Luceños Diskretisierungstechnik für stetige Verteilungen – Implementierung und statistische Anwendung

Diplomarbeit
von
Michael Obermeier

Betreuung:
Prof. Dr. Thomas Augustin

04. Oktober 2006

Institut für Statistik
Ludwig-Maximilians-Universität München

Inhaltsverzeichnis

1. Einleitung

Die vorliegende Diplomarbeit beschäftigt sich mit zwei von Alberto Luceño aufgestellten Theoremen, die 1999 im *Journal of the Royal Statistical Society* [9] veröffentlicht wurden. Luceño beschreibt darin, wie mithilfe der Gaußschen Quadratur mit N orthogonalen Polynomen stetige Zufallsvariablen diskret approximiert werden können. Dabei überführt er in seinem ersten Theorem das Vorgehen bei der Gaußschen Quadratur direkt in die Statistik und baut in Theorem 2 diese Theorie noch weiter aus. Die neue, diskrete Variable hat nach diesem Verfahren zumindest die gleichen ersten $2N - 1$ Momente wie die stetige. Da diskrete Zufallsvariablen numerisch und analytisch generell einfach zu handhaben sind, empfiehlt es sich bei speziellen Situationen in der Statistik, möglicherweise vorkommende stetige Variablen zu diskretisieren. In diesem Zusammenhang sind auch Luceños Theoreme und diese Diplomarbeit zu sehen.

In Kapitel 2 beschreibe ich zunächst die Theoreme Luceños (Unterkapitel 2.1), wobei ich auch kurz auf die Grundzüge der Gaußschen Quadratur eingehe. Eine gute Einführung in dieses Thema bieten beispielweise Stoer und Bulirsch ([16]). Die Implementierung der Theoreme folgt dann in 2.2, wobei sie sich hauptsächlich auf die Ergebnisse aus Theorem 2 beschränkt. Als statistische Software verwende ich das Programmpaket `R`, in dem es bereits implementierte Verfahren für die direkte Verknüpfung von Gaußscher Quadratur und Wahrscheinlichkeitsverteilungen, also Theorem 1, gibt.

In Kapitel 3 gehe ich der Frage nach, inwieweit die Approximation einer stetigen Zufallsvariable durch eine mit Luceños zweitem Theorem gefundene diskrete Variable glücken kann. Anhand verschiedener Tests auf Verteilung soll diese Frage beantwortet werden. Kurz gehe ich in diesem Zusammenhang in 3.5 noch auf eine andere Diskretisierungstechnik und deren Eigenschaften im Vergleich mit Luceños Diskretisierung ein.

Die klassische Anwendung der Gaußschen Quadratur liegt in der numerischen Berechnung von Integralen. Dementsprechend liegt es nahe zu fragen, inwiefern sie direkt bei in der Statistik typischen Integralberechnungen eingesetzt werden kann. In Kapitel 4 behandle ich dieses Thema überblicksweise.

Vielversprechend scheinen die Möglichkeiten der Anwendung dieser Diskretisierungsmethode in der Entscheidungstheorie, speziell in Situationen unter komplexer Unsicherheit. Das ist das Thema der umfangreicheren Kapitel 5 und 6, die auch den Schwerpunkt dieser Forschungsarbeit bilden. In Kapitel 5 behandle ich dabei die klassische Entscheidungstheorie, in der die Gaußsche Quadratur gewinnbringend bei großem Aktionenraum eingesetzt werden kann. Die nichtklassische Entscheidungstheorie behandle ich im sechsten Kapitel. Zunächst geht es dabei um die Frage, wie Entscheidungskriterien berechnet werden können mit einer Zustandsverteilung aus einer bekannten Familie, bei der lediglich Parameter dieser Verteilung unsicher sind. Ziel des Kapitels ist es dann aber vor allem, eine Strategie der Entscheidungsfindung darzulegen, wenn als Zustandswahrscheinlichkeiten lediglich obere und untere Grenzen in Form von sogenannten F-Wahrscheinlichkeiten (siehe [20]) bekannt sind.

In Kapitel 7 folgt noch eine zusammenfassende Bewertung der gezeigten Ergebnisse, sowie ein kurzer Ausblick auf mögliche weitergehende Studien zu diesem Thema.

2. Luceños Diskretisierungstechnik

Luceños Abhandlung über diskrete Approximationen für stetige Verteilungen [9] stellt, plakativ ausgedrückt, eine Übersetzung der aus der Numerik bekannten Gaußschen Quadratur in die Statistik dar. Dabei werden als Gewichtsfunktion die Dichtefunktion einer stetigen Zufallsvariablen und als Integralgrenzen die Grenzen ihres Trägers verwendet. Die so gefundenen Stützstellen und Gewichte zur numerischen Approximation des Integrals lassen sich als neue, diskrete Zufallsvariable interpretieren, die ähnliche Eigenschaften wie die stetige aufweist. Die Gaußsche Quadratur kann somit zur Diskretisierung stetiger Verteilungen eingesetzt werden.

Theorem 1 von Luceño entspricht dabei exakt der aus der Numerik bekannten Gaußschen Quadratur zur effizienten Berechnung von Integralen, wenn als Gewichtsfunktion des Integrals Dichtefunktionen stetiger Verteilungen betrachtet werden. Im zweiten Theorem wird die Diskretisierung durch eine Aufteilung des Trägers der stetigen Variable in Intervalle verbessert.

2.1 Theoretische Grundlagen

Die Grundidee der Gaußschen Quadratur liegt darin, dass mithilfe von rekursiv definierten, orthogonalen Polynomen die Integrale

$$\int_a^b h^r(x)w(x)d(x)$$

einer Funktion $h(x)$ und ihrer Gewichtsfunktion $w(x)$ durch die Summe

$$\sum_{j=1}^{N} h^r(x_j)w_j$$

ersetzt und numerisch approximiert werden sollen. Dabei wird $h(x)$ nurmehr an den Stellen x_j, den sogenannten Stützstellen, berechnet und die stetige Gewichtsfunktion $w(x)$ durch die diskreten Gewichte w_j ersetzt. Stützstellen und zugehörige Gewichte ergeben sich aus den Nullstellen der Polynome vom Grad N, welche wiederum aus der Gewichtsfunktion $w(x)$ hervorgehen. Im Gegensatz zu anderen numerischen Integrationsmethoden, wie beispielsweise den Newton-Cotes Formeln, werden hier die x_j und w_j dynamisch, also möglichst „geschickt", gewählt, was zu sehr genauen Approximationen der Integralwerte führt (Zur Gaußschen Quadratur und anderen numerischen Integrationsmethoden siehe beispielsweise [16]).

Direkt angewandt auf die Statistik, stellt die Gewichtsfunktion $w(x)$ die Dichtefunktion einer stetigen Zufallsvariablen X dar und entsprechen die w_j der Dichte einer diskreten Zufallsvariable Y.

2.1.1 Theorem 1

Es sei X eine stetige, univariate Zufallsvariable mit finiten r-ten Momenten für $r = 0, 1, \ldots$ und der Dichtefunktion $w(x)$ im Intervall $-\infty \leq a \leq x \leq b \leq \infty$. Außerdem sei

$$\langle f(x), g(x) \rangle = \int_a^b f(x)g(x)w(x)dx = \mathbb{E}\{f(x)g(x)\}$$

das Skalarprodukt auf dem linearen Raum $L^2[a, b]$ aller reellen Funktionen f, für die das Integral $\int_a^b f(x)^2 w(x)dx$ existiert und endlich ist.

Die diskrete Zufallsvariable Y, deren Wahrscheinlichkeitsfunktion durch

$$prob(Y = x_j) = w_j \ > 0 \quad (j = 1, \ldots, N)$$

gegeben ist, hat die gleichen r-ten Momente wie X für $r = 0, \ldots, 2N-1$ dann und nur dann, wenn die $x_1, \ldots, x_N$ die Nullstellen des Polynoms $P_N(x)$ vom Grad N darstellen, das durch folgende sogenannte „Drei-Term-Rekursion" definiert wird:

$$\begin{aligned}
P_{-1}(x) &\equiv 0 \\
P_0(x) &\equiv 1 \\
P_{i+1}(x) &= (x - \delta_{i+1})P_i(x) - \gamma_{i+1}^2 P_{i-1}(x), \qquad i \geq 0, \qquad (2.1)
\end{aligned}$$

wobei gilt:

$$\delta_{i+1} = \mathbb{E}\{XP_i^2(X)\}/\mathbb{E}\{P_i^2(X)\}, \qquad i \geq 0, \qquad (2.2)$$

$$\gamma_{i+1}^2 = \begin{cases} 0 & , \text{ für } i = 0 \\ \mathbb{E}\{P_i^2(X)\}/\mathbb{E}\{P_{i-1}^2(X)\} & , \text{ für } i \geq 0. \end{cases} \qquad (2.3)$$

Die Wahrscheinlichkeiten $w_1, \ldots, w_N$ sind gegeben durch das lineare Gleichungssystem

$$\sum_{j=1}^{N} P_k(x_j)w_j = \begin{cases} 1, & k = 0 \\ 0, & k = 1, \ldots, N-1. \end{cases}$$

Mithilfe der Wahrscheinlichkeitsfunktion von Y kann man nun (numerisch) die r-ten Momente einer gegebenen Funktion $h(x)$ erhalten:

$$\int_a^b h^r(x)w(x)d(x) \cong \sum_{j=1}^{N} h^r(x_j)w_j. \tag{2.4}$$

2.1.2 Theorem 2

Das zweite Theorem Luceños geht einen Schritt weiter: Hiermit können nicht nur Aussagen über die Momente, sondern auch über die Verteilungsfunktion der neu erzeugten Zufallsvariablen getroffen werden.

Zu einer beliebigen Aufteilung des Intervalls $[a, b]$ in M Teilintervalle kann man nämlich erreichen, dass die Verteilungsfunktionen der stetigen Zufallsvariablen X und der diskreten Y an den Grenzen dieser Teilintervalle übereinstimmen. Dazu sei $a = c_0 < \ldots < c_M = b$ eine Partition des Intervalls $[a, b]$, außerdem die Zufallsvariable X definiert wie oben. Als Gewichtsfunktion der Diskretisierung mit N Punkten im Intervall $[c_{i-1}, c_i]$ benutzt man

$$w(x)/I_i,$$

wobei $c_{i-1} \leq x \leq c_i$ gilt, sowie $I_i = \int_{c_{i-1}}^{c_i} w(x)dx < 1$. Es ergeben sich für jedes Intervall i die Punkte $x_{i1}, \ldots, x_{iN}$ mit ihren Gewichten $w_{i1}, \ldots, w_{iN}$. Auf dem ganzen Intervall $[a, b]$ definiert man nun die Zufallsvariable Y, die aus den Werten x_{ij} und den dazugehörigen Wahrscheinlichkeiten $I_i w_{ij}$ besteht.

Für $i = 1, \ldots, M$ und $j = 1, \ldots, N$ stimmen auch hier die r-ten Momente von X und Y bis zu $r = 2N - 1$ überein. Außerdem entsprechen sich die Werte der Verteilungsfunktionen von X und Y zumindest an den Partitionsstellen $a = c_0, \ldots, c_M = b$.

2.2 Implementierung

Bei der Implementierung benutze ich das statistische Programmpaket `R`, das selbst bereits vordefinierte Funktionen zu diesem Thema besitzt.

2.2.1 Klassischer Fall

Für den Fall, dass es sich bei der Gewichtsfunktion um eine „klassische“ Dichtefunktion handelt, existieren bereits bekannte Polynome P_j, die es überflüssig machen, den beschwerlichen Weg der oben beschriebenen Rekursion zu gehen. Für den Fall der Normalverteilung beispielsweise kann man auf die Gauß-Hermite Polynome zur Gewichtsfunktion $W(x) = e^{-x^2}$ auf dem Intervall $-\infty < x < \infty$ zurückgreifen:

$$\begin{aligned} H_{j+1} &= 2xH_j - 2jH_{j-1} \\ & \quad \text{mit } H_1 = 1 \end{aligned} \tag{2.5}$$

Gegeben die Nullstellen $x_j^{(GH)}$ und Gewichte $w_j^{(GH)}$ des Gauß-Hermite Polynoms N-ten Grades, erhält man für die Normalverteilung die Zufallsvariable Y und ihre Wahrscheinlichkeitsfunktion $w(Y)$ durch (siehe [9], S. 347)

$$\begin{aligned} Y_j &= \mu + \sigma x_j^{(GH)} \sqrt{2} \\ w(Y_j) &= \frac{1}{\sqrt{\pi}} w_j^{(GH)}. \end{aligned}$$

Für die Gamma-, die Beta- und die Gleichverteilung gibt es in Gestalt von Gauß-Laguerre, Gauß-Jacobi und Gauß-Legendre ebenfalls bekannte Polynome, deren Ergebnisse leicht vom Allgemeinen auf den Fall der speziellen Verteilung übertragen werden können:

Verteilung	Zufallsvariable	Wahrscheinlichkeit	Polynom
$\text{gamma}(t, \alpha)$	$\frac{1}{\alpha} x_j^{(GLa)}$	$\frac{1}{\Gamma(t)} w_j^{(GLa)}$	Gauß-Laguerre
$\text{beta}(\alpha, \beta)$	$\frac{1}{2}(1 + x_j^{(GJ)})$	$\frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha,\beta)}$	Gauß-Jacobi
$U(a, b)$	$a + \frac{b-a}{2}(1 + x_j^{(GLe)})$	$\frac{1}{2} w_j^{(GLe)}$	Gauß-Legendre

In `R` existieren hierzu in dem Paket `statmod` die Funktionen `gauss.quad` und `gauss.quad.prob`. Ersteres gibt für ein vorgegebenes N, sowie eventuell vorgegebene α und β die Nullstellen und Gewichte der Gaußschen Quadratur (also der Hermite, Laguerre, Jacobi oder Legendre-Polynome) aus, `gauss.quad.prob` transformiert diese nach obiger Tabelle gleich in die entsprechenden Werte (Zufallsvariable Y und $w(Y)$) für die angegebenen Verteilung.

Wichtig werden diese Funktionen für das Theorem 1, bei der die Gaußsche Quadratur auf das Gesamtintervall des Trägers angewendet und somit die

Gewichtsfunktion nicht verändert wird. Allerdings beschränkt sich die Zahl der möglichen Verteilungen auf die oben erwähnten „klassischen“ Fälle der Normal, Beta-, Gamma- und Gleichverteilung. Für den nichtklassischen Fall, sprich einer Verteilung aus einer anderen Familie oder nichtklassischen Gewichten, wie sie in Theorem 2 benutzt werden, gibt es keine entsprechende Implementierung in `R`.

2.2.2 Nichtklassischer Fall

Im zweiten Theorem Luceños werden für die Gaußsche Quadratur in den Einzelintervallen die Gewichtsfunktionen $w(x)/I_i$ verwendet, die sich keiner klassischen Verteilungsfamilie mehr zuordnen lassen. Es ist also nötig, die Polynome und ihre Nullstellen z.B. über die Drei-Term-Rekursion (2.1) zu berechnen. Das Integral $I_i = \int_{c_{i-1}}^{c_i} w(x)dx$ stellt dabei, falls $w(x)$ zu einer der vier obigen Verteilungsfamilien gehört, kein numerisches Problem dar, da die entsprechenden Stammfunktionen alle bekannt sind. Bei der sukzessiven Berechnung der Polynome steht man dann allerdings endgültig vor dem Problem der Gewichtsfunktionen aus nichtklassischen Verteilungsfamilien: Es stellt sich die Frage, wie die Skalarprodukte aus (2.2) und (2.3) bestimmt werden können.

Gautschi verwendet in älteren Arbeiten zur Berechnung der Skalarprodukte im Falle nichtklassischer Gewichtsfunktionen das Wissen um die Momente dieser Gewichtsfunktion (siehe dazu [5] und [6]). Sein entsprechender Algorithmus zur Berechnung der Nullstellen bei nichtklassischen Gewichten ist allerdings numerisch instabil und liefert für die δ_i und γ_i^2 Ergebnisse, die erst mit $N \to \infty$ gegen die tatsächlichen Werte konvergieren. Bei $M \cdot N$ zu berechnenden Gauß-Quadraturen ist diese Methodik somit nicht praktikabel.

Eine mathematisch saubere und zugleich numerisch stabile Lösung für das Problem nichtklassischer Gewichte bieten Sack und Donavan ([15]) an. Diese basiert auf dem Wissen um die sogenannten „modifizierten Momente“ $\nu_j = \int_a^b \pi_j(x)w(x)dx$ einer Menge von orthogonalen Polynomen π_j. Mit deren Hilfe entwickelten sie eine numerisch stabile Methode, die Koeffizienten γ_j^2 und δ_j der gesuchten Rekursion zu berechnen, die von Wheeler ([21]) zu einem $O(N^2)$ Algorithmus verfeinert wurde. Die π_j müssen dabei folgende

Rekurrenz, ähnlich zu (2.1), erfüllen:

$$\begin{aligned}
\pi_{-1}(x) &\equiv 0 \\
\pi_0(x) &\equiv 1 \\
\pi_{j+1}(x) &= (x-\alpha_j)\pi_j(x) - \beta_j\pi_{j-1}(x) \qquad j = 0, 1, \ldots, 2N-1
\end{aligned}$$

Der darauf aufbauende Algorithmus hat mit der Dreitermrekursion aus (2.1) nichts mehr zu tun, die γ_i und δ_i können ohne die Skalarprodukte (2.2) und (2.3) berechnet und somit das Problem umgangen werden.

In einer späteren Arbeit ([7]) bietet Gautschi ein ganzes Paket an Möglichkeiten zur Lösungen des Problems nichtklassischer Gewichte. Unter Anderem greift er nochmals den Algorithmus von Sack und Donavan bzw. Wheeler auf. Als weitere Möglichkeit schlägt er vor, die Skalarprodukte lediglich durch eine geeignete Quadratur zu approximieren, was im Wesentlichen auch der Weg dieser Diplomarbeit ist.

Denn im Interesse dieser Arbeit liegen weniger die exakte Berechnung eines Integrals, als vielmehr das Finden der Stützpunkte und Gewichte der Gaußschen Quadratur, also der neuen Zufallsvariablen. Insofern erscheint es aus Effizienzgründen sinnvoll, bei Berechnungen, die lediglich das Erzeugen der neuen Zufallsvariablen zum Ziel haben, auf im Detail komplizierte Verfahren, wie das der modifizierten Momente, zu verzichten. Stattdessen verwende ich die bereits in `R` implementierte Funktion `integrate`, um die entsprechenden Integrale numerisch zu berechnen. Diese Funktion verwendet die sogenannten „Quadpack“ Routinen `dqags` und `dqagi` von Piessens und deDoncker-Kapenga (siehe [22], [11]), welche zur numerischen Berechnung ebenfalls angepasste Quadratur-Verfahren (in Verbindung mit Extrapolation durch den Epsilon Algorithmus zur Konvergenzbeschleunigung) benutzen. Genau genommen werden auch bei `integrate` Stützstellen nach Gauß-Legendre verwendet, nämlich durch Anwendung der sogenannten 21-Punkt Kronrod Regel (Näheres dazu siehe [11]). Bei dieser Funktion greift `R` auf die entsprechenden in `C` implementierten Algorithmen zurück, was zu sehr effektiven und schnellen Berechnungen führt. Die Ergebnisse stellen selbst wieder approximative, weil numerische, Berechnungen eines Integrals dar und stehen somit im Einklang mit dem oben beschriebenen Vorgehen nach Gautschi ([7]).

Auf den Punkt gebracht, verwende ich zur approximativen Berechnung eines Integrals wiederum approximative Berechnungen von Intervallen, gefunden allerdings durch einen anderen, bereits implementierten Algorithmus. Gerechtfertigt wird diese einfachere Vorgehensweise, wie bereits erwähnt, auch dadurch, dass, speziell bei Theorem 2, das Interesse dieser Arbeit nicht dem Wert des Integrals gilt, sondern den Stützpunkten und Gewichten für die Gaußsche Quadratur. Die hier vorgestellte Implementierung ist keine Funktion, um ein Integral effizient zu berechnen, sondern eine Funktion zum Finden einer neuen, diskreten Zufallsvariablen anstelle einer stetigen.

