programm zu verpacken.

5.14 Ein Ausflug in die Wahrscheinlichkeitsrechnung

Suchen Sie eine numerische Approximation für

$$\int_2^4 \frac{1}{\sigma\sqrt{2\pi}}\, e^{-\left(\frac{X-\overline{X}}{\sigma}\right)^2} \mathrm{d}X \quad \text{mit} \quad \overline{X} = 3{,}5 \quad \text{und} \quad \sigma = 1{,}5$$

Welche Bedeutung hat dieses Integral?

Vergleichen Sie Ihr Ergebnis mit einem Tabellenwert!

6 Bemerkungen zur Pulcherrima

(mit freundlicher Genehmigung von David Bowers, Ipswich)

Einer der Teilnehmer am Workshop, den die beiden Autoren zum Thema dieses Buches im Rahmen der 3. Internationalen DERIVE- und TI-92 Konferenz, Juli 1998, in Gettysburg gehalten haben, war unser Freund David Bowers aus Ipswich. Im August erhielt Josef Böhm einen Brief von David, der, angeregt vom „schönen" Namen „Pulcherrima" – „die Schönste"– sich näher mit dieser numerischen Methode befasste. Die Autoren danken David für die Erlaubnis, seine „revision notes" hier aufnehmen zu können und hoffen, damit vielen Lesern eine interessante Idee zu vermitteln.

Die Methode „Pulcherrima" erinnert mich an die **Quadratur-Formel von Newton-Cotes**, die ich vor vielen Jahren kennengelernt habe:

Der Anfang ist wohl bekannt. Um das Integral $I = \int_a^b f(x)\,\mathrm{d}x$ zu berechnen wird der Integrationsbereich in n gleiche Streifen der Breite $h = \frac{b-a}{n}$ geteilt.

Daher gilt $x_0 \equiv a$ und $x_k = x_0 + k\,h$ mit $k = 1, 2, .., n$.

Nun wird $f(x)$ durch das Lagrange-Interpolationspolynom approximiert:

$$\sum_{k=0}^{n} l_k(x)\cdot y_k \quad \text{mit } y_k \equiv f(x_k) \text{ und}$$

$$l_k(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)} = \prod_{\substack{i=0\\ i\neq k}}^{n} \frac{x-x_i}{x_k-x_i}$$

Damit ergibt sich für das Intergral I

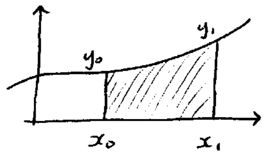
$$I \approx \int_{x_0}^{x_n} \sum_{k=0}^{n} l_k(x) \cdot y_k \,\mathrm{d}x = \sum_{k=0}^{n} \left(\int_{x_0}^{x_n} l_k(x)\,\mathrm{d}x \right) \cdot y_k = \sum_{k=0}^{n} A_k \, y_k \quad \text{mit} \quad A_k = \int_{x_0}^{x_n} l_k(x)\,\mathrm{d}x$$

Dies ist bekannt unter dem Namen

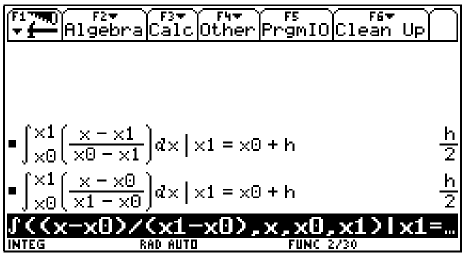
„(n+1)-Punkte-Quadratur-Formel von Newton-Cotes".

Spezialfälle ergeben sich für n = 1, 2, 3, etc.

Der Fall für *n* = 1



$$A_k = \int_{x_0}^{x_1} l_k(x)\,\mathrm{d}x \; ; \qquad k = 0{,}1$$



Die beiden Werte A_0 und A_1 ergeben sich leicht mit dem TI-92.

(Die händische Rechnung stellt eine gute Manipulationsübung mit einem einfachen Integral für die Schüler dar!)