Das Integral $I_i = \int_{c_{i-1}}^{c_i} w(x)dx$ wird in der von mir implementierten Funktion entweder über die Werte der Stammfunktionen bei den vier obigen Verteilungsfamilen berechnet oder im Falle einer völlig unbekannten Verteilung der Effizienz halber über `integrate`.

Die eigentlich gesuchte diskrete Zufallsvariable ist am effizientesten mit folgender Tridiagonalmatrix, bestehend aus den γ_i^2 und δ_i der Rekursion, zu berechnen:

$$J_N = \begin{pmatrix} \delta_1 & \gamma_2^2 & & & \\ \gamma_2^2 & \delta_2 & \cdot & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & \cdot & \gamma_N^2 \\ & & & \gamma_N^2 & \delta_N \end{pmatrix} \tag{2.6}$$

Die Nullstellen des N-ten Orthogonalpolynoms P_N sind die Eigenwerte dieser Matrix J_N und für die Gewichte w_i der Gaußschen Quadratur gilt:

$$w_i = (v_1^{(i)})^2, \quad i = 1, \ldots, N,$$

falls $v^{(i)} = (v_1^{(i)}, \ldots, v_N^{(i)})$ Eigenvektor zum Eigenwert x_i von J_N ist (siehe dazu, auch für den Beweis [16], S.179f.).

Die von mir implementierte Funktion `gauss.quad.prob.inter` (siehe Anhang) ermittelt die Zufallsvariable Y mithilfe des zweiten Theorems von Luceño. Übergeben werden N als der Grad des Polynoms, M als die Anzahl der Intervalle, in die der Träger aufgeteilt wird, sowie die Dichtefunktion der ursprünglichen, stetigen Verteilung. Benutzt wird dabei die Funktion `discretization`, die ebenfalls im Anhang zu finden ist. Sie liefert zusammen mit der Hilfsfunktion `polynom` für eine beliebig gegebene Gewichtsfunktion $w(x)$ mithilfe der Drei-Term-Rekursion (2.1) die Stützstellen und Gewichte der Gauß Quadratur zurück. Wie beschrieben, werden die dabei benötigten Integrale, speziell die Skalarprodukte (2.2) und (2.3), mit der Funktion `integrate` und die Nullstellen und Gewichte selbst durch eine Eigenwertzerlegung von (2.6)[1] berechnet.[2]

An `gauss.quad.prob.inter` werden außerdem Parameter übergeben, die die Größe und Gestalt der Intervalle bestimmen. Hierbei gibt es folgende Möglichkeiten: eine äquidistante Aufteilung des Trägers zwischen a und b (es wird also bei der Intervallaufteilung mit Ausnahme der Trägergrenzen keine Information über das Aussehen der stetigen Verteilung übergeben), eine selbstgewählte Partition, sowie eine an die Verteilungsfunktion der stetigen Zufallsvariable angepasste Aufteilung, sofern diese aus einer der oben erwähnten klassischen Familie kommt. Bei der dritten Möglichkeit der Partition teilt das Programm die Quantile der stetigen Verteilung äquidistant auf und ermittelt daraus die entsprechenden Intervallgrenzen. Verwendet wird dabei eine der Funktionen `qnorm`, `qbeta`, `qgamma` oder `qunif`. Die Aufteilung der angepassten Intervalle erscheint als die effektivste, da bei ihr in Gegenden, in denen die ursprüngliche Dichte groß ist, die Intervalle kleiner sind und dementsprechend hier mehr neue Zufallsvariablen erzeugt werden als mit äquidistanten Intervallen, bei denen die Intervalle in „unwichtigen“ Gegenden des Trägers genauso groß sind wie in „wichtigen“. Allerdings setzt diese Methode bereits explizites Wissen über die Verteilung der stetigen Variable voraus.

Zudem kann es Probleme bei Trägern geben, die von $-\infty$ bis $+\infty$ gehen, speziell also bei der Normalverteilung: es kann bei zu großer Standardabweichung passieren, dass `integrate` die Integrale für die Gewichtsfunktion in den äußeren Intervallen nicht mehr berechnet. Aus diesem Grund setzt die Funktion speziell für die Normalverteilung die Grenzen des Trägers, also die äußeren beiden Stützstellen x_1 und x_N mit Hilfe von σ heuristisch auf $x_2 - 10\sigma$ bzw. $x_{N-1} + 10\sigma$ fest.

2.2.3 Veranschaulichung: Normal- bzw. Gammaverteilung

Im folgenden Abschnitt werde ich am Beispiel der Standardnormal- sowie der Gamma(2,1)-Verteilung die Wirkungsweise von Luceños Theoremen veranschaulichen. Die Ausdrücke

```
gauss.quad.proc(15,"normal")
```

[1] Die Eigenwertzerlegung erfolgt durch die `R`-Funktion `eigen`, die die `LAPACK`-Funktionen `DSYEVR`, `DGEEV`, `ZHEEV` und `ZGEEV` (siehe [24]) verwendet.

[2] Das Problem der eventuellen numerischen Instabilität von Drei Term Rekursionen in Vorwärtsrichtung (siehe [3]) ist hier vernachlässigbar, wenn, wie auch von Luceño vorgeschlagen, bei Theorem 2 nur Polynome kleinen Grades, also $N = 3$ oder $N = 4$ verwendet werden.

```
gauss.quad.proc(15,"gamma",alpha=2,beta=1)
```

liefern die Ergebnisse zu Theorem 1, nämlich jeweils als Nullstellen der Polynome 15ten Grades die Zufallsvariablen Y_{norm} und Y_{gamma} mit einem aus 15 Stützstellen bestehenden Träger, deren Wahrscheinlichkeitsfunktionen in Abbildung (2.1) dargestellt sind.

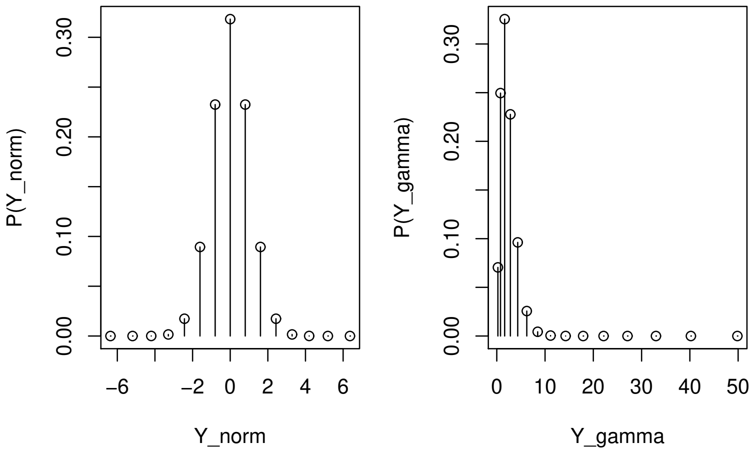

Abb. 2.1: Stützstellen und Gewichte der Gaußschen Quadratur der $N(0,1)$ sowie der Gamma(2,1)-Verteilung mit $N = 15$

Diese Variablen besitzen nun die exakt gleichen ersten $2 \cdot N - 1 = 29$ Momente wie die ursprünglichen Verteilungen. Hier liefert die einfache Summation

$$M_k = \sum_{i=1}^{15} Y_i^k * P(Y_i)$$

das jeweils k-te Moment. Bei der exakten Berechnung der Momente der stetigen Verteilungen ist der Rechenaufwand dagegen ungleich größer: Für die Normalverteilung existiert noch eine einfache rekursive Formel zu deren Bestimmung (siehe [10]), bei der Gammaverteilung muss man auf die momentenerzeugende Funktion und ihre Ableitungen zurückgreifen. Beide Verfahren werden jedoch für große N relativ rechenaufwändig.[3]

[3] Selbsverständlich gibt es auch andere numerische Verfahren, mit denen man die Momente sehr effektiv berechnen kann, wie beispielsweise die Funktion `integrate`. Speziell auf dieses Verfahren und die kleinen, aber feinen Unterschiede zur hier beschriebenen Methode werde ich in Kapitel 4 eingehen.

Die Abbildungen 2.2 und 2.3 zeigen für die Standardnormal- sowie die Gamma(2,1)-Verteilung die Verteilungsfunktionen der nach Luceños zweitem Theorem erzeugten Variable Y zusammen mit der ursprünglichen Verteilung. N ist dabei 3 und M, also die Anzahl der Intervalle, variiert zwischen 2, 5 und 20. Die Plots der ersten Zeile wurden dabei jeweils mit angepassten, die in der zweiten mit äquidistanten Intervallen erzeugt.

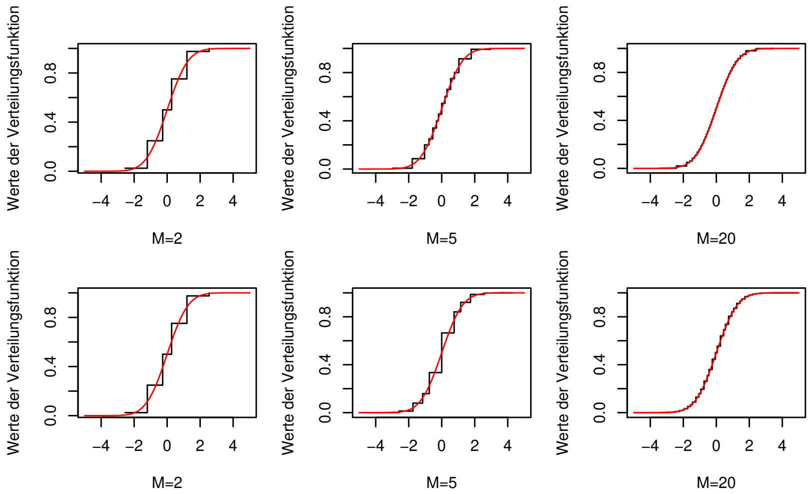

Abb. 2.2: Diskretisierung der Standardnormalverteilung nach Theorem 2 für $N = 3$ mit und ohne angepassten Intervallen

Man sieht sehr deutlich, dass sich mit wachsender Anzahl von Intervallen die Verteilungsfunktion der diskreten Variable immer besser an die der stetigen anpasst. Wie nicht anders zu erwarten, gelingt diese Approximation bei den angepassten Intervallen besser. Man muss jedoch beachten, dass bei $M = 2$ mit beiden Methoden die gleichen Intervalle gewählt werden. Ein deutlicher Unterschied fällt dagegen bei $M = 5$ auf. Neben den Verteilungsfunktionen, die sich Stück für Stück immer mehr angleichen, besitzen beide Zufallsvariablen, freilich nur approximativ, die gleichen ersten $2 \cdot N - 1 = 5$ Momente. Approximativ deswegen, weil für das Finden der Werte, wie oben beschrieben, selbst Approximationen verwendet wurden.

Es mag im ersten Moment verwirrend erscheinen, dass die Verteilungsfunktionen an keiner Sprungstelle der diskreten Variable übereinstimmen. Zunächst sieht das vielleicht nach einem Widerspruch zu der Aussage des zweiten Theorems aus, die besagt, dass an den Intervallgrenzen die Verteilungsfunktionen die gleichen Werte besitzen. Jedoch muss man hier klar unterscheiden zwischen den Grenzen der Intervalle und den Stützstellen der neuen Variable:

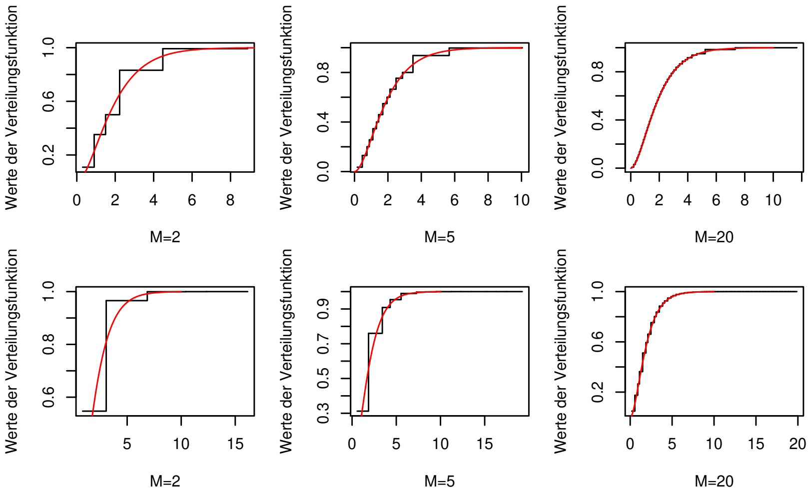

Abb. 2.3: Diskretisierung der Gamma(2,1)-Verteilung nach Theorem 2 für $N = 3$ mit und ohne angepassten Intervallen

bei zweiteren befinden sich in diesen Graphiken die Sprungstellen. In Abbildung 2.2 beispielsweise liegt in den Bildern links ($M = 2$) die Grenze zwischen beiden Intervallen bei 0. An dieser Stelle stimmen auch die Werte der Verteilungsfunktionen überein.

Verwendet man die Funktion `integrate` gleich selbst zum Berechnen der Momente in der Form $\int_a^b x^k f(x)dx$, erhält man diese zum einen unkomplizierter und zum anderen auch genauer. Das Hauptaugenmerk bei Luceños zweitem Theorem liegt allerdings nicht auf den Momenten der neu gefundenen Variable, sondern auf den Eigenschaften ihrer Verteilungsfunktion.

3. Diskretisieren von stetigen Verteilungen: Anpassungsgüte

Luceños zweites Theorem besagt, dass die durch Diskretisierung über den in Intervalle aufgeteilten Träger gefundene neue Zufallsvariable zumindest an den Intervallgrenzen die gleiche Verteilungsfunktion besitzt wie die ursprüngliche Variable. Der Schritt liegt nahe zu fragen, ab welcher Anzahl an Intervallen die Verteilungsfunktionen derart ähnlich sind, dass die diskrete Zufallsvariable als Approximation der stetigen angesehen werden kann. Wie in den Abbildungen 2.2 und 2.3 bereits gut zu sehen war, gleichen sich die jeweiligen Kurven mit wachsendem M immer weiter an, bis sie rein optisch kaum mehr zu unterscheiden sind.

3.1 Kolmogorov-Smirnov Test

In den folgenden Untersuchungen wende ich die Diskretisierungstechnik auf den Fall der Standardnormalverteilung an und überprüfe die neu gefundenen Zufallsvariable auf Ähnlichkeit mit der ursprünglichen. Ein dabei möglicher anzuwendender, sogenannter Goodness of Fit-Test ist der nichtparametrische Kolmogorov-Smirnov Test, mit dem man feststellen kann, ob die Variable einer Stichprobe einer erwarteten Verteilung (nicht) angehört. Die Statistik für die zweiseitige Fragestellung

$$H_0 : F(x) = F_0(x), \forall x \in \mathbb{R}$$

stützt sich dabei auf den absoluten Abstand der empirischen Verteilungsfunktion von gezogenen Zufallszahlen zu der unter H_0 angenommenen Verteilungsfunktion:

$$D_n = \sup_x |F_n(x) - F_0(x)| \tag{3.1}$$

Je größer dieser Abstand ist, desto eher wird die Nullhypothese abgelehnt. Bei einer genügend großen Anzahl M an Intervallen sollte es also für den

Test nicht mehr möglich sein, die approximierte Zufallsvariable als solche zu erkennen.

Es ist klar, dass der Test an dieser Stelle etwas konstruiert erscheint: Aus einer eigentlich bekannten Verteilung werden künstlich Zufallsvariablen gezogen, um daraus abzuleiten, ob sie Ähnlichkeiten mit der Originalverteilung aufweist. Man könnte auch einfach nur einen Abstand zwischen beiden Verteilungen, ähnlich zu (3.1), betrachten. Der Vorteil der Methode über die Stichproben besteht allerdings darin, dass es für die entsprechenden Testgrößen bekannte Verteilungen und somit Signifikanzniveaus gibt. Die Qualität der Unterschiede lässt sich also messen und vergleichen.

Bekanntlich wirken in der Neymann-Pearsonschen Testtheorie dieser und die folgenden Tests auf Verteilung allerdings nur in eine Richtung, sprich bei der Frage, ab wann man mit vorgegebener Irrtumswahrscheinlichkeit α die Nullhypothese $H_0: \ F(x) = F_0(x), \forall x \in \mathbb{R}$ ablehnen kann. Die umgekehrte, und hier interessierende Fragestellung, ob nämlich mit bestimmter Sicherheit $F(x) = F_0(x)$ gilt, ist somit schwierig zu beantworten. Über die Anwendung verschiedener Tests, die alle unterschiedliche Schwerpunkte setzen, versuche ich im Folgenden dennoch eine Antwort darauf zu finden, ab wann die Diskretisierung der Normalverteilung als Äquivalent zu dieser angesehen werden kann.

Abbildung 3.1 und Abbildung 3.2 zeigen graphisch die Ergebnisse einer entsprechenden Untersuchung: Für $M = 1, \ldots, 50$ Intervalle und Grad $N = 3$ wurde dabei die Standardnormalverteilung diskretisiert und aus der sich ergebenden Variable eine Stichprobe der Größe n mit Zurücklegen gezogen, wobei n entweder 50, 100, 1000 oder 10000 war. Diese wurde daraufhin mit Hilfe des Kolmogorov-Smirnov-Tests auf Ähnlichkeit mit der $N(0,1)$-Verteilung überprüft (je größer die Testgröße, desto unähnlicher der Normalverteilung). Zur Kontrolle und Einordnung der Ergebnisse wurde die gleiche Anzahl an Stichproben aus der Standardnormalverteilung gezogen und ebenfalls „überprüft“. Für jedes M führte der Rechner dieses Stichproben-ziehen und -testen 500 mal aus, um die Ergebnisse zu validieren und die entsprechenden Mittelwerte der Testgrößen mit Konfidenzintervallen zu versehen.

Eingetragen in die Graphiken sind die Mittelwerte der Testgrößen der diskreten Variable zusammen mit den Grenzen des Konfidenzintervalls, sowie die entsprechenden Werte der Standardnormalverteilung. Die horizontalen Linien stellen die Grenze zum Ablehnbereich dar für $\alpha = \{0.1, 0.05, 0.025, 0.01, 0.005\}$. Zu beachten ist dabei, dass für $\alpha = 0.005$ der Wert am größten ist, da er auch die höchste Sicherheit gegenüber dem fälschlichen Ablehnen der Nullhypothese bietet. In der umgekehrten Ausrichtung des Testes, wie hier verwendet, stellt er somit die geringste Hürde und das kleinste Sicherheitsniveau dar. Außerdem ist klar, dass bei einer 500 maligen Wiederholung des Tests die eingetragenen Signifikanzniveaus nicht eigentlich das messen, was sie bei einem einzigen Test messen sollten. Wie bereits oben andeutungsweise erwähnt, dienen sie hier, in dieser konstruierten Situation, hauptsächlich der Einordnung und der Beantwortung der Frage: ab wann kann man in etwa damit rechnen, dass eine Stichprobe aus der diskreten Verteilung bei dem entsprechenden Test nicht mehr als nicht aus der Originalverteilung stammend abgelehnt wird?

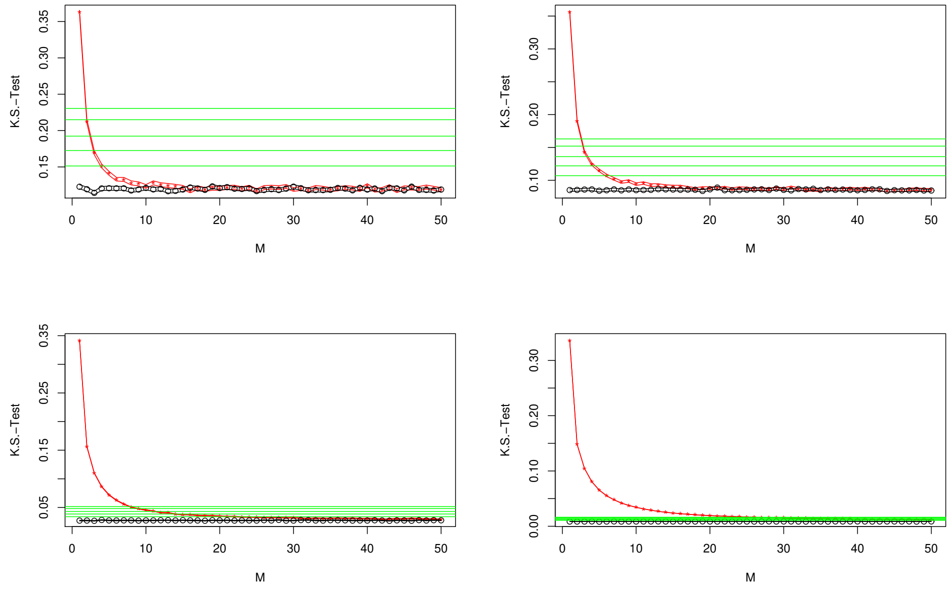

Abb. 3.1: Approximationsgüte der Standardnormalverteilung: wiederholter Kolmogorov-Smirnov Test bei $n = 50, 100, 1000$ oder 10000 gezogenen Zufallszahlen mit angepassten Intervallen

Zunächst fällt der deutliche Unterschied zwischen der Methode mit angepassten und der ohne angepasste Intervalle auf: wie nicht anders zu erwarten, ist erstere deutlich effektiver. Zieht man nur wenige Zufallsvariablen, so kann der Test hier schon bei $M \geq 15$ nicht mehr zwischen der tatsächlichen und der approximierten Normalverteilung unterscheiden. Schon sehr viel früher ist der Test bereits unter allen angegebenen Ablehnschranken. Bei größeren Stichproben werden allerdings mehr Intervalle benötigt: Für $n = 1000$ benötigt man in etwa 30, für $n = 10000$ wohl sogar 40 oder mehr Intervalle, um

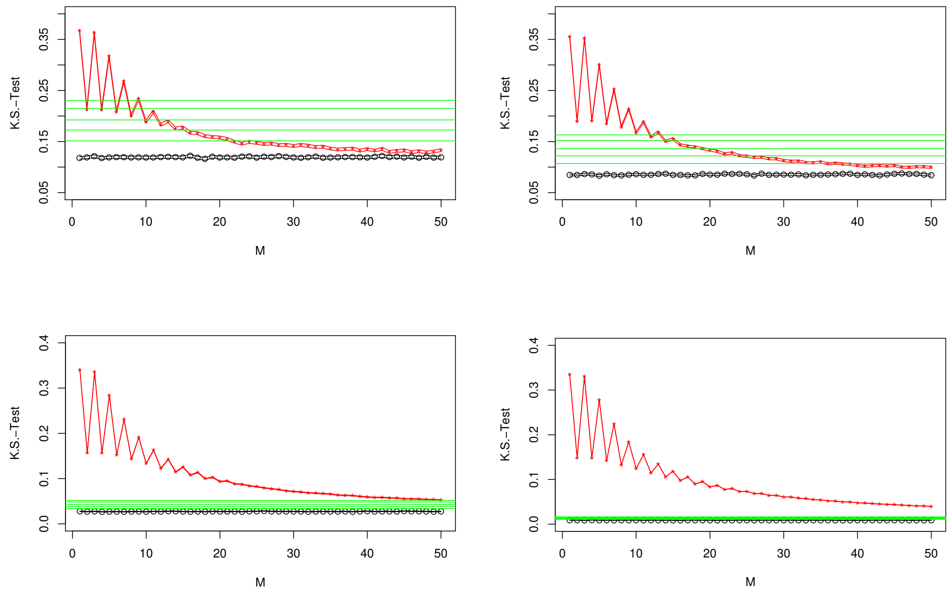

Abb. 3.2: Approximationsgüte der Standardnormalverteilung: wiederholter Kolmogorov-Smirnov Test bei $n = 50, 100, 1000$ oder 10000 gezogenen Zufallszahlen ohne angepasste Intervalle

unter allen Schranken und möglichst nah bei der Originalverteilung zu sein.

Bei der Methode mit gleichbleibenden Intervallgrößen springt das deutliche Zick-Zack am Anfang der Kurven ins Auge: hier macht sich einfach die gewählte Trägerlänge bemerkbar. Als untere Grenze wählte ich hier -10, als obere Grenze 10, um möglichst die ganze Wahrscheinlichkeitsmasse unterzubringen. Teilt man dieses große Intervall in drei gleich große Teilintervalle, um die Normalverteilung zu approximieren, werden viele Stützpunkte der Verteilung in die Ränder verlagert, bei einer geraden Anzahl von Intervallen kommen die mittleren Bereiche der Verteilung besser zur „Geltung“. Außerdem werden die Testwerte der Normalverteilung in keinem Fall erreicht, für sehr großes n kommt man mit den approximierten Werten mit keinem M unter eine der Ablehngrenzen, was bedeutet, dass man diese Stichproben immer als nicht zur Normalverteilung gehörend ablehnen müsste.

3.2 Test nach A. Weichselberger

Ein weiterer nichtparametrischer Anpassungstest wurde von A. Weichselberger entwickelt (siehe [19]). Er benutzt die Werte des einseitigen Kolmogorov-Smirnov Tests

$$\begin{aligned} D_n^+ &= \sup_x (F_n(x) - F_0(x)) \quad \text{und} \\ D_n^- &= \sup_x (F_0(x) - F_n(x)). \end{aligned}$$

und verknüpft sie in seiner Teststatistik:

$$K_n = D_n^+ - D_n^-.$$

Dieser Test reagiert insbesondere auf Verschiebungen von der unter H_0 angenommenen Verteilung empfindlicher als der von Kolmogorov-Smirnov. Auch wenn dies hier nicht Ziel der Untersuchung sein soll, so ist der Blick auf verschiedene Teststatistiken doch von Vorteil, um die Ergebnisse zu validieren.

Abbildung 3.3 zeigt die entsprechenden Testergebnisse bei Stichproben der Größe 50 bzw. 1000 für die Methode mit und ohne angepasste Intervalle. Zu beachten sind bei diesen Graphiken die zwischen beiden Methoden unterschiedlichen Niveaus der X-Achsen. Zudem sind nicht die tatsächlichen Werte der Teststatistik K_n notiert, sondern die Werte der Transformation $\sqrt{n}K_n$, da für diese die Quantile in [19] tabelliert sind. Es ist, wie auch schon beim Kolmogorov-Smirnov Test, deutlich zu sehen, dass zum einen immer nach gewisser Zeit sämtliche angegebenen Konfidenzniveaus unterschritten werden und zum anderen die Methode mit angepassten Intervallen erwartungsgemäß schneller zu einem befriedigenden Ergebnis führt. Die horizontalen Linien stellen dabei die Ablehnschranken für $\alpha = \{0.1, 0.05, 0.025, 0.01\}$ dar, außerdem sind wieder zum Vergleich die Ergebnisse der Tests für Stichproben aus der Standardnormalverteilung eingetragen.

3.3 Anderson-Darling Test

Als dritten Goodness of Fit-Test verwende ich hier den sogenannten Anderson-Darling Test, dessen Teststatistik zwar ebenfalls auf dem Abstand der empirischen zu einer angenommen Verteilung basiert, der aber dennoch andere Akzente setzt und somit als zusätzlicher Test zur Validierung Sinn ergibt. Im Gegensatz zu den vorhergehenden Tests ist er nämlich nicht verteilungsfrei,

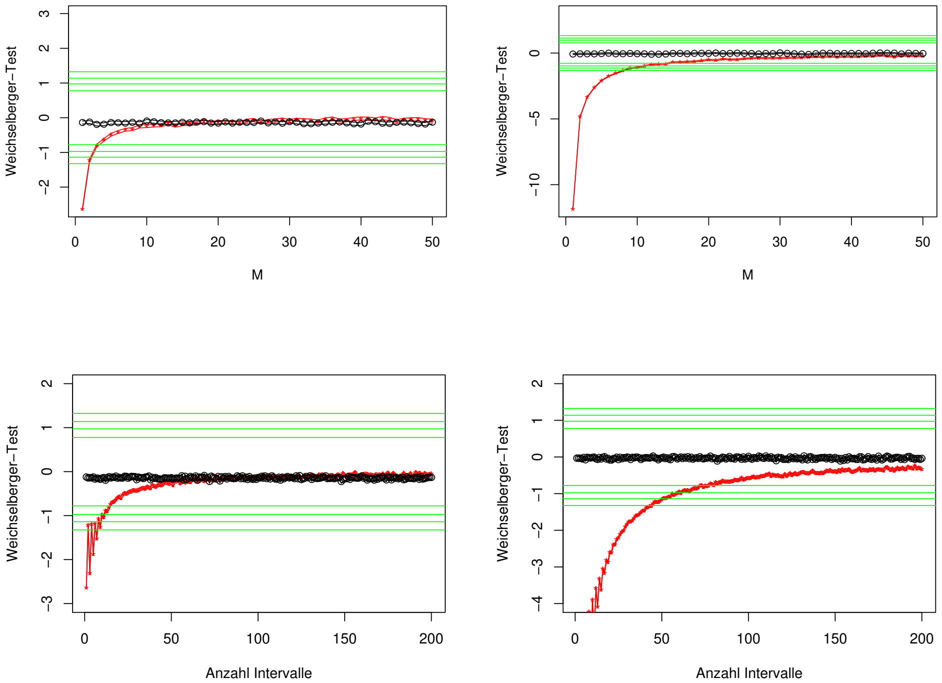

Abb. 3.3: Approximationsgüte der Standardnormalverteilung: wiederholter Weichselberger Test bei $n = 50$ bzw $n = 1000$ gezogenen Zufallszahlen mit (oben) und ohne (unten) angepassten Intervallen

sondern seine Werte sind von der jeweils angenommenen Verteilung abhängig, was ihn bei richtiger Verteilungsannahme zu einem mächtigeren Test macht. Außerdem gewichtet er die Enden der Verteilung stärker als die von Kolmogorov-Smirnov und A. Weichselberger. Seine Teststatistik lautet wie folgt:

$$A^2 = -N - \frac{1}{N}\sum_{i=1}^{n}(2i-1)\Big[ln(F(x_i)) + ln(1 - F(x_{N+1-i}))\Big],$$

wobei die x_i die geordneten Daten der Stichprobe darstellen.

Die Abbildungen 3.4 und 3.5 zeigen, wie die vorhergehenden Graphiken, für verschiedene Stichprobengrößen ($n = \{50, 100, 1000, 10000\}$) und M Intervalle jeweils 500 Durchläufe des Tests. Dabei sind in Abbildung 3.4 die Intervalle in ihrer Größe an die Normalverteilung angepasst, in Abbildung 3.5 nicht. Zur Kontrolle wurden auch wieder jeweils Stichproben aus der „wah-

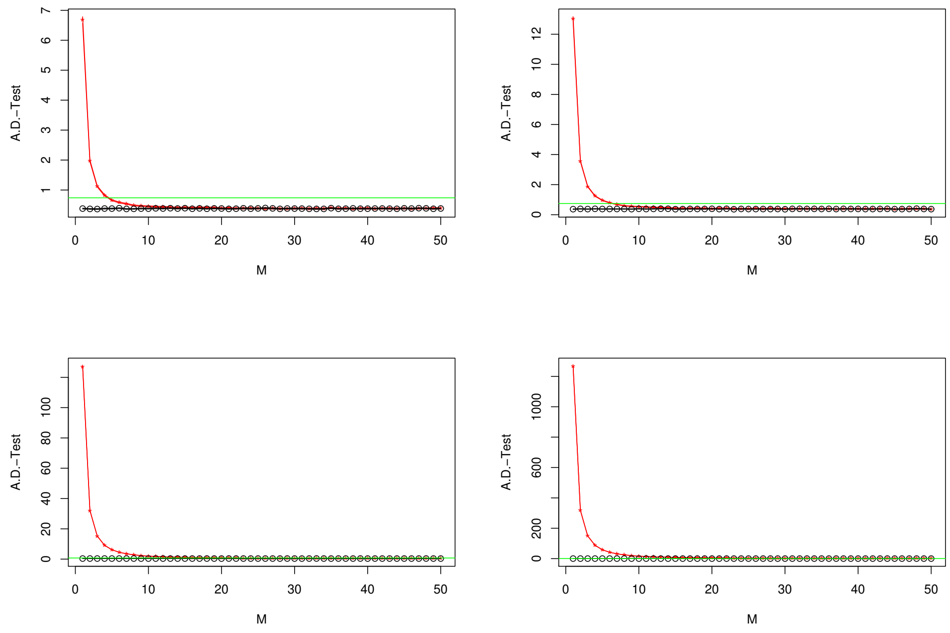

Abb. 3.4: Approximationsgüte der Standardnormalverteilung: wiederholter Anderson-Darling Test bei $n = 50, 100, 1000$ oder 10000 gezogenen Zufallszahlen mit angepassten Intervallen

ren“ Verteilung gezogen und die Teststatistiken berechnet. Die horizontale Linie stellt die Grenze des 5%-Ablehnbereiches dar.

Vom Prinzip her wiederholen sich die aus den beiden anderen Tests bekannten Muster. Bei kleineren Stichproben werden hier etwas mehr Intervalle benötigt, um die jeweilige Teststatistik unter die 5%-Hürde zu „drücken“, bei größeren etwas weniger. Das sind jedoch lediglich testtypische kleinere Schwankungen, am grundsätzliche Ergebnis ändert sich nichts.

Die aus der „wahren“ Verteilung gezogenen Zufallszahlen dienen, wie bereits erwähnt, zur Kontrolle der Approximation und zur Beantwortung der Frage: wie sehr ähneln sich die Teststatistiken und ab wann sind sie nicht mehr voneinander zu unterscheiden?

Lässt man in den vorhergehenden Graphiken die Werte mit niedrigem M weg, so ändert sich der Maßstab der Betrachtung und die Unterschiede zu bzw. Gemeinsamkeiten mit der Originalverteilung werden deutlich.

Abbildung 3.6 zeigt das Entsprechende: Für 50 bzw. 1000 gezogene Zufalls-

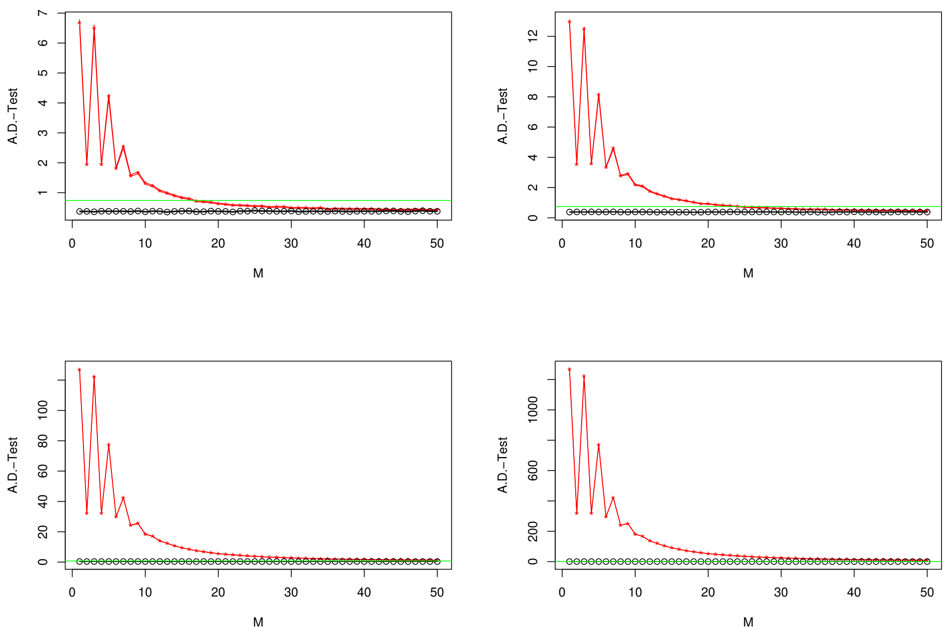

Abb. 3.5: Approximationsgüte der Standardnormalverteilung: wiederholter Anderson-Darling Test bei $n = 50, 100, 1000$ oder 10000 gezogenen Zufallszahlen ohne angepasste Intervalle

zahlen sieht man die Ergebnisse des (wiederum wiederholten) Anderson-Darling Testes für angepasste bzw. nichtangepasste Intervalle. Die horizontale Linie stellt das (aus den vorhergehenden Untersuchungen empirisch ermittelte) Durchschnittsniveau des Tests für Zufallszahlen aus der Normalverteilung dar. Zu beachten sind außerdem die unterschiedlichen Niveaus der X-Achsen.

Logischerweise ist diese Abschätzung wieder direkt abhängig von der Anzahl der gezogenen Zufallszahlen. Eine kleine Stichprobe von 50 Zufallszahlen bei angepassten Intervallen lässt sich mit Hilfe des Anderson-Darling Testes spätestens ab einer Anzahl von $M = 100$ Intervallen nicht mehr von einer Normalverteilung unterscheiden. Bei einer Stichprobe der Größe 1000 benötigt man für ein solches Resultat schon die doppelte Anzahl. Sind die Intervalle alle von gleicher Größe, dauert es bis zu einem befriedigenden Ergebnis erwartungsgemäß deutlich länger.

Das große Problem bei der Approximierung einer stetigen Verteilung durch eine diskrete bleibt bei allen verschiedenen Stichproben natürlich bestehen: Es kommt beim Ziehen mit Zurücklegen naturgemäß immer wieder zu Bin-

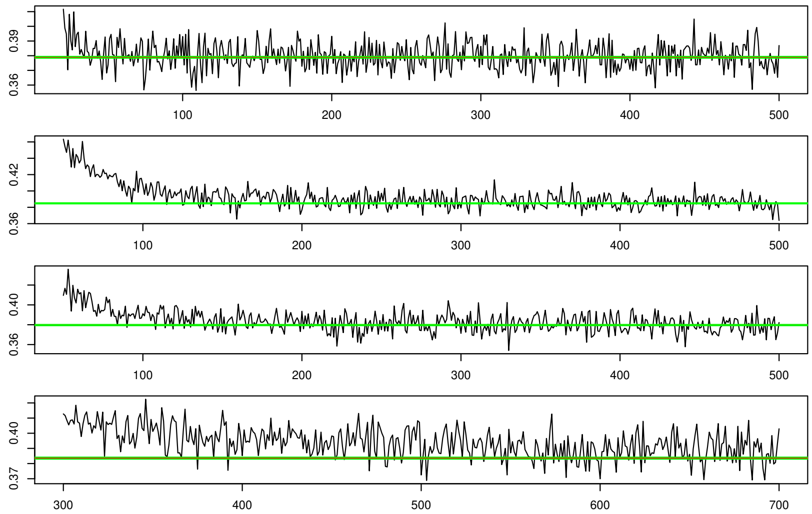

Abb. 3.6: Wiederholter Anderson-Darling Test bei $n = 50$ oder 1000 gezogenen Zufallszahlen mit und ohne angepasste Intervalle (1. angepasst, $n = 50$; 2. angepasst, $n = 1000$; 3. nicht angepasst, $n = 50$; 4. nicht angepasst, $n = 1000$)

dungen in den Teststatistiken, sprich der gleiche Wert erscheint öfter als einmal in der Stichprobe, was bei einer stetigen Verteilung nur mit einer Wahrscheinlichkeit von 0 möglich ist. Der Kolmogorov-Smirnov Test in `R` liefert auch dementsprechende Warnungen. Inwiefern man das und damit auch Änderungen in den Eigenschaften des jeweiligen Tests in Kauf nehmen will, hängt von der Zielsetzung der Untersuchung ab.