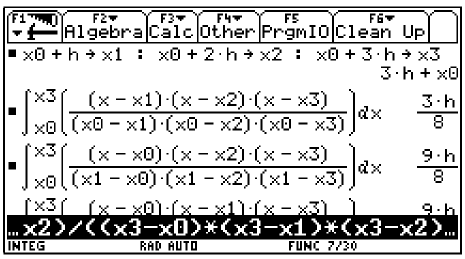
Daher ergibt sich nach Einsetzen in die Quadraturformel:

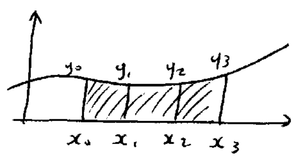
$$I \approx \frac{h}{2} y_0 + \frac{h}{2} y_1 = \underline{\underline{\frac{h}{2}\left(y_0 + y_1\right)}}$$. Und darin erkennt man die **Trapezregel**.

Der Fall für *n* = 2:

.... führt zur **Simpsonregel.** Der Nachweis bleibe dem Leser überlassen.

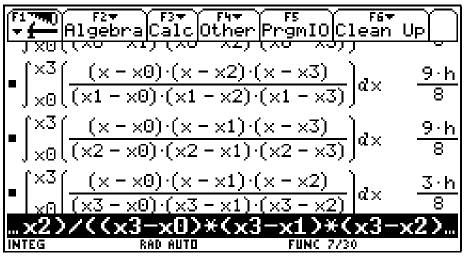
Der Fall für *n* = 3:





Der TI-92 übernimmt wieder das Integrieren der A_k!

$$A_k = \int_{x_0}^{x_1} l_k(x)\,\mathrm{d}x \; ; \qquad k = 0{,}1{,}2{,}3$$



Und es ergibt sich für die Approximation des Integrals:

$$I \approx \frac{3h}{8}\left(y_0 + 3y_1 + 3y_2 + y_3\right)$$

Das ist Deine „**Pulcherrima**". Bei uns in England heißt sie manchmal „**Three-eights Rule**" - die „**Drei-Achtel-Regel**".

Zusammenfassung der Newton-Cotes-Regel

$n = 1$	$\frac{h}{2}\left(y_0 + y_1\right)$	Ordnung des Fehlers h^3
$n = 2$	$\frac{h}{3}\left(y_0 + 4y_1 + y_2\right)$	h^5
$n = 3$	$\frac{3h}{8}\left(y_0 + 3y_1 + 3y_2 + y_3\right)$	h^5
$n = 4$	$\frac{2h}{45}\left(7y_0 + 32y_1 + 12y_2 + 32y_3 + 7y_4\right)$	h^7
$n = 5$	$\frac{5h}{288}\left(19y_0 + 75y_1 + 50y_2 + 50y_3 + 75y_4 + 19y_5\right)$	h^7
$n = 6$	$\frac{h}{140}\left(41y_0 + 216y_1 + 27y_2 + 272y_3 + 27y_4 + 216y_5 + 41y_6\right)$	h^9

Beachte, dass gilt: $\sum_{k=0}^{n} A_k = n\,h =$ Integrationsintervall.

Und schließlich noch ...

Weddle's Regel ist eine heuristische Approximation der 7-Punkt-Newton-Cotes-Regel, die von den Ingenieuren in den Prä-Taschenrechnerzeiten für händische Berechnungen verwendet wurde. Und das wohl wegen der angenehmen Koeffizienten

$$I \approx \frac{3h}{10}\left(y_0 + 5y_1 + y_2 + 6y_3 + y_4 + 5y_5 + y_6\right).$$