3.4 χ^2-Anpassungstest

Um diesem Problem der Bindungen zu entgehen, verwende ich an dieser Stelle zusätzlich den χ^2-Anpassungstest, der auf gruppierte Daten angewendet wird: Es wird dabei der Träger in k Intervalle aufgeteilt und anschließend die Anzahl der im jeweiligen Intervall „gelandeten“ Werte der Stichprobe bestimmt. Die Prüfgröße T_n des Tests ist der χ^2-Abstand zwischen der Häufigkeit $Z = (Z_1, \ldots, Z_k)$ der tatsächlich im jeweiligen Intervall liegenden Werte und der theoretisch erwarteten Häufigkeit $\pi = (\pi_1, \ldots, \pi_k)$:

$$T_n = \sum_{j=1}^{k} \frac{(Z_j - n\pi_j)^2}{n\pi_j}$$

Die Anzahl k der Intervalle ist dabei frei wählbar, beeinflusst jedoch stark das Testergebnis. Anerkannt sind allerdings folgende zwei Daumenregeln: Erstens sollte die Ungleichung $n\pi_j \geq 5$ für alle j erfüllt sein und zweitens sollte so gut wie möglich die Gleichung $\pi_j = 1/k$ für alle j gelten (vgl. [14], S. 215). Der in `R` implementierte χ^2-Test gibt entsprechend Warnungen aus, falls $n\pi_j \geq 5$ nicht für alle j erfüllt ist.

Die hier angewandten χ^2-Tests versuchen dem gerecht zu werden, indem die Anzahl der Gruppen mit der Größe der Stichprobe steigt und außerdem die Intervalle für die Gruppierung mit Hilfe der Verteilungsfunktion der Normalverteilung ermittelt werden: jede Gruppe bekommt dadurch die gleiche Wahrscheinlichkeit zugeordnet, nämlich $1/k$.

Die Ergebnisse des χ^2-Tests sind in Schaubild 3.7 dargestellt: Beispielhaft für kleine bzw. große Stichproben wurden $n = 50$ bzw. $n = 1000$ Zufallszahlen gezogen und die Anzahl der Gruppen k mit 5 bzw. 20 festgesetzt. Wie in den vorhergehenden Tests wurde zur Validierung für jedes M 500 Mal eine Stichprobe gezogen, zum Vergleich gibt es jeweils die Ergebnisse des Tests für aus der „tatsächlichen" Normalverteilung stammende Werte. Die horizontal eingezeichneten Linien stellen wiederum die Grenzen des 10%, 5% und 1%-Ablehnbereiches dar. Zu beachten bei der Interpretation sind die unterschiedlichen Niveaus der X-Achsen.

Auch in diesem völlig anders konzipierten Test werden die vorherigen Ergebnisse, wenn auch mit gewissen Einschränkungen, bestätigt: Ab einer bestimmten Anzahl M an Intervallen werden alle Ablehnschranken unterschritten, der Test kann die Stichprobe nicht mehr als nicht von einer Normalverteilung stammend identifizieren. Auch kann die Anpassung mit wachsendem M so gut gelingen, dass die Ergebnisse gar nicht mehr von den Referenzwerten der Normalverteilung zu unterscheiden sind. Allerdings bestehen große Unterschiede zwischen den Graphen: das liegt vor allem an den Eigenheiten des Tests, der, wie die Technik des Diskretisierens selbst, den Träger in Intervalle aufteilt. Werden bei der Diskretisierungstechnik nun die Intervalllängen angepasst, so kann der Wert des χ^2-Abstandes nicht groß sein, wenn M ein Vielfaches von k ist: bei dem hier verwendeten χ^2-Test und bei der Diskretisierung sind dann die Intervalle sowie die darauf liegenden Wahrscheinlichkeiten die gleichen. Speziell an der Graphik rechts oben kann man das gut

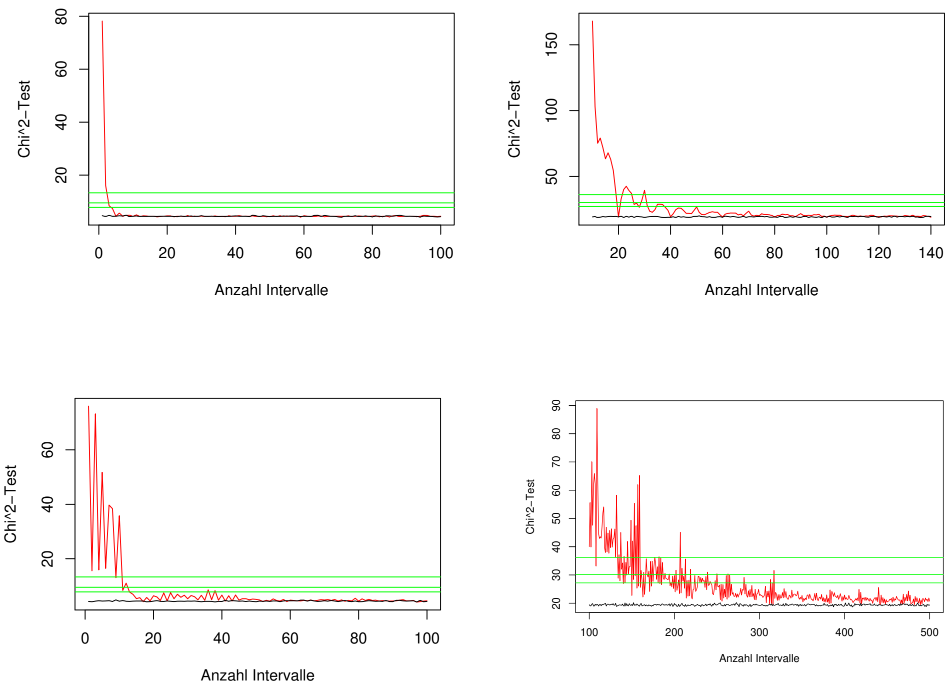

Abb. 3.7: Wiederholter χ^2-Test bei $n = 50$ oder 1000 gezogenen Zufallszahlen mit und ohne angepasste Intervalle (l.o. angepasst, $n = 50$; r.o. angepasst, $n = 1000$; l.u. nicht angepasst, $n = 50$; r.u. nicht angepasst, $n = 1000$)

erkennen, wenn für $M = 20 = k$ die Teststatistik plötzlich sehr klein wird, um anschließend wieder anzusteigen. In den unteren beiden Bildern, also bei den Verfahren ohne angepasste Intervalle, dauert es, wie schon bei den vorangegangenen Tests, wesentlich länger, bis gute Ergebnisse erreicht werden. Bei genügend großen M werden allerdings auch hier alle Ablehnschranken unterschritten.

3.5 Eine naive Diskretisierungsmethode

Die Ergebnisse dieses Kapitels zeigen, dass die diskrete Approximierung einer stetigen Verteilung bei ausreichender Anzahl an Intervallen durchaus gelingen kann. Die Güte des Ergebnisses ist abhängig von der Auswahl der Intervalle.

Ist, wie in dem hier gewählten Beispiel der Normalverteilung, die Verteilungsfunktion der stetigen Variable zumindest numerisch zugänglich, so sollte man die Intervalle an diese anpassen, da hier schneller befriedigende Ergebnisse erreicht werden. Aber auch bei einer lediglich äquidistanten Aufteilung des Trägers kann ab einer bestimmten Anzahl an Intervallen die stetige Variable gut approximiert werden.

Allerdings kann man an dieser Stelle einwerfen, dass es durchaus einfachere Wege gibt, dieses Ziel zu erreichen: Man könnte als neue Zufallsvariable lediglich die M Mitten der Intervalle verwenden, die den Träger der stetigen Verteilung aufteilen. Als Wahrscheinlichkeiten ergäben sich einfach die Differenzen der Verteilungswerte benachbarter Variablen, man könnte sich das numerisch aufwendige Prozedere der Gaußschen Quadratur sparen. Abbildung 3.8 zeigt die Ergebnisse eines Anderson-Darling Testes auf Standardnormalverteilung für ein solches Verfahren: Es wurde aus M gleichgroßen Intervallen jeweils die Klassenmitte als Wert der diskreten Variable und das Integral über die Dichte des jeweiligen Intervalls als Wahrscheinlichkeit genommen und dabei M jeweils in Zehnerschritten erhöht. Aus der neuen Variable wurde anschließend 500 Mal eine Stichprobe für $n = 50$ bzw. $n = 1000$ gezogen und dafür dann der Anderson-Darling Test berechnet. Beim letzten Intervall ist die Intervallmitte und somit der Wert der Zufallsvariable heuristisch auf 4, beim ersten auf -4 festgesetzt.

Im Graphen dargestellt sind in der oberen Reihe die Ergebnisse für $n = 50$, entsprechend in der unteren Reihe die für $n = 1000$. Die rechten Schaubilder starten dabei erst bei $M = 90$, um deutlicher zu zeigen, wann sich die Werte um das aus den vorhergehenden Tests empirisch ermittelte Niveau der Werte der „wahren“ Normalverteilung (horizontale Linie) einpendeln. Die Graphiken auf der linken Seite beginnen bereits bei einem M von 10, zeigen also auch, wann die Testwerte unter die Grenze des 5%-Ablehnbereiches (obere horizontale Linie) fallen.

Auch hier kann man davon sprechen, dass ab einem bestimmten M (für $n = 50$: ≈ 60, für $n = 1000$: ≈ 700) die Ergebnisse nicht mehr vom Original zu unterscheiden sind. Die Frage stellt sich, wo denn dann der Vorteil der Diskretisierungstechnik nach Luceño liegt, die ja bedeutend umständlicher zu berechnen ist?

Die Antwort liegt in den Momenten: Zumindest die ersten $2 \cdot N - 1$ Momente sind bei der Diskretisierung mit denen der stetigen Verteilung approximativ identisch. Bei dem beschriebenen heuristischen Verfahren dagegen lassen sich die Momente auch bei großem M nur schlecht approximieren, eine mathematische Aussage über Identität gibt es im Allgemeinen nicht. Lediglich für

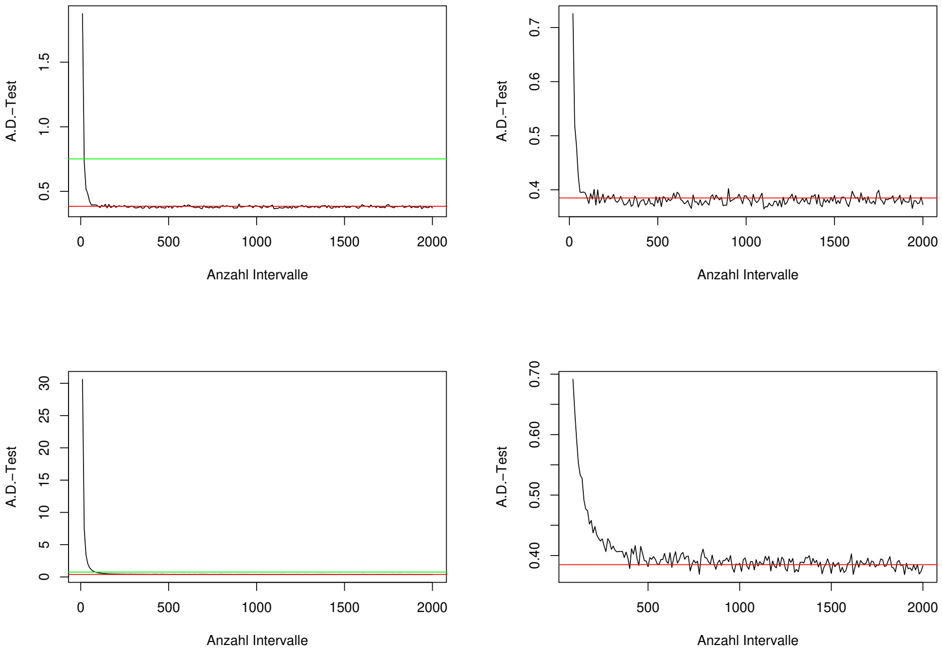

Abb. 3.8: Wiederholter Anderson-Darling Test für $n = 50$ (oben) bzw. $n = 1000$ (unten) gezogene Zufallszahlen ohne Diskretisierungstechnik

den (hier behandelten) Spezialfall der Normalverteilung kann man mit Hilfe der sogenannten Sheppardschen Korrektur die Momente approximieren. Für die ersten vier Momente gilt beispielsweise:

$$\begin{aligned}
\hat{\mu}_1 &= m_1 \\
\hat{\mu}_2 &= m_2 - \frac{1}{12}c^2 \\
\hat{\mu}_3 &= m_3 \\
\hat{\mu}_4 &= m_4 - \frac{1}{2}m_2 + \frac{7}{240}c^2
\end{aligned}$$

mit den Momenten m_i der diskreten Zufallsvariable und den Intervallbreiten c. Dies gilt allerdings nur, wie gesagt, im Spezialfall der Normalverteilung. Zu Näherem, auch zur Güte dieser Korrektur siehe unter anderem [8].

Sucht man im Allgemeinen eine diskrete Zufallsvariable, die eine stetige in jeder Hinsicht, also auch im Hinblick auf höhere Momente, gut approximiert, so stellt die hier behandelte eine sehr effektive Methode dar.

4. Numerische Integration in der Statistik

Das ursprüngliche Anwendungsgebiet der Gaußschen Quadratur liegt in der numerischen Bestimmung von Integralen, für die sie sehr exakte Ergebnisse liefert.[1] Zwar gibt es bereits implementierte Verfahren, die Integrale numerisch berechnen können und teilweise mit Gauß-Stützstellen arbeiten, wie beispielsweise die Funktionen des Quadpack-Paketes, die in Kapitel 2.2.2 erwähnt wurden, für spezielle Fälle kann jedoch auch die Verwendung des Gauß-Algorithmus „von Hand“ nützlich sein.

4.1 Anwendung in der Bayes-Inferenz

Die bayesianische Inferenz unterscheidet sich grundlegend von der „klassischen“ Likelihood-Inferenz: Die zu schätzenden, unbekannten Parameter θ werden nicht als feste, nichtstochastische Größen gesehen, sondern als ebenfalls einer Verteilung unterworfene Zufallsgrößen. Ausgehend von einer Priori-Verteilung $p(\theta)$ für den Parameter θ und einer Verteilungsannahme über die Daten $p(x|\theta)$ in Abhängigkeit von θ, die die Likelihood darstellt, berechnet man die Posteriori-Verteilung

$$p(\theta|x) = \frac{p(x|\theta)\ p(\theta)}{\int p(x|\theta)\ p(\theta) d\theta}, \tag{4.1}$$

um aus ihr Schlüsse zu ziehen. Der Erwartungswert der posteriori-Verteilung kann als Punktschätzer für θ verwendet werden. Ein sogenanntes Kredibilitätsintervall, das einem Bereich im Parameterraum entspricht, welcher eine vorgegebene Wahrscheinlichkeitsmasse enthält, ist das Äquivalent zum Konfidenzintervall der Likelihood-Inferenz.

[1] Als einführende Literatur zum Thema „numerische Integration“ mit dem Gauß Algorithmus empfehlen sich die *Numerical Recipes* ([12] bzw. [23]).

Ein zentrales Problem dieses Verfahrens ist dabei die Berechnung der posteriori-Verteilung in (4.1) bzw. ihrer Momente: beide Male sind hochdimensionale Integrale zu lösen, was analytisch im Allgemeinen nicht möglich ist. Eine populäre Art, dieses Problem zu umgehen, sind die Markov Chain Monte Carlo (MCMC) Methoden, bei denen die posteriori-Verteilung nicht berechnet wird, sondern stattdessen Zufallsvariablen aus ihr gezogen werden, mit deren Hilfe man dann wieder empirische Schätzungen über die Parameter der gesuchten Verteilung bestimmen kann.

Grundsätzlich kann das Problem nun aber auch numerisch, z.B. über die Gaußsche Quadratur bzw. die hier behandelte Diskretisierungsmethode, angegangen werden: Die Dichtefunktion $p(\theta)$ wird dabei durch die Gewichte einer Gaußschen Quadratur und der in $p(x|\theta)$ vorkommende Parameter durch die entsprechende diskretisierte Variable ersetzt. Anstatt des Integrals steht nun eine einfache Summe.

Anhand eines auch bei Luceño verwendeten Beispiels soll die grundsätzliche Vorgehensweise verdeutlicht werden: Gesucht sei der Erwartungswert

$$\mathbb{E}(\mu|\mathbf{x}) \quad = \quad \frac{\int_0^1 \mu \; p(\mathbf{x}|\mu) \; p(\mu) d\mu}{\int_0^1 p(\mathbf{x}|\mu) \; p(\mu) d\mu}, \tag{4.2}$$

wobei die Stichprobe $\mathbf{x}|\mu$ binomial- und der Parameter μ Beta(α, γ)-verteilt sei. Entsprechend sind die $p(\cdot)$ die Wahrscheinlichkeits- bzw. Dichtefunktion der jeweiligen Variable. Ersetzt man bei der Berechnung mittels Gaußscher Quadratur die stetige Dichtefunktion $p(\mu)$ durch die diskrete Wahrscheinlichkeitsfunktion

$$prob(\mu_j) = \frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha, \beta)},$$

sowie $\mu_j = (1 + x_j^{(GJ)})/2$, so hat man statt der Integrale in (4.2) nun Summen zu berechnen:

$$\mathbb{E}(\mu|\mathbf{x}) \quad = \quad \frac{\sum_{j=1}^N \frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha,\beta)} \cdot \frac{1+x_j^{(GJ)}}{2} \cdot p(\mathbf{x}|\mu_j)}{\sum_{j=1}^N \frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha,\beta)} \cdot p(\mathbf{x}|\mu_j)} \tag{4.3}$$

mit den Nullstellen $x_j^{(GJ)}$ und Gewichten $w_j^{(GJ)}$ der Gauß-Jacobi Polynome. Speziell für den Fall von Bernoulli-Experimenten ergibt sich als Erwartungswert für μ:

$$\mathbb{E}(\mu|\mathbf{x}) \quad = \quad \frac{\sum_{j=1}^N \frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha,\beta)} \left(\frac{1+x_j}{2}\right)^{y+1} \left(\frac{-x_j}{2}\right)^{n-y}}{\sum_{j=1}^N \frac{2^{1-\alpha-\beta} w_j^{(GJ)}}{beta(\alpha,\beta)} \left(\frac{1+x_j}{2}\right)^{y} \left(\frac{-x_j}{2}\right)^{n-y}}, \tag{4.4}$$

wobei n dabei die Anzahl an Bernoulli-Experimenten und y die Anzahl der Treffer ist.

So einfach das klingt, der große Nachteil aller numerischen Integrationsmethoden bleibt bestehen: bei etwas größeren Stichproben reicht die Rechnergenauigkeit nicht mehr aus, um die Summanden exakt zu berechnen. Werden die Werte von $n - y$ und y, die in die Potenzen eingehen, zu groß, so kann der Rechner die Ergebnisse nicht mehr von 0 unterscheiden, da für diesen Werte ab 10^{-324} mit 0 gleichgesetzt werden. Somit ist der gesamte Nenner 0 und der Erwartungswerte nicht zu berechnen, `R` gibt den fehlenden Wert `NaN` aus.

Es stellen sich also folgende Fragen:

1. Ab welchem Grad N des Polynoms sind die Ergebnisse genau?
2. Bis zu welcher Stichprobengröße liefert Gauß vernünftige Werte, speziell, ab wann werden `NaN`-Werte ausgegeben?
3. Wo hat Gauß Vorteile gegenüber anderen numerischen Methoden (hier speziell `integrate`)?

Die Funktion `bayes` (siehe Anhang) berechnet für gegebenen Grad N des Polynoms, gegebene Parameter der Priori-Verteilung sowie „Anzahl Versuche“ n und „Anzahl Treffer“ y im Bernoulliexperiment den gesuchten Erwartungswert (4.2) sowohl numerisch mit Hilfe von Gauß bzw. `integrate` als auch analytisch. Der Vorteil dieses einfachen Beispiels liegt ja darin, dass die beiden betrachteten Verteilungen konjugiert zueinander sind, die Posteriori-Verteilung also aus der gleichen Familie wie die Priori stammt und somit der Posteriori-Erwartungswert analytisch bestimmt werden kann.

Aufgerufen mit unterschiedlichem N ceteris paribus dient `bayes` somit zur Beantwortung von Frage 1. In Abbildung 4.1 sieht man für $n = 1000$ und $y = 200$ bei verschiedenen N und den Priori-Parametern $a = 1$ und $b = 1$ den Abstand des mit Gauß berechneten Erwartungswertes zum analytisch gefundenen Wert. Die Posteriori-Verteilung ist hier eine Beta-Verteilung mit den Parametern $\alpha' = \alpha + y$ und $\gamma' = \gamma + n - y$. Der Erwartungswert liegt also für $n = 1000$ und $y = 200$ bei $\frac{\alpha'}{\alpha'+\gamma'} \frac{201}{1001} = 0.2007992$.

Es ist in der Graphik gut zu erkennen, dass der Unterschied zum analytisch bestimmten, also hier gesuchten Wert ab $N = 40$ sehr klein wird. Der berechnete Unterschied zwischen beiden Werten liegt für $N = 41$ bei etwa 10^{-4} und am Ende für $N = 100$ bei 10^{-10}, kann also vernachlässigt werden. Zum Vergleich ist in der gleichen Graphik noch der entsprechende Abstand des mit `integrate` berechneten Wertes vom analytischen eingetragen: Er liegt bei etwa 10^{-3}, reicht also nicht an die Genauigkeit der Gaußschen Werte mit hohem N heran.

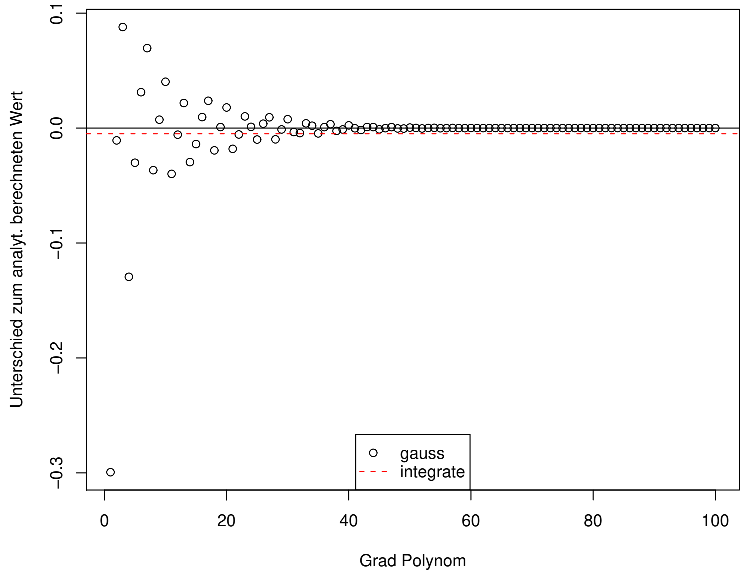

Abb. 4.1: Abstand der Gauß-Werte zum analytisch berechneten Wert für verschiedene N ($n = 1000,\ y = 200$)

Auch für andere Werte von n und y wiederholen sich diese Ergebnisse, allerdings mit unterschiedlichen „burn-in"-Phasen für Gauß. Ab einem N von 60 sind die Berechnungen aber in aller Regel valide in dem Sinn, dass sie sehr nahe an den gesuchten Werten liegen.

Wo liegen jedoch die Grenzen der Einsetzbarkeit von Gauß? Ab wann sind die Potenzen in (4.4) so groß, dass eine numerische Berechnung mithilfe der Polynome nicht mehr möglich ist? Im Prinzip hängt dabei alles an dem Ausdruck $(\frac{1+x_j}{2})^y\,(\frac{-x_j}{2})^{n-y}$ aus (4.4): Je größer n und gleichzeitig y werden, desto kleiner werden die Faktoren, bis sie unter die Schwelle von 10^{-324} kommen und vom Rechner auf 0 gerundet werden. Ist dieser Ausdruck für alle j gleich 0, so liefert die Berechnung von (4.4) einen `NaN`-Wert als Ergebnis.

In Abbildung 4.2 sieht man die Grenzen von Gauß: Für eine gegebene Anzahl an Versuchen (1000 bis 15000) und Treffern (0 bis 1000) zeigt sie, wo Gauß den Erwartungswert mit $N = 50$ und $a = b = 1$ erfolgreich berechnen konnte. In den schwarz eingefärbten Bereichen war dies möglich, im übrigen Gebiet ergaben sich `NaN`-Werte.

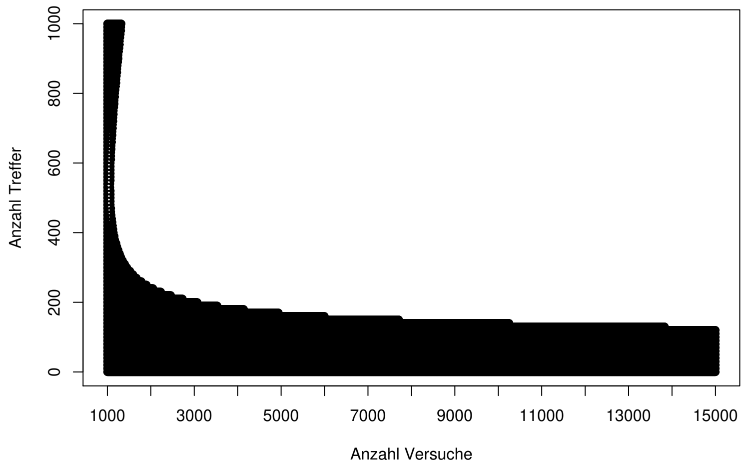

Abb. 4.2: Wirksamkeit von Gauß: bis zu welchen Werten liefert er ein vernünftiges Ergebnis?

Bereits für ein n, das etwas größer als 1000 ist, wird der Erwartungswert nicht mehr berechnet, wenn y aus einem „mittleren" Bereich stammt. Je näher der Wert von y bei dem von n bzw. 0 liegt, desto eher führt die Berechnung zu einem vernünftigen Ergebnis. Die kleinen Sprungstellen in der Graphik ergaben sich dadurch, dass die Werte für n und y in Zehnerschritten „abgegangen" wurden.

Für die `integrate`-Funktion ergibt sich in etwa das gleiche Bild, wie im Weiteren jedoch noch zu sehen, gibt es dennoch kleinere Unterschiede.

Dadurch, dass bei Gauß die Anzahl der Stützstellen nicht von vornherein fest ist, sondern bei der Berechnung von Hand angegeben werden kann, ergibt sich ein etwas größerer Spielraum nach oben: Für große Stichprobenumfänge n und sehr großes oder sehr kleines μ kann man mit der Gaußschen Quadratur eventuell da noch vernünftige Ergebnisse erzielen, wo `integrate` bereits 0 bzw. `NaN` zurückgibt. Die Ursache dafür liegt vor allem, wie erwähnt, in der Anzahl der Stützstellen: Diese liegen (logischerweise) hier immer im Intervall $[0, 1]$. Je höher nun N ist, desto mehr füllen sie dieses Intervall aus, desto näher befinden sich die äußeren Stützstellen also an der 0 bzw. der 1. Und je näher sie sich an der 1 befinden, desto höher ist der Wert ihrer Potenz in $(\frac{1+x_j}{2})^y\,(\frac{-x_j}{2})^{n-y}$ bzw. $(\frac{1+x_j}{2})^{y+1}\,(\frac{-x_j}{2})^{n-y}$ und somit können sie desto eher vom Rechner noch als von 0 verschieden erkannt werden.

Allerdings gilt dies nur für sehr großes oder sehr kleines μ, wenn also y sehr nah an 0 oder n liegt und somit wenigstens bei einem der beiden Faktoren die Potenz relativ klein ist. Nur dann werden auch für kleinere Stützstellen noch Ergebnisse größer 0 geliefert. Abbildung 4.3 zeigt, wo Gauß bzw. `integrate` noch verwertbare Ergebnisse liefern.

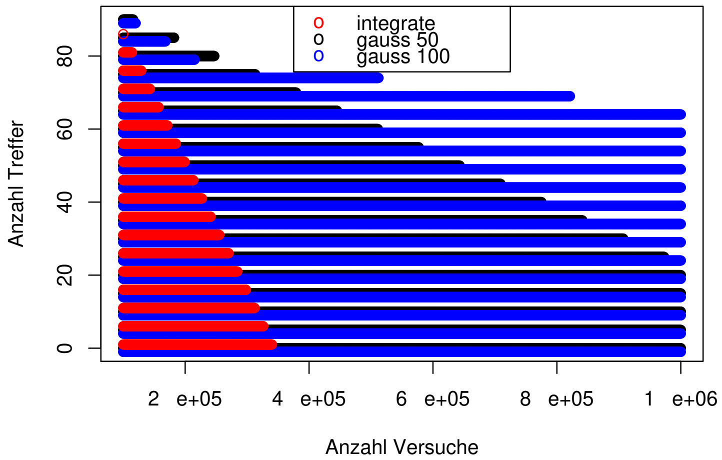

Abb. 4.3: Wirksamkeit von Gauß und `integrate`: bis zu welchen Werten liefern beide ein numerisches Ergebnis?

Die jeweils kürzere der drei horizontalen Linien steht für die Ergebnisse von `integrate`, die beiden anderen für das Verfahren nach Gauß mit $N = 50$ bzw. $N = 100$. Für jeweils unterschiedliche Anzahl an Versuchen $(10^5, \ldots 10^6)$ und Treffern $(0, 5, \ldots, 200)$ zeigt die Graphik, wann die jeweiligen Verfahren Lösungen ungleich 0 liefern. Deutlich zu sehen ist dabei, dass die Methoden mit Gauß länger Lösungen liefern und unter diesen wiederum erwartungsgemäß die mit $N = 100$. Lediglich für etwas höhere y besteht kaum ein Unterschied zwischen den beiden Gauß-Methoden mit unterschiedlicher Anzahl an Stützstellen, die mit kleinerem N scheint stellenweise sogar etwas effektiver.

Ein Blick auf die Genauigkeit der Ergebnisse zeigt, dass die Gaußsche Quadratur mit 50 bzw. 100 Stützstellen auch in diesen Randbereichen gute Ergebnisse liefert: Der relative Fehler zum analytisch berechneten Wert, also $\frac{\hat{\mu}_{gauss} - \hat{\mu}_{analytisch}}{\hat{\mu}_{analytisch}}$ liegt für alle hier berechneten Werte in etwa bei 10^{-4}.

Generell stellt sich im Vergleich mit der `integrate`-Funktion die Frage, welche der beiden Methoden hier die genaueren Ergebnisse liefert. Wie bereits in Abbildung 4.1 gesehen, wird die Gaußsche Methode generell für größeres N genauer. Wie aber verhält es sich über verschiedene n und y hinweg und wie schneidet im Vergleich dazu die `integrate`-Funktion ab?

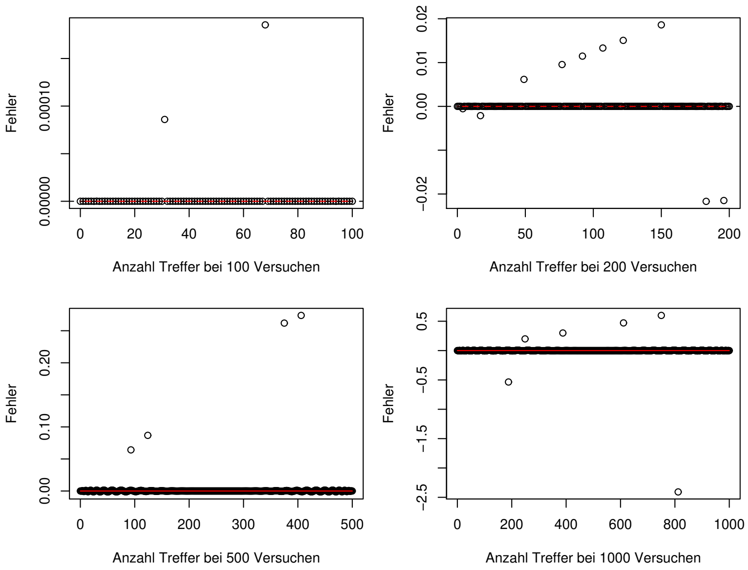

Abb. 4.4: Genauigkeit von Gauß und integrate. ($N = 100, a = b = 1$). Legende: — Gauß, ∘ integrate

Abbildung 4.4 zeigt für die Stichprobenumfänge $n = 100, n = 200, n = 500$ und $n = 1000$ die Fehler (d.h. den jeweiligen absoluten Abstand zum analytisch ermittelten Posteriori-Wert) der mit Gauß (bei $N = 100$) bzw. `integrate` berechneten Erwartungswerte. Die Anzahl der Treffer variiert dabei jeweils über das gesamte Spektrum des Stichprobenumfanges. Die rote, durchgezogene Linie stellt die Fehler des Gauß-Algorithmus, die Kreise die der `integrate`-Funktion dar. Zu beachten sind bei der Bewertung die stark unterschiedlichen Niveaus der Fehlerachsen.

Auffällig sind die Ausreißer in den numerisch mit `integrate` ermittelten Werten, die in ihrer Lage keiner klaren Systematik folgen. Je größer der Stichprobenumfang jedoch ist, desto größer werden sie. Abgesehen von diesen wenigen Ausreißern kann man auf den beiden unteren Graphiken in den äußeren Bereichen kleinere, systematische Schwankungen um die mit dem Gaußschen Verfahren ermittelten Werte, welche sich durch Konstanz auszeichnen, erkennen.

Grundsätzlich ist es bei der numerischen Berechnung von Integralen problematisch, wenn die entsprechende Funktion über den gesamten Träger sehr nahe an 0 liegt, da nur eine endliche Anzahl an Stützstellen verwendet wird. Das Resultat sowie die Abschätzung des Fehlers können dann unter Umständen grob falsch sein. Die hier zu integrierenden Funktionen sind genau von dieser Gestalt, was zu den unkalkulierbaren Fehlern in der numerischen Berechnung führt. Zusätzlich hat Gauß den Vorteil, dass die Gewichtsfunktion des Integrals genau auf die verwendete Gaußsche Quadratur passt, wohingegen bei `integrate` immer die Gauß-Legendre Polynome als Stützstellen mitverwendet werden. Bei Gauß werden im Falle klassischer Wahrscheinlichkeitsfamilien die Stützstellen genau adäquat gewählt, bei `integrate` geht die rechentechnisch günstige Standardisierung der Polynome einher mit Genauigkeitsverlust in speziellen Fällen.

Dies sind wohl die beiden größten Vorteile, die die Verwendung der Gaußschen Quadratur an dieser Stelle gegenüber der herkömmlichen numerischen Berechnung über `integrate` besitzt: Da man im Zweifelsfall immer die Anzahl der Stützstellen bei der Berechnung erhöhen kann und zudem eine genau passende Gewichtsfunktion verwendet, sind die Ergebnisse im Allgemeinen glaubwürdiger (für kleinere N kommt man auch dementsprechend bei dem hier berechneten Erwartungswert mit Gauß zu ungenaueren Ergebnissen). Als Strategie empfiehlt es sich deswegen beispielsweise, mit verschiedenen N der Reihe nach zu experimentieren, bis sich das Ergebnis von $N + 1$ nicht mehr wesentlich (d.h. mit einer vorgegebenen Fehlerschranke) von dem mit N Stützstellen unterscheidet.

Es wurden an dieser Stelle einige Situationen im Bayes-Kontext, speziell beim Bernoulli-Experiment mit Beta-Priori, beschrieben, in denen die Methode nach Gauß Vorteile gegenüber der auf Quadpack basierenden `integrate`-Funktion aufweist. Deutlich wurden dabei aber auch die Grenzen des Möglichen, speziell bei Bild 4.2. Bei großen Stichproben kommt man an den MCMC-Verfahren nicht vorbei, für kleinere Stichproben stellt Gauß jedoch durchaus eine mögliche Alternative dar, falls auf Simulationen verzichtet werden soll, wobei immer im Hinterkopf behalten werden muss, dass lediglich für die klassischen Fälle der Normal-, Gleich, Beta- und Gammaverteilung die Stützstellen effektiv berechnet werden können.

4.2 Integration mit normalverteilten Größen

Generell kann die Gaußsche Quadratur nach Luceño oft da gewinnbringend eingesetzt werden, wo in einem Integral als Gewichtsfunktion eine der oben erwähnten Dichtefunktionen dient. Klassischerweise denkt man in diesem Zusammenhang zunächst an die Momente einer Zufallsvariablen. Aufgrund von (2.4) können dies auch die Momente einer transformierten Variable sein. Die Frage ist, wie bereits im vorhergehenden Unterkapitel, wo und in welcher Form die Berechnung nach Gauß Vorteile gegenüber implementierten Integrationsverfahren wie `integrate` besitzt.

An einem Integral mit einer normalverteilten Größe soll dies beispielhaft demonstriert werden. Es werde im Kontext der Lebensdaueranalyse der Einfluss einer latenten Variable ξ auf die abhängige Variable T (= Überlebenszeit) untersucht. Gemessen werden dazu jedoch nicht die Ausprägungen von ξ direkt, sondern die einer Variable X, eines sogenannten „Surrogats“. Dabei wird folgender Zusammenhang zwischen X und ξ angenommen: die Verteilung von ξ sei abhängig von X.

An einem Beispiel veranschaulicht bedeutet dies: gesucht sei der Einfluss der latenten Variable „Stress“ (ξ) auf die Überlebenszeiten (T) von Mäusen. Gemessen wird als Surrogat die Variable „Durchschnittspuls“(X), von der angenommen wird, dass sie eng mit der Variable „Stress“ zusammenhängt, ohne mit ihr identisch zu sein.

Ersetzt man in der statistischen Analyse einfach ξ durch X, so sind die entsprechenden Schätzer verzerrt (siehe [2]). Eine Korrekturmöglichkeit besteht darin, direkt die Verteilung gegeben X zu ermitteln, um damit eine ML-Schätzung durchzuführen. Gesucht sei nun der Wert der Survivorfunktion $S\ (t|X)$, also das Integral

$$S\ (t|X) \ = \ \int_{\mathbb{R}} S(t|\xi) \cdot f(\xi|X) d\xi. \tag{4.5}$$

Dabei gilt, wobei hier t mit den Parametern $\lambda = \exp(\xi\beta)$ und α weibullverteilt ist:

$$S\ (t|\xi) = exp(-exp\ (\alpha\xi\beta)\ t^{\alpha})$$

und

$$[\xi|X] \sim N(\mu, \sigma^2)$$

mit μ abhängig von X.

Mit Luceños Diskretisierung für die Normalverteilung lässt sich das Integral in (4.5) elegant lösen bzw. numerisch approximieren: Aufgrund von (2.4) gilt

$$S\ (t|X) \cong \sum_{i=1}^{N} \exp(-\exp\ (\alpha x_i \beta)\ t^{\alpha}) \cdot w_i,$$

wobei sich die w_i und x_i mit $w_i = \frac{1}{\sqrt{\pi}} w_i^{(GH)}$ und $x_i = \mu + \sigma x_i^{(GH)} \sqrt{2}$ durch die Gauß-Hermite Polynome ergeben (die x_i sind somit über μ abhängig von X). Man erhält also eine Summe in Abhängigkeit von t. Diese Summe kann man leicht für verschiedene t weiterverwenden, während man mit Standardfunktionen wie `integrate` das Integral (4.5) immer wieder neu berechnen müsste.

Die numerische Berechnung über die Gauß-Hermite Polynome bietet sich außerdem wegen der Gewichtsfunktion des Integrals an: sie ist die genau passende in diesem Fall, während standardisierte Verfahren, wie das bei `integrate` verwendete, nicht unbedingt die optimalen Stützstellen und Gewichte liefern. Da die Funktion $S\ (t|\xi)$ für hohe ξ sehr schnell gegen 0 geht, gibt es für `integrate` bei etwas größerem μ leicht Probleme. Bei $\mu = 7$ und $\sigma = 1$ beispielsweise liefert die Funktion für den Wert des Integrals 0 zurück (bei $\alpha = \beta = t = 1$ für die übrigen Variablen). Die äußeren, linken Stützstellen der standardisierten Prozedur in `integrate` liegen also nicht mehr in einem Bereich, in dem $S\ (t|\xi)$ größer 0 ist.