Eine allerletzte Bemerkung:
Leider ist die Programmierung der Formeln für den allgemeinen Fall der Lagrange-Interpolation wegen des „fehlenden" Faktors zwischen $(x_k - x_{k-1})$ und $(x_k - x_{k+1})$ im Nenner schwierig. Wenn aber, wie hier, die Stützpunkte gleiche Abstände haben, ist es eine reizvolle Aufgabe zu zeigen, dass der Nenner des Lagrange Polynoms für die n-Punkt-Formel durch den folgenden Term gegeben ist:

$$(n-k)!\,(-1)^{n-k}\;k!\,h^n$$

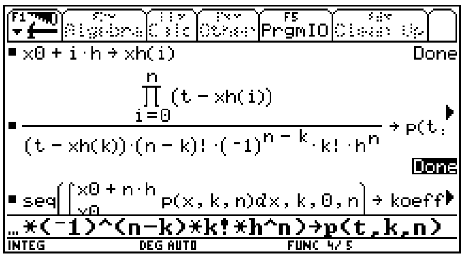
Daher können alle l_k erzeugt werden als

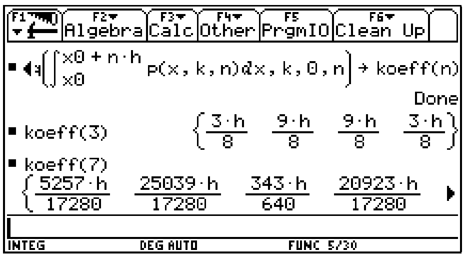
> das Produkt aller $(x - x_i)$ für $i = 0$ bis n, dividiert durch $(x - x_k)$ und außerdem dividiert durch den obigen Ausdruck.

Ich schließe die DERIVE-Datei PULCH.MTH an, die die genannten Koeffizienten erzeugt. Meine „Bibel" für Probleme der numerischen Analysis ist immer wieder „Introduction to Numerical Analysis" von Carl-Erik Froberg.

David Bowers

Gemäß den Absichten dieses Buches haben die Autoren die DERIVE - Datei buchstäblich in die TI-92-Syntax übertragen. (Nur der VECTOR-Befehl aus DERIVE wurde in den `seq()` Befehl des TI-92 „übersetzt".)





Literaturhinweise

[1] F. J. Santonja Gómez; *The Riemann Integral with the TI-92*, DERIVE Newsletter 1999

[2] J. Böhm, Hg; *Teaching Mathematics with DERIVE*, Chartwell Bratt 1992

[3] J. Böhm, *Riemann at Random, True Riemann Rectangles*, DERIVE Newsletter #7,#8

[4] T. Etchells, *True Riemann Rectangles*, DERIVE Newsletter #8

[5] S. Grosser, *Numerische Integration (mit Computereinsatz) als Unterrichtsmodul*, Didaktikhefte, Heft 26, Österreichische Mathematische Gesellschaft 1996

[6] F.S. Budnick, *Applied Mathematics for Business, Economics, and the Social Sciences*, McGraw-Hill 1986

[7] J. Böhm und W. Pröpper, *From Counting Raindrops to the Fundamental Theorem* (Lecture and Workshop), International DERIVE and TI-92 Conference 1998, Gettysburg, CD mit den Konferenzbeiträgen erhältlich bei Mathware, Urbana, IL

[8] D. Bowers, "Revision notes .....", private Korrespondenz mit den Autoren.

[9] T. Etchells & J. Berry, *Learning Numerical Analysis through DERIVE*, Chartwell-Bratt 1997

[10] St. Schonefeld, *Numerical Analysis via DERIVE*, Mathware 1994

[11] B. Kutzler, *Symbolrechner TI-92*, Addison-Wesley 1996

[12] TI-92 Handbuch, Texas Instruments

Weitere empfehlenswerte Bücher (mit vielen anwendungsorientierten Aufgaben zur Integralrechnung):

[13] Finney, Thomas, Demana und Waits, *Calculus*, Addison-Wesley 1995

[14] S Waner & S.R. Costenoble, *Calculus Applied to the Real World*, Harper Collins 1996

[15] L Edwards, *Calculus, An Applied Approach*, Houghton Mifflin 1999

[16] R Harshbarger & J R Reynolds, *Calculus, Mathematical Applications*, Houghton Mifflin 2000

[17] R. Baumann, *Analysis 2*, Klett 2002