Anders bei den mit den Gauß-Hermite Polynomen ermittelten Stützstellen: bei $N = 100$ wird hier beispielsweise ein Wert von $1.33 \cdot 10^{-9}$ ermittelt. Von den 100 Stützstellen lagen sehr viele innerhalb des entsprechenden Bereiches, sodass für das Integral ein positiver Wert ermittelt wurde. Auch bei $N \neq 100$ wurde der entsprechende Wert bestätigt.

Über alle möglichen Variablen weg lässt sich für das vorliegende Integral festhalten: Je größer μ bei gleichzeitig kleinem σ ist, desto größere Probleme ergeben sich mit `integrate`. Bei der Methode nach Gauß konnte man auch für diese Werte noch vernünftige Ergebnisse erzielen, die sich für verschiedene Polynomgrade N auch bestätigten. Sie ist der standardisierten Methode hier deutlich überlegen.

Ziel dieses Kapitels war es, beispielhafte Situationen in der Statistik zu vermitteln, in denen bei der numerischen Berechnung von Integralen die „von Hand“ Methode mit Stützstellen der Gauß Quadratur besser geeignet sein kann als standardmäßige Verfahren. Die großen Vorteile dieses Verfahrens können dabei in der genau auf ein Gauß Polynom passenden Gewichtsfunktion, der individuellen Bestimmung des Polynomgrades N, sowie der Wiederverwendbarkeit der Stützstellen für verschiedene Parameter liegen.

5. Anwendung in der klassischen Entscheidungstheorie

Eine große Bedeutung für die Statistik erhält die Gaußsche Quadratur bei ihrer Anwendung in der Entscheidungstheorie. Die klassische Situation des datenfreien Entscheidungsproblems wird dort durch einen Aktionenraum $\mathbb{A}$, einen Zustandsraum Θ, sowie einer Nutzenfunktion $u(\cdot)$ beschrieben: Der Nutzen $u_i(\theta_j)$ der Aktion $a_i \in \mathbb{A}$ ist abhängig von dem (bei der Entscheidungsfindung unbekannten) Umweltzustand $\theta_j \in \Theta$. In einfachen Worten, der Nutzen (u) der Aktion „ich nehme bei einer Wanderung eine schwere Regenausrüstung mit“ (a_1) ist davon abhängig, ob es sonnig (θ_1) oder regnerisch (θ_2) wird.

Die Auswahl einer Aktion wird von ihrer Nutzenbewertung geleitet. Ein klassisches Bewertungskriterium ist das des Erwartungsnutzens, der im Bayesianischen Kontext zu sehen ist: Nach dem Bayes-Postulat kann jede Situation unter Unsicherheit durch einen Wahrscheinlichkeitsraum beschrieben werden, was für das datenfreie Entscheidungsproblem $(\Theta, \mathcal{A}, u(\cdot))$ bedeutet, dass sich immer eine Priori-Verteilung über die Zustände finden lässt, mit deren Hilfe man den Erwartungsnutzen bestimmen kann.

Nichtwissen über die Verteilung der Zustände wird in diesem Zusammenhang durch sogenannte „nichtinformative Prioris“ ausgedrückt, häufig dadurch, dass als Priori die Gleichverteilung über den Zustandsraum Θ gewählt wird (Laplace Regel), was allerdings nicht unproblematisch ist. Im weiteren Verlauf (Kapitel 6) werde ich auf diese Problematik nochmals zu sprechen kommen und mit der nichtklassischen Entscheidungstheorie einen Lösungsweg aufzeigen.

5.1 Effizienzgewinn bei stetigem Zustands- und großem Aktionenraum

Mithilfe der Priori $\pi(\theta)$ und der Nutzenfunktion $u_i(\theta)$ lässt sich in der Bayes-Entscheidungstheorie bei stetigen Zuständen $\theta \in \Theta$ der erwartete Nutzen einer Aktion über das Integral

$$\mathbb{E}(a_i) = \int u_i(\theta) d\pi(\theta) \tag{5.1}$$

berechnen. Optimale und zu wählende Aktion a_{opt} ist die mit dem höchsten Erwartungsnutzen. Luceños Diskretisierung kommt bei der Berechnung dieses Integrals ins Spiel.

Für einen diskreten Zustandsraum Θ ist die Bestimmung nämlich nicht weiter schwierig: der Erwartungsnutzen berechnet sich nach (2.4) aus der Summe

$$\mathbb{E}(a_i) = \sum_j u_i(\theta_j)\pi(\{\theta_j\}), \tag{5.2}$$

die numerisch kein Problem darstellt. Anders sieht es bei einem stetigen Zustandsraum (in Verbindung mit einer stetigen Nutzenbewertung $u(\cdot)$) aus: Hier muss das Integral in (5.1) für jede Aktion mit numerischen Approximationsmethoden bestimmt werden. Bei einer überschaubaren Anzahl an möglichen Aktionen stellt auch das noch kein großes Problem dar, je größer der Aktionenraum jedoch wird, desto ineffizienter wird diese Methode: bei einer Anzahl von 10^6 möglichen Aktionen, die in einem datengestützten Entscheidungsproblem schnell erreicht werden können, muss das Integral (5.1) 10^6-mal numerisch bestimmt werden, was rechentechnisch ein großer Aufwand ist.

Wie sieht es dagegen bei ähnlich großem Aktionen-, jedoch diskretem Zustandsraum aus? Hier gibt es statt 10^6 einzelnen Integralberechnungen eine einzige Matrixoperation:

$$\begin{pmatrix} \int u_1(\theta)d\pi(\theta) \\ \\ \vdots \\ \\ \int u_j(\theta)d\pi(\theta) \\ \\ \vdots \\ \\ \int u_k(\theta)d\pi(\theta) \end{pmatrix} \Longrightarrow \begin{pmatrix} u_1(\theta_1) & u_1(\theta_2) & \dots & u_1(\theta_N) \\ \\ \vdots & \ddots & & \\ \\ u_j(\theta_i) & & & \vdots \\ \\ \vdots & & & \\ \\ u_k(\theta_1) & & \dots & u_k(\theta_N) \end{pmatrix} \bullet \begin{pmatrix} \pi(\{\theta_1\}) \\ \pi(\{\theta_2\}) \\ \\ \vdots \\ \\ \\ \pi(\{\theta_N\}) \end{pmatrix}$$

stetiger Fall diskreter Fall

In kompakter Schreibweise:

$$\mathbb{E} \;=\; U_{\mathbb{A}}(\Theta) \cdot \pi(\Theta) \qquad \text{mit } \Theta = (\theta_1, \theta_2, \dots, \theta_N) \tag{5.3}$$

Der Rechenaufwand ist im Gegensatz zum Fall des stetigen Zustandsraums wesentlich geringer. Falls der Zustandsraum Θ nun aber stetig ist, so liegt hier der Punkt, an dem die Diskretisierungmethode ansetzt: Ersetzt man die stetige Zufallsvariable θ durch die diskreten Werte y_j zusammen mit den dazugehörigen Wahrscheinlichkeiten w_j, so entsprechen aufgrund von (2.4) die Erwartungswerte der Nutzenfunktionen $u_i(y_j)$ numerisch denen von $u_i(\theta)$.

Im Endeffekt dreht man bei dieser Vorgehensweise nur die Reihenfolge um: denn auch die Integrale (5.1) werden normalerweise numerisch bestimmt, was bedeutet, dass jedesmal günstige Stützstellen gesucht werden, um zu der Summe (5.2) zu gelangen. Durch die vorherige Diskretisierung erhält man feste y_i und w_i, die für alle Erwartungsnutzen verwendet werden können und spart sich dadurch das jeweilige Berechnen der Stützstellen. Diese Methode wird für eine große Anzahl an zu berechnenden Intervallen auch von Piessens et al. im Zusammenhang mit den `QUADPACK`-Prozeduren ([11]) vorgeschlagen.

Bsp. 1 (radioaktiv markierte Moleküle):

Zur Veranschaulichung werde ich im Folgenden die oben beschrieben Vorgehensweise an einem fiktiven Beispiel anwenden.

Der Zustandsraum Θ sei hier in einem chemischen Prozess die Anzahl k radioaktiv markierter Moleküle zum Zeitpunkt t in einer Lösung, die weiterverarbeitet werden muss. Erwartet werden zu diesem Zeitpunkt $k = 10^6$ derartige Moleküle. Als Priori wird deshalb eine Normalverteilung um $\mu = 10^6$ mit $\sigma = 2.5 \cdot 10^5$ angenommen. (Als Zusatzannahme kommt selbstverständlich hinzu, dass die Anzahl der entsprechenden Moleküle zum Zeitpunkt t nicht bekannt ist.) Der Aktionenraum $\mathbb{A}$ besteht nun aus den verschiedenen Arten der Weiterverarbeitung. Der Einfachheit halber sei diese lediglich durch unterschiedlich hohen Zusatz eines chemischen Stoffes charakterisiert. Die Aktion a_j stellt also eine bestimmte Dosierung des Zusatzstoffes bei der Weiterverarbeitung dar. Theoretisch ist der Aktionenraum dadurch stetig, praktisch ist die Menge der Aktionen zwar sehr groß, jedoch diskret und sogar endlich, da die Grenzen der Feinabstimmung bei der Dosierung durch die verwendeten Instrumente bestimmt werden. Für jede einzelne Aktion gibt es eine entsprechende Nutzenfunktion $u_j(\theta)$ und man nehme an, diese seien bekannt.

Exemplarisch betrachte man drei Aktionen a_1, a_2 und a_3 mit den Nutzenfunktionen

$$\begin{aligned} u_1(\theta) &= -10\ \theta^2 + 2.1 \cdot 10^7\theta - 9.8 \cdot 10^{12}, \\ u_2(\theta) &= -5\ \theta^2 + 1.1 \cdot 10^7\theta - 5 \cdot 10^{12} - 3 \cdot 10^{10} \quad \text{und} \\ u_3(\theta) &= -2\ \theta^2 + 3.2 \cdot 10^6\theta + log(|\theta|)^{10} - 7 \cdot 10^{11}, \end{aligned}$$

die in Graphik (5.1) dargestellt sind. Die erste Funktion hat dabei ihr Maximum bei 10^6, die zweite bei $1.1 \cdot 10^6$ und die dritte bei $8 \cdot 10^5$.

Der Erwartungsnutzen jeder Aktion lässt sich nun leicht berechnen: Wie in Kapitel 2.2.1 beschrieben, liefert die Funktion `gauss.quad.prob` für eine gegebene Verteilung die entsprechenden Stützstellen und Gewichte der Gaußschen Quadratur zum Grad N. Bei $N = 10$ berechnet man mit den Gewichten für die Normalverteilung mit $\mu = 10^6$ und $\sigma = 2.5 \cdot 10^5$ aus der Summe (5.2) die Erwartungswerte

$$\begin{aligned} \mathbb{E}(a_1) &= 5.75 \cdot 10^{13} \\ \mathbb{E}(a_2) &= 6.575 \cdot 10^{13} \\ \mathbb{E}(a_3) &= 6.26 \cdot 10^{11}. \end{aligned}$$

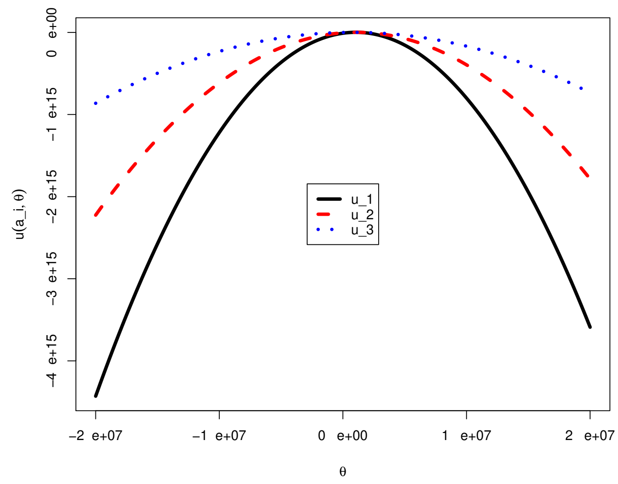

Abb. 5.1: Nutzenfunktionen: $u_1(\theta) = -10\ \theta^2 + 2.1 \cdot 10^7\theta - 9.8 \cdot 10^{12}$, $u_2(\theta) = -5\ \theta^2 + 1.1 \cdot 10^7\theta - 5 \cdot 10^{12} - 3 \cdot 10^{10}$, $u_3(\theta) = -2\ \theta^2 + 3.2 \cdot 10^6\theta + log(|\theta|)^{10} - 7 \cdot 10^{11}$

Optimale Aktion a_{opt} ist die mit dem höchsten Erwartungsnutzen, in diesem Fall also Aktion a_2. Die Funktion `integrate` (mit $\mu \pm 10 \cdot \sigma$ als Integralgrenzen), liefert für jede Aktion praktisch das gleiche Ergebnis:

$$\begin{aligned}\mathbb{E}(a_1) &= 5.75 \cdot 10^{13}\\ \mathbb{E}(a_2) &= 6.575 \cdot 10^{13}\\ \mathbb{E}(a_3) &= 6.26 \cdot 10^{11}.\end{aligned}$$

Bei den beiden relativ „braven“ Nutzenfunktionen von a_1 und a_2 ist der Unterschied der mit beiden Methoden berechneten Werte in der Größenordnung von 10^{-2} bzw. 10^{-3} und damit verschwindend gering. Anders sieht es beim dritten Erwartungswert aus: Hier unterscheiden sich die jeweiligen Werte um etwa 10^5.

Für größere N gleichen sie sich auch hier etwas an: Bei $N = 200$ beispielsweise liegt der Unterschied im Bereich von 10^4, bei $N = 1000$ noch bei 10^3. In der Relation mit den Erwartungswerten ist der Fehler somit nicht sehr groß. (Zum Vergleich der Genauigkeit von `integrate` und Gauß siehe Kapitel 4)

Generell lässt sich sagen, dass mit zunehmender Varianz der Priori die Bestimmung des Erwartungswertes unsicherer wird: Die Anzahl der Gewichte w_i ist ja durch N vorgegeben und bei großer Varianz im Verhältnis zur Breite des betrachteten Intervalls möglicherweise sehr klein. Im laufenden Beispiel sorgen 10 Stützstellen bei einem Erwartungswert von 10^6 und einem σ von $2.5 \cdot 10^5$ für die Berechnung des Integrals. Je komplizierter die entsprechende Nutzenfunktion über Θ ist, desto ungenauer wird das Resultat der numerischen Berechnung des Integrals. Je größer also die Varianz, umso höher sollte das N der Gaußschen Quadratur gewählt werden.

Da sich die Fehler jedoch in relativ überschaubarem Rahmen halten, rückt bei der Gaußschen Methode der Vorteil der Effizienz in den Vordergrund: Für die drei beschriebenen Nutzenfunktionen lassen sich die Berechnungen mit Hilfe von `integrate` noch leicht durchführen, bei einer angenommenen Anzahl von beispielsweise 10^6 möglichen Aktionen sieht das Ganze schnell anders aus: Jedes Integral einzeln zu lösen (und somit jedes Mal erneut nach Stützstellen zu suchen), beansprucht deutlich mehr Rechnerkapazität als mit den festen Gewichten und Stützstellen der Gaußschen Quadratur eine einzige Matrixoperation durchzuführen.

Zur Veranschaulichung dieses Effizienzvorteils dienen die folgenden Simulationen: Für eine gegebene Anzahl n an Aktionen wurden mit Hilfe von `integrate` bzw. `gauss.quad.prob` die entsprechenden n Erwartungswerte berechnet. Als Nutzenfunktion dient dabei für jede Aktion die Funktion $u_1(\theta)$ aus dem Beispiel, was entscheidungstheoretisch natürlich keinen Sinn ergibt. Es interessiert jedoch auch nicht der berechnete Wert, sondern die zum Rechnen benötigte Zeit der beiden Verfahren. Es wird also der Einfachheit halber n Mal der gleiche Erwartungswert berechnet, wobei angenommen wird, dass dies in etwa die gleiche Zeit verbrauchen sollte, wie die Berechnung von n verschiedenen Nutzenfunktionen, was der realistische Fall ist.

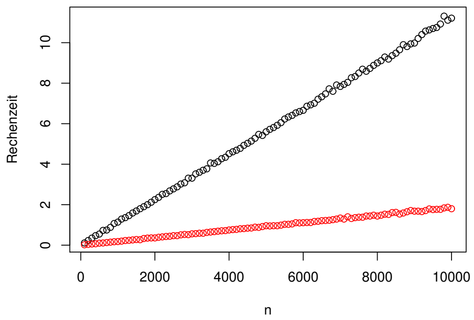

Abb. 5.2: benötigte Rechenzeit für n Erwartungswerte

Graphik 5.2 zeigt die vom Rechner benötigten Zeiten für beide Verfahren bei einem n von 100 bis 10000, also 100 bis 10000 angenommenen Aktionen (auf einem Rechner mit 2.6 Ghz CPU und 240 MB RAM). Für 10000 zu berechnende Erwartungswerte braucht `integrate` bereits 11, das Gaußsche Verfahren nur etwa 1.8 Sekunden. Da beide Kurven sehr deutlich linear ansteigen, lassen sich mit der Methode der kleinsten Quadrate die jeweils passenden Steigungsparameter berechnen, um auf größere n hochzurechnen. Die Gleichungen lauten entsprechend:

$$\begin{aligned} \texttt{integrate}: \quad t &= -0.005994 + 0.001125 \cdot n \\ \texttt{gauss.quad.prob}: \quad t &= 0.0014303 + 0.0001852 \cdot n \end{aligned}$$

Die Effizienz unterscheidet sich hier also in etwa um den Faktor 6. Bei angenommenen 10^6 Nutzenfunktionen nähme die Bestimmung der Erwartungswerte auf diesem Rechner für `integrate` etwa 18 Minuten, für `gauss.quad.prob` dagegen 3 Minuten in Anspruch.

Wie erwartet ergibt sich durch vorherige Diskretisierung ein Gewinn an Effizienz im Sinne von verbrauchten Rechnerkapazitäten, der zwar nicht gewaltig ist, sich aber in wiederholenden Prozessen mit vielen möglichen Aktionen durchaus bemerkbar machen kann.

5.2 Stetiger Aktionenraum

Bisher wurde angenommen, dass die Anzahl der Aktionen zwar sehr groß, die zu jeder Aktion gehörende Nutzenfunktion jedoch explizit bekannt und implementiert sei. Bei 10^6 oder mehr möglichen Aktionen ist diese letzte Annahme allerdings vielleicht problematisch. Viel eher erscheint es sinnig, den Aktionenraum als stetig und das Integral des Erwartungsnutzens als von a abhängige Funkion zu betrachten (wobei a für einen bestimmten Wert der Aktion steht, im obigen Beispiel als Dosierung des Zusatzstoffes):

$$\mathbb{E}(a) = \int u(a,\theta) d\pi(\theta) \tag{5.4}$$

Statt vieler unterschiedlicher $u_j(\theta)$ wird nun also eine einzige, von a und (dem stetigen) $\theta \in \Theta$ abhängige Nutzenfunktion $u(a,\theta)$ angenommen. Das führt zu einem Problem bei der Maximierung der Funktion (5.4). Im Allgemeinen ist es nämlich nicht möglich, das Integral dieser Funktion analytisch zu lösen, um aus der Stammfunktion das Maximum des Erwartungsnutzens zu bestimmen.

Den Erwartungsnutzen könnte man nun als Funktion in Abhängigkeit zweier Parameter betrachten und mit entsprechenden Algorithmen optimieren, was numerisch vielleicht möglich ist, jedenfalls aber kein triviales Problem darstellt.

Eine andere mögliche Herangehensweise an dieses Problem wäre, aus dem stetigen Aktionenraum einen engmaschigen, diskreten Raum zu machen, in diesem jeweils den Erwartungsnutzen jeder Aktion zu bestimmen und darauf zu bauen, dass sich dieser zwischen den Aktionen gutartig (also stetig und monoton) verhält. Die Aktion mit dem höchsten Nutzen ist anschließend zu wählen. Mit dieser Variante diskretisiert man also statt des Zustands- den Aktionenraum.

Bei dieser Methode ist es allerdings schwierig, die Stützstellen geeignet zu wählen, so dass sie den stetigen Aktionenraum gut approximieren. Auch muss die Nutzenfunktion zwischen den gewählten Stützstellen der Aktionen monoton sein, um sicherzugehen, dass die optimale Aktion gewählt wird. Auch dies ist ein nicht-triviales Problem.

Mit Luceños Diskretisierung kann man dieses Problem umgehen, indem man, wie oben, den stetigen Zustandsraum durch einen diskreten, also durch Stützstellen θ_j mit der zugehörigen Wahrscheinlichkeit $\pi(\{\theta_j\})$ ersetzt. Es wird nicht der Aktionenraum, sondern der Zustandsraum diskretisiert. Analog zu (5.2) ergibt sich dann für stetiges a der Erwartungswert als Summe in Abhängigkeit von a:

$$\mathbb{E}(a) = \sum_j u(a, \theta_j)\pi(\{\theta_j\}), \tag{5.5}$$

Von dieser Funktion wiederum kann man leicht das Maximum in Abhängigkeit von a, also die im Sinne des Erwartungsnutzens optimale Aktion a_{opt}, bestimmen.

Bsp. 2 (Fortsetzung: Beispiel 1):

Eine Modifikation des obigen Beispiels dient im Folgenden zur Veranschaulichung: Eine Aktion a sei durch x, die Höhe des Zusatzes bei der Weiterverarbeitung, eindeutig gekennzeichnet. Die Nutzenfunktion jeder Aktion sei nun wie folgt definiert:

$$u(a, \theta) = u(x, \theta) = \begin{cases} \left(\frac{1000000}{x}\right)^4 \cdot I(\theta < x), & \text{für } x \leq 1000000 \\ \left(\frac{1000000}{(2000000 - x)}\right)^4 \cdot I(\theta > x), & \text{für } x > 1000000, \end{cases} \tag{5.6}$$

wobei $I(\theta < x)$ als Indikatorfunktion den Wert 1 für $x > \theta$ und den Wert 0 sonst annimmt ($I(\theta > x)$ entsprechend). Für ausgewählte Aktionen zeigt Graphik 5.3 die Verläufe der entsprechenden Nutzenfunktionen.

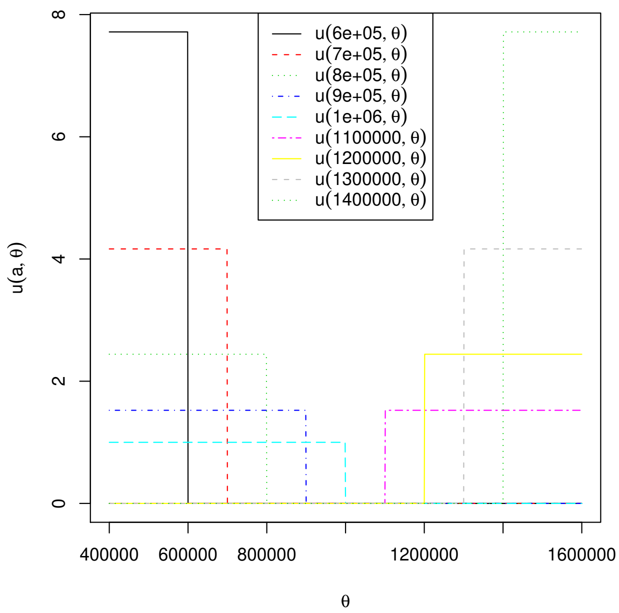

Abb. 5.3: Verlauf der Nutzenfunktionen für $x = 600000$, $x = 700000, \ldots, x = 1400000$

Die Aktionen, die einen positiven Nutzen um den Erwartungswert des Umweltzustandes θ aufweisen, sind gleichzeitig die Aktionen mit den niedrigsten positiven Ausprägungen. Mit wachsendem Abstand zu diesem Erwartungswert wächst auch der möglicherweise auftretende positive Nutzen. Der höchstmögliche Nutzen der Aktion mit $x = 10^6$ liegt beispielsweise bei 1, der der Aktion mit $x = 7 \cdot 10^5$ bei 1.43.

Die Ausprägungen der Nutzenfunktionen sind denen der priori-Verteilung über den Zustandsraum also genau entgegengerichtet: Dort, wo diese ihr Maximum hat, befinden sich die Aktionen mit dem geringstmöglichen Nutzen. Welche Aktion ist somit zu wählen? Mit Hilfe von (5.5) kann der Erwartungswert möglicher Aktionen als Summe in Abhängigkeit von a bzw. x dargestellt und darüber maximiert werden.

Der folgende `R`-Code ist die Implementierung dieser Funktion. `gq` ist dabei ein mithilfe von Luceños Theoremen erstelltes Objekt, `gq$nodes` die neue, diskrete Zufallsvariable über den Zustandsraum und `gq$weights` deren Wahrscheinlichkeitsfunktion.

```
> u = function(x) {
+     res = rep(0, length(gq$nodes))
```

```
+     I = 1:length(gq$nodes)
+     for (i in I) {
+         if if (x <= 1000000 && x > gq$nodes[i]) {
+                 res[i] = (1e+06/x)^4
+             }
+             if (x>1000000 && x < gq$nodes[i]) {
+                 res[i] = (1e+06/(2e+06 - x))^4
+             }
+         }
+     res = sum(res * gq$weights)
+     res
+ }
```

Für einen bestimmten Wert x, also eine Aktion, wird mithilfe der ermittelten Stützstellen für die angenommenen Verteilung der Zustände der entsprechende Erwartungswert berechnet. Das Maximum der Erwartungswerte über alle x lässt sich anschließend beispielsweise mit der `optimize`-Funktion[1] ermitteln.

Es stellt sich jedoch die Frage, welche Diskretisierungsmethode hier anzuwenden ist: Die Unterschiede zwischen den Erwartungswerten der Aktionen mit den Nutzenfunktionen in (5.6) sind stark abhängig von den angenommenen Werten für θ und somit auch von dessen Verteilungsfunktion. Nur Luceños erstes Theorem anzuwenden, könnte an dieser Stelle also nicht ausreichen. Stattdessen bietet sich Theorem 2 an, das eine Zufallsvariable produziert, die eine zur ursprünglichen Variable sehr ähnliche Verteilungsfunktion besitzt.

Wie genau „treffen" die beiden Diskretisierungsmethoden nun das wahre Maximum, können sie also die optimale Aktion identifizieren? Und wie sehr unterscheiden sich die Ergebnisse beider Methoden? Abbildung 5.4 zeigt die mit `optimize` gefundenen Stellen der Maxima des Erwartungswertes zwischen 10^6 und $1.5 \cdot 10^6$ bei einer angenommenen Normalverteilung mit $\mu = 10^6$ und $\sigma = 2.5 \cdot 10^5$ als Priori. Zu erwähnen ist dabei, dass die Funktion `optimize` lediglich ein einziges Maximum liefert, es aber leicht zu sehen ist, dass es aus Symmetriegründen jeweils ein zweites an der Stelle $10^6 - (max_1 - 10^6)$ geben muss.

Die mit Theorem 1 gefundenen Werte (links) schwanken stark und weisen überdies eine deutliche Systematik auf. Es ist auch nicht klar erkennbar, ob sie mit wachsendem N konvergieren, es also bei genügend hohem Grad des

[1] `optimize` verwendet zum Finden des globalen Optimums einer Funktion eine Kombination aus dem Verfahren des Goldenen Schnitts und sukzessiver parabolischer Interpolation.

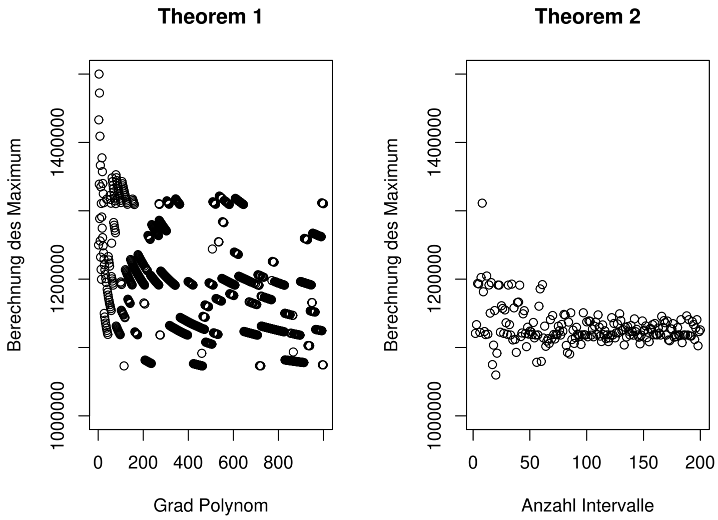

Abb. 5.4: Berechnung des Maximum bei beiden Theoremen (bei Theorem 2: $N = 3$)

Polynoms ein eindeutiges Maximum gibt. Anders bei der sich auf Theorem 2 stützenden Methode, bei dem an die Normalverteilung angepasste Intervalllängen verwendet wurden: Hier schwanken die Maxima wesentlich geringer und scheinen sich mit zunehmender Anzahl an Intervallen auch um einen Wert einzupendeln.

Die Ursachen hierfür liegen darin, dass mit der einfachen Gaußschen Quadratur zwar gute Stützstellen für eine Funktion mit entsprechender Gewichtsfunktion geliefert werden, bei der Berechnung des Erwartungsnutzens im Beispiel jedoch diese Gewichte nicht die einzige dominierende Rolle spielen. Wie erwähnt, sind die Ausprägungen der Nutzenfunktion denen der Verteilungsfunktion genau entgegengerichtet. Die Stützstellen der Quadratur in den Bereichen rund um μ liegen zu weit auseinander, um eine feine Abstufung zwischen den Erwartungsnutzen der einzelnen Aktionen zu erreichen.

Anders, wenn man zuerst den Zustandsraum in (passende) Intervalle aufteilt und anschließend diskretisiert: die mittleren Bereiche der Verteilungsfunktion werden nicht nur durch einige stark gewichtete Stützstellen repräsentiert, sondern erhalten entsprechend mehr und dichtere Stützstellen und somit eine feinere Abstufung, was sich bei der Nutzenfunktion bemerkbar macht. Einen Überblick über die Verläufe der Erwartungsnutzenfunktionen bietet die Graphick 5.5. Für $N = 300$ (Theorem 1) bzw. $N = 3$ und $M = 100$ (Theorem

2) zeigt sie die mit der oben beschriebenen Funktion u berechneten Erwartungsnutzen verschiedener Aktionen.

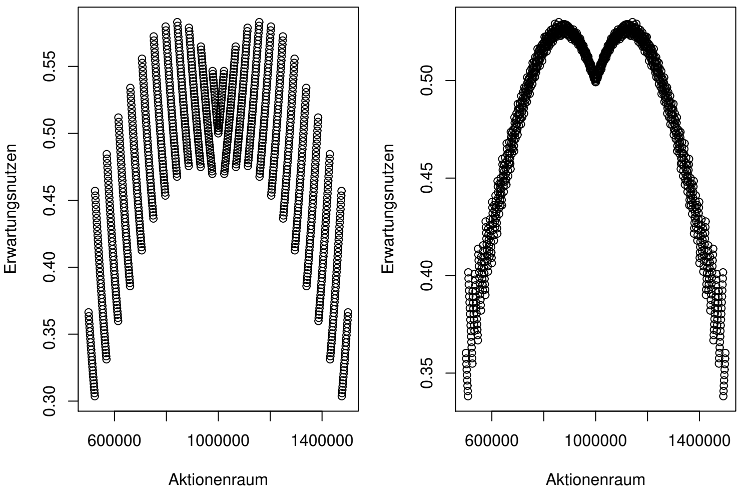

Abb. 5.5: Verlauf der Erwartungsnutzen. links: Theorem 1 ($N = 300$), rechts, Theorem 2 ($N = 3$ und $M = 100$)

Es ist gut zu erkennen, dass die Diskretisierung mit Theorem 1 (links) wesentlich gröbere Ergebnisse liefert. Aufgrund der Eigenheiten der Nutzenfunktion werden an den Stützstellen lokale Maxima angenommen, was beim Finden der optimalen Aktion zu Schwierigkeiten führt. Auch bei Theorem 2 sind diese Strukturen der lokalen Maxima zu erkennen, allerdings ist die Funktion in den wichtigen mittleren Bereichen vielfach feiner, die Probleme beim Finden des Optimums sind somit kleiner.

Die Erwartungswerte sind zudem stark abhängig von der angenommenen Streuung σ. Wird diese zu groß, so überwiegt bei der Berechnung des Erwartungswertes der Einfluss der Nutzenfunktion, die links und rechts von 10^6 monoton ansteigt, gegenüber dem der Priori-Verteilung, die ja genau anders herum aufgebaut ist. Die Funktion steigt also nach außen hin monoton an, was bedeutet, dass eine Aktion umso besser ist, je weniger sie sich auf den Erwartungswert des Zustandes verlässt. Auch bei kleinerem σ steigen die Erwartungsnutzen am Rand des Aktionenraums wieder an, da auch hier die Nutzenfunktion, die nach außen hin sehr groß wird, die hier sehr kleine Verteilungsfunktion dominiert. Allerdings sollte man sich beim Finden der optimalen Aktion hier wohl auf die mittleren Bereiche des Aktionenraumes beschränken.

Generell ist die Funktion zur Bestimmung des Erwartungswertes an den äußeren Stützstellen bei beiden Theoremen dadurch etwas problematisch, da diese im Vergleich zu den inneren relativ weit auseinander liegen. An diesen Stellen hat die Diskretisierung mit Intervallen die gleichen Probleme, die die einfache Diskretisierung bereits in den mittleren Bereichen besitzt.

5.3 Lineare Optimierung

Eine alternative Möglichkeit, den globalen Erwartungsnutzen zu maximieren, bietet die lineare Optimierung unter Nebenbedingungen. Sie verwendet dabei die gemischte Erweiterung des Aktionenraumes und identifiziert so die optimale Aktion.

Ein Vorteil der linearen Optimierung liegt in ihrer Anwendbarkeit in der nichtklassischen Entscheidungstheorie (siehe Kap. 6): beispielsweise kann der maximale Erwartungsnutzen aller Aktionen bzw. die jeweils optimale Aktion in Abhängigkeit vom μ einer Priori betrachtet werden.

Wir betrachten zunächst noch einmal die Problemstellung aus dem Unterkapitel 5.1: Im datenfreien Entscheidungsproblem sei der Zustandsraum Θ stetig, ebenso wie die Nutzenfunktion $u_i(\theta)$ der Aktion a_i. Zudem sei die Anzahl der möglichen Aktionen a_i zwar groß, aber nicht ∞. Wie bereits erläutert, kann bei bekannten Nutzenfunktionen dieses Problem durch Diskretisierung des Zustandsraumes und anschließender Berechnung der Erwartungsnutzen über (5.2) angegangen werden.

Eine andere Möglichkeit zur Lösung dieses Problems bietet die lineare Optimierung: Man betrachte dazu die gemischte Erweiterung der Aktionenmenge $\mathbb{A}$:

$$\lambda = (\lambda(a_1), \ldots, \lambda(a_S)).$$

Die Aktion a_i wird dabei mit der Wahrscheinlichkeit $\lambda(a_i)$ gewählt. Der Erwartungsnutzen dieser gemischten Erweiterung ergibt sich bei stetigem Zustandsraum zu

$$\mathbb{E}(\lambda) = \int \left(\sum_i u_i(\theta)\lambda(a_i) \right) d\pi(\theta) \tag{5.7}$$

Beim Maximieren dieses Erwartungswertes stößt man allerdings auf das Problem, dass das Integral im Allgemeinen nicht analytisch lösbar ist. Hier kommt wieder die Diskretisierungstechnik ins Spiel, mit der man, wie bereits oben, den stetigen Zustandsraum in einen diskreten, θ also in θ_i, überführt. Der Erwartungsnutzen lautet dann wie folgt:

$$\begin{aligned} \mathbb{E}(\lambda) &= \sum_j \left(\sum_i u_i(\theta_j)\lambda(a_i) \right) \pi(\{\theta_j\}) \\ &= \sum_i \left(\sum_j u_i(\theta_j)\pi(\{\theta_j\}) \right) \lambda(a_i) \end{aligned} \tag{5.8}$$

Mithilfe linearer Optimierung unter Nebenbedingen kann (5.8) maximiert werden. Die Nebenbedingungen sind dabei trivialerweise

$$\begin{aligned} \sum_i \lambda(a_i) &= 1 \\ \lambda(a_i) &\geq 0 \quad \forall i. \end{aligned}$$

Alle möglichen Lösungen, sprich alle λ, die diese Nebenbedingungen erfüllen, bilden zusammen ein konvexes Polyeder, ein sogenanntes Simplex.

Bei der linearen Optimierung (z.B. durch den Simplexalgorithmus) unter Nebenbedingungen wird in diesem Polyeder das optimale λ (also die optimale Aktion) gesucht. Dabei gilt, dass sich eine optimale Lösung auf einer Ecke der Menge aller zulässigen Lösungen befindet. Implementierte Algorithmen konzentrieren sich somit auf die Ecken des Polyeders und liefern aus diesen eine optimale Lösung. Für die gewählte Aktion bedeutet dies, dass sie (obwohl wir ja eigentlich die gemischte Erweiterung betrachten) den reinen Aktionen zuzuordnen ist, da die Ecken in diesem Simplex bei $\lambda(a_i) = 1$ und $\lambda_j = 0 \; \forall j \neq i$ liegen. Dies geht auch konform mit der für die bayesianische Entscheidungstheorie gültigen Aussage, dass es zu jeder randomisierten Bayes-Aktion (also optimalen Aktion) eine reine Aktion gibt, die auch Bayes-Aktion ist.

In `R` ist ein Algorithmus zur linearen Optimierung unter Nebenbedingungen beispielsweise in dem Packet `lpSolve` in Form der Funktion `lp` implementiert. Sie ruft den `lp_solve 5.5 solver` auf, einen sogenannten Mixed Integer Programming (MIP) Solver (Näheres dazu siehe [25]). Maximiert bzw. minimiert wird das Skalarprodukt $c'x$ unter den Nebenbedingungen $Ax \geq b$ und $x \geq 0$. In unserem Fall ist x der gesuchte Vektor λ, die Matrix der entsprechenden Nebenbedingungen ist A , sowie $c'x$ der zu maximierende Erwartungsnutzen aus (5.8).

Bsp. 3 (Fortsetzung: Beispiel 1):

Zur Veranschaulichung dient wiederum die Situation aus Beispiel (1) mit den drei Aktionen a_1, a_2, a_3 und deren Nutzenfunktionen

$$\begin{aligned} u_1(\theta) &= -10\ \theta^2 + 2.1 \cdot 10^7\theta - 9.8 \cdot 10^{12}, \\ u_2(\theta) &= -5\ \theta^2 + 1.1 \cdot 10^7\theta - 5 \cdot 10^{12} - 3 \cdot 10^{10} \quad \text{und} \\ u_3(\theta) &= -2\ \theta^2 + 3.2 \cdot 10^6\theta + log(|\theta|)^{10} - 7 \cdot 10^{11}. \end{aligned}$$

Welche Aktion ist nun für eine angenommene Priori-Normalverteilung über dem Zustandsraum mit $\mu = 10^6$ und $\sigma = 2.5 \cdot 10^5$ zu wählen?

Zunächst muss diese Priori diskretisiert und die entsprechenden Ergebnisse in den Vektor c eingetragen werden. Um den Erwartungsnutzen aus (5.8) in der Vektorschreibweise $c'x$ auszudrücken, muss gelten:

$$\begin{aligned} x &= (\lambda(a_1), \lambda(a_2), \lambda(a_3)) \qquad \text{und} \\ c &= \left(\sum_j u_1(\theta_j) \cdot \pi(\{\theta_j\}), \sum_j u_2(\theta_j) \cdot \pi(\{\theta_j\}), \sum_j u_3(\theta_j) \cdot \pi(\{\theta_j\}) \right). \end{aligned}$$

Da die Funktion `lp` die Bedingung $\lambda(a_i) \geq 0 \quad \forall i$ sowieso beachtet, müssen die übrigen Nebenbedingungen nur noch $\sum_i \lambda(a_i) = 1$ gewährleisten. Normalerweise fließen diese Bedingungen in Form der Ungleichung $Ax \geq b$ in den Algorithmus ein. Soll eine Gleichheit erfüllt sein (wie hier), so wird zusätzlich $-Ax \geq -b$ gefordert und in diese Ungleichung aufgenommen. Beim Aufrufen der Funktion `lp` kann dieser Schritt bequemerweise weggelassen und statt dessen die Gleichung $Ax = b$ als Nebenbedingung angegeben werden. Sie sieht also wie folgt aus:

$$\begin{array}{cccc} (1,1,1) & \begin{pmatrix} \lambda(a_1) \\ \lambda(a_2) \\ \lambda(a_3) \end{pmatrix} & = & 1 \\ A & x & = & b \end{array}$$

Der Aufruf

```
lp(direction = "max", c, A, "=",b)
```

liefert ein Objekt der Klasse `lp`, in dem unter Anderem der optimale Vektor x, sowie der entsprechende Wert von $c'x$ aufgeführt ist. Da, wie oben beschrieben, eine Lösung auf einer Ecke der Menge aller zulässigen Lösungen zu finden ist, gilt $\lambda(a_i) = 1$ für ein i und sonst $\lambda(a_j) = 0, \ \forall j \neq i$. Der Wert von $c'x$ mit dem optimalen $x = (\lambda(a_1), \lambda(a_2), \ldots)$ entspricht also sowohl dem optimalen Erwartungsnutzen der Aktionen in der gemischten Erweiterung, als auch speziell dem Erwartungsnutzen der optimalen Aktion a_i für die angenommene priori-Verteilung. Da es bei der Diskretisierung in diesem Fall nicht darum geht, die Verteilungsfunktion möglichst exakt im Diskreten darzustellen, sondern um die numerische Bestimmung eines Integral, ist hier das unkompliziertere Theorem 1 von Luceño (siehe (2.4)) vorzuziehen.

Ein Aufruf von `lp` mit den Nutzenfunktionen dieses Beispiels und der mit $N = 10$, $\mu = 10^6$ und $\sigma = 2.5 \cdot 10^5$ diskretisierten Normalverteilung, führt zu folgender optimalen Lösung:

$$\begin{aligned} x &= (0, 1, 0) \\ c'x &= 6.575 \cdot 10^{11} \end{aligned}$$

Mit der Aktion a_2 maximiert man also den erwarteten Nutzen zu $6.575 \cdot 10^{11}$. Das war natürlich schon aus dem Beispiel 1 bekannt. Hier wurde lediglich ein alternativer Weg der Berechnung aufgezeigt. Die eigentlichen Stärken der Anwendung der linearen Optimierung in der Entscheidungstheorie werden in Situationen unter komplexer Unsicherheit deutlich, wie sie im nächsten Kapitel behandelt werden.

6. Entscheidung unter komplexer Unsicherheit: Intervallwahrscheinlichkeiten

Die im Anhang aufgeführte Funktion `linopt` automatisiert das gerade beschriebene Vorgehen: Für gegebenes N, einer Matrix mit Nutzenfunktionen verschiedener Aktionen, sowie dem angegebenen Typ („normal“, „uniform“, „beta“, „gamma“) der Priori mit den entsprechenden Parametern a und b berechnet sie mithilfe der Diskretisierungstechnik aus Theorem 1 und linearer Optimierung die optimale Aktion a_{opt} und den dazugehörigen Erwartungswert. Der maximale Erwartungswert, bzw. die zu wählende Aktion, kann also sozusagen als Funktion in Abhängigkeit dieser Parameter der Priori betrachtet werden. Abbildung 6.1 zeigt für das vorhergehende Beispiel die optimale Aktion bzw. den maximal zu erwartenden Nutzen in Abhängigkeit von σ bei gleichbleibenden übrigen Variablen. Diskretisiert wurde dabei mit einem Polynom vom Grad $N = 10$.

Diese Tatsache, dass sich der Erwartungsnutzen leicht als Funktion in Abhängigkeit der Priori-Parameter schreiben lässt, kann in der nichtklassischen Entscheidungstheorie verwendet werden.

6.1 Die nichtklassische Entscheidungstheorie

Das Bayessche Postulat besagt, dass jede Situation unter Unsicherheit mit einem Wahrscheinlichkeitsraum beschrieben werden kann. Darauf stützten sich die vorhergehenden Berechnungen: Die vorgegebene Priori-Verteilung beschreibt die Unsicherheit über dem Zustandsraum und mit ihrer Hilfe kann man das Entscheidungskriterium „Erwartungsnutzen“ berechnen. Ellsberg ([4]) stellte jedoch in seinem berühmt gewordenen Experiment fest, dass es in realen Entscheidungssituationen nicht nur die klassische Wahrscheinlichkeit ist, die das (rationale) Handeln der Akteure bestimmt, dass also die

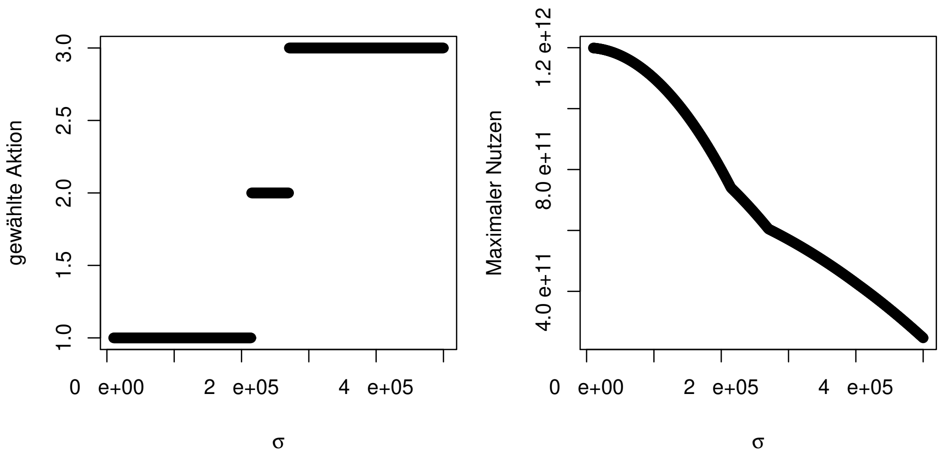

Abb. 6.1: Optimale Aktion im Sinne des Erwartungsnutzens in Abhängigkeit von σ (links), maximal möglicher Nutzen in Abhängigkeit von σ (rechts)

„normale“ Entscheidungstheorie in bestimmten Situationen nicht ausreicht, um die Unsicherheit über Umweltzustände und somit das tatsächliche Entscheidungsverhalten zu beschreiben.

Deshalb geht die nichtklassische Entscheidungstheorie einen Schritt weiter: Für sie ist Unsicherheit ein mehrdimensionales Phänomen, in dem der Begriff der Wahrscheinlichkeit nur eine Dimension darstellt. Eine weitere Dimension ist die Unbestimmtheit (Ambiguity), die ausdrückt, wie gut das Wissen über den Wahrscheinlichkeitsraum der Zustände ist. Denn neben der Unsicherheit **im** Wahrscheinlichkeitsraum gibt es die **über** den Wahrscheinlichkeitsraum. So kann z.B. im Bernoulliexperiment Unsicherheit über den Wert der Wahrscheinlichkeit p herrschen.[1] In die Entscheidung fließt dann neben dem unter p erwarteten Nutzen auch diese Unsicherheit mit ein. Die Unbestimmtheit geht bei absolutem Nichtwissen dabei so weit, dass über p lediglich die triviale Tatsache bekannt ist, dass sich dieses irgendwo zwischen 0 und 1 befindet.

Wie bereits in Kapitel 5 kurz erwähnt, gibt es auch in der Bayes-Entscheidungstheorie Lösungsansätze zu diesem Problem des Nichtwissens über die Parameter, die sogenannten nichtinformativen Prioris. Die einfachste Lösung ist dabei, dass als Priori die Gleichverteilung über den Träger gewählt wird. Der Ausdruck „nichtinformativ“ ist speziell bei der Gleichverteilung jedoch irreführend und auch für andere Lösungen umstritten (siehe

[1] Für weitere Informationen zum Thema „komplexe Unsicherheit“ siehe [26]

dazu [13], Kap. 2.5).

In der erweiterten Entscheidungstheorie kann die Unbestimmtheit dadurch modelliert werden, dass mit unteren (L) und oberen (U) Grenzen für klassische Wahrscheinlichkeiten (nach den Axiomen von Kolmogorov) über den Zustandsraum gearbeitet wird. Dabei gilt in der Axiomatik von K. Weichselberger ([20]) für einen Messraum $(\Omega, \mathcal{A})$ und die Menge $\mathcal{Z}_0 := \{[L, U] : 0 \leq L \leq U \leq 1\}$ abgeschlossener Intervalle in $[0, 1]$:

Eine Funktion $P(\cdot)$ über $\mathcal{A}$ heißt *F-Wahrscheinlichkeit*, wenn gilt

$$P(\cdot) : \mathcal{A} \to \mathcal{Z}_0,\ A \mapsto P(A) = [L(A), U(A)], \tag{6.1}$$

die Menge $\mathcal{M}$ klassischer Wahrscheinlichkeiten $\pi(\cdot)$

$$\mathcal{M} := \{\pi(\cdot) \mid L(A) \leq \pi(A) \leq U(A),\ \forall A \in \mathcal{A}\} \tag{6.2}$$

ist nichtleer, sowie

$$L(A) = \inf_{\pi(\cdot) \in \mathcal{M}} \pi(A) \quad \text{und} \quad U(A) = \sup_{\pi(\cdot) \in \mathcal{M}} \pi(A) \quad \forall A \in \mathcal{A}. \tag{6.3}$$

$\mathcal{M}$ ist dabei konvex und stellt für endliches Ω der Mächtigkeit N einen $(N-1)$ dimensionalen Polyeder, die sogenannte „Struktur“ dar.

Für $|\Omega| = |A, B| = 2$ beispielsweise ist die Struktur eine Strecke im zweidimensionalen Raum, die durch $L(A), L(B)$ und $U(A), U(B)$ sowie die Bedingung $\pi(A) + \pi(B) = 1$ definiert ist. Die Skizze einer solchen Struktur ist für gegebene $L(A), L(B)$ und $U(A), U(B)$ in Abbildung 6.2 zu sehen.

Wenn zusätzlich gilt, dass $L(\cdot)$ 2-monoton ist, also

$$L(A \cup B) + L(A \cap B) \geq L(A) + L(B), \tag{6.4}$$

so heißt $P(\cdot)$ über $\mathcal{A}$ *C-Wahrscheinlichkeit*. Für $L(A) = U(A)$ ergibt sich im Übrigen aus der C-Wahrscheinlichkeit eine klassische Wahrscheinlichkeit.

An einem Beispiel von Troffaes [18] soll die Anwendung von Intervallwahrscheinlichkeiten in der Entscheidungstheorie unter komplexer Unsicherheit verdeutlicht werden:

Bsp. 4 (Kriterien nichtklassischer Entscheidungstheorie):

Sei Θ das unbekannte Ergebnis eines Münzwurfes mit den Ausprägungen Kopf (K) und Zahl (Z). Das Wissen über die Wahrscheinlichkeit p_K sei unscharf, bekannt sei lediglich, dass p_K zwischen 0.28 und 0.7 liegt, somit ist $L(p_K) = 0.28$ und $U(p_K) = 0.7$. Die möglichen Aktionen $a_1, \ldots, a_6$ besitzen folgende Nutzenfunktionen und entsprechenden Erwartungsnutzen:

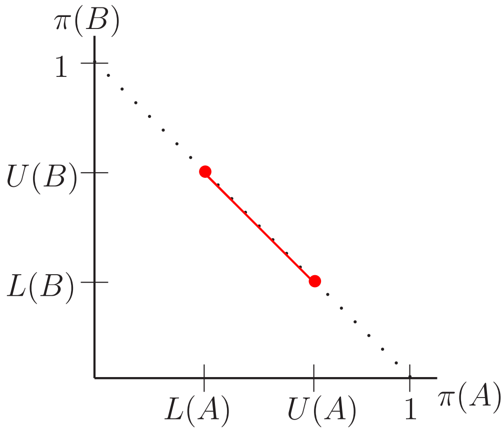

Abb. 6.2: Skizze einer Struktur bei $|\Omega| = 2$

	$K \mathrel{\widehat{=}} \theta_1$	$Z \mathrel{\widehat{=}} \theta_2$	$\mathbb{E}_p(a_i)$
a_1	4	0	$4 \cdot p_K$
a_2	0	4	$4 - 4 \cdot p_K$
a_3	3	2	$2 + 1 \cdot p_K$
a_4	1/2	3	$3 - 2.5 \cdot p_K$
a_5	47/20	47/20	47/20
a_6	41/10	-3/10	$-0.3 + 4.35 \cdot p_K$

Wie bereits in Abbildung 6.1 variieren auch hier die optimale Aktion und der entsprechende Erwartungswert über verschiedene Werte der klassischen Wahrscheinlichkeit p_K. Abbildung 6.3 zeigt die Verläufe der einzelnen Erwartungswerte, sowie, als dickere Linie, den unter allen Aktionen jeweils optimalen Erwartungswert.

Diese optimalen Aktionen entsprechen einem Prinzip der nichtklassischen Nutzentheorie, das unter dem Namen *E-Admissibilität* bekannt ist: Die Menge E-admissibler Aktionen ist die Menge optimaler Aktionen über der Menge aller in Betracht kommenden klassischen Wahrscheinlichkeitsmaße in der Unsicherheitssituation, d.h. die Menge der Aktionen, die unter wenigstens einem möglichen Wahrscheinlichkeitsmaß optimal in dem Sinne sind, dass keine andere Aktion unter diesem Wahrscheinlichkeitsmaß einen höheren Erwartungsnutzen besitzt. E-Admissibilität kann man somit im weiteren Sinn als eine Verallgemeinerung des Kriteriums der Zulässigkeit aus der klassischen Entscheidungstheorie betrachten. In diesem Beispiel ist die Menge der

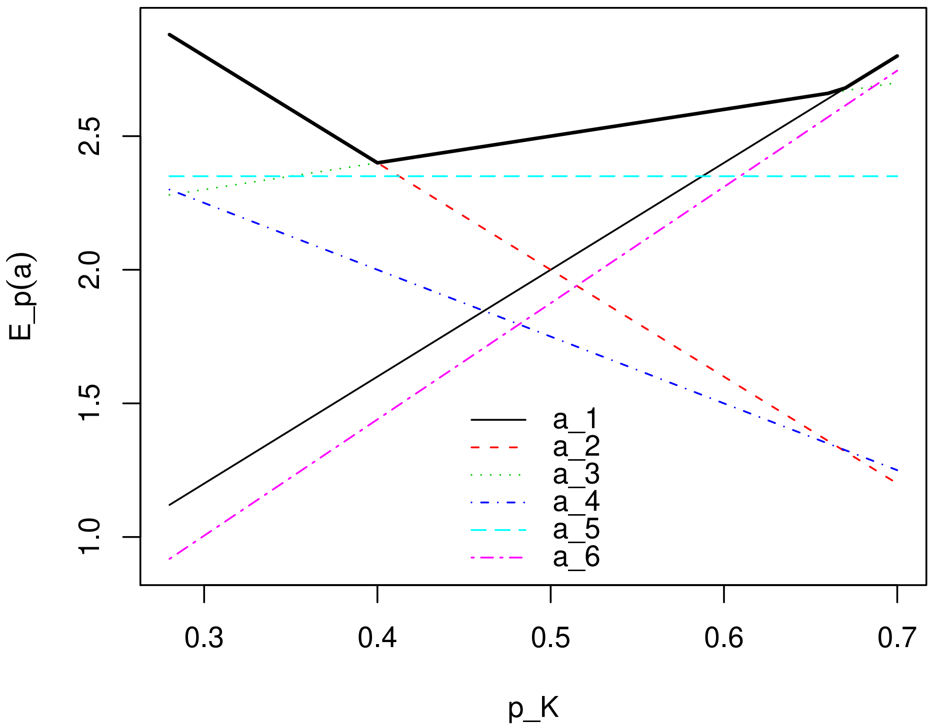

Abb. 6.3: (optimale) Erwartungswerte der Aktionen in Troffaes Beispiel

E-admissiblen Aktionen $\{a_1, a_2, a_3\}$.

Ein weiteres Kriterium der nichtklassischen Nutzentheorie ist das Γ-*Maximin Konzept*[2]: hier wird der ungünstigste Erwartungswert jeder Aktion über der Menge der möglichen Wahrscheinlichkeitsmaße betrachtet. Gewählt wird die Aktion, die diesen minimal zu erwartenden Nutzen maximiert. In Troffaes Beispiel ist die nach diesem Kriterium zu wählende Aktion a_5. Das Γ-Maximin Konzept ist eine sehr konservative Entscheidungsregel, die sich am schlechtest möglichen Fall orientiert und ähnlich dem aus der klassischen Entscheidungstheorie bekannten Maximin-Prinzip funktioniert: im Fall völliger Unbestimmtheit, also bei $L(p_K) = 0$ und $U(p_K) = 1$, entspricht sie sogar exakt dem Maximin-Kriterium.

[2] Zu den Eigenschaften und Vorteilen der beiden (und anderer) Kriterien siehe [18]

6.2 Die nichtklassische Entscheidungstheorie bei stetigem Zustandsraum

Wie sich nun diese Kriterien auf den stetigen Zustandsraum verallgemeinern lassen, hängt davon ab, wie die Unsicherheitssituation genau aussieht, welche Wahrscheinlichkeitsmaße also überhaupt in Betracht kommen: Es kann sein, dass lediglich Unsicherheit über einen (oder mehrere) Parameter einer feststehenden Verteilungsfamilie herrscht: In diesem Fall ist die optimale Aktion, bzw. ihr Erwartungsnutzen eine Funktion in Abhängigkeit dieser Parameter der Priori-Verteilung. Das Problem muss also nicht mit F-Wahrscheinlichkeiten angegangen werden. Zusätzlich kann aber auch Unsicherheit über die Verteilungsfamilie herrschen, wobei bestimmte obere und untere Grenzen für die Werte der Verteilungsfunktion bekannt sein können, was zu den angesprochenen Intervallwahrscheinlichkeiten führt.

6.2.1 Variante 1: Unsicherheit über Priori-Parameter

Betrachten wir zunächst die etwas „harmlosere“ Variante, bei der lediglich Unsicherheit über einzelne Priori-Parameter herrscht, es also klar ist, zu welcher Verteilungsfamilie die Priori gehört. Wie bereits in Bild 6.1 zu sehen war, kann auch im Falle eines stetigen Zustandsraumes mit Hilfe der Diskretisierung der optimale Erwartungsnutzen als Funktion in Abhängigkeit von Parametern der Priori betrachtet werden. Dadurch ergeben sich innerhalb des Raumes über die möglichen Ausprägungen dieser Parameter unterschiedliche Lösungen des Optimierungsproblems. Die Menge der Aktionen, die über der Menge aller in Betracht kommenden Ausprägungen irgendwo optimal sind, ist die Menge E-admissibler Aktionen.

Abbildung 6.1 stellt im Kontext des Beispiels 1 die Situation dar, in der von der Priori die Verteilungsfamilie der Normalverteilung sowie der Parameter $\mu = 10^6$ bekannt sind. Unsicherheit herrscht dagegen über σ: Von diesem ist lediglich bekannt, dass es sich irgendwo zwischen 10^4 und $5 \cdot 10^5$ befindet. Links ist die jeweils optimale Aktion für ein bestimmtes σ zu sehen, rechts der entsprechende Erwartungsnutzen. Die Menge E-admissibler Aktionen besteht also aus allen drei Aktionen a_1, a_2 und a_3.

Wenn im Zuge dieses Beispiels zusätzlich der Priori-Parameter μ unsicher ist, erhöht sich die Dimension des Raums, über den Unsicherheit herrscht, um 1 und das zweidimensionale Bild aus 6.1 wandelt sich zu einem dreidimensionalen.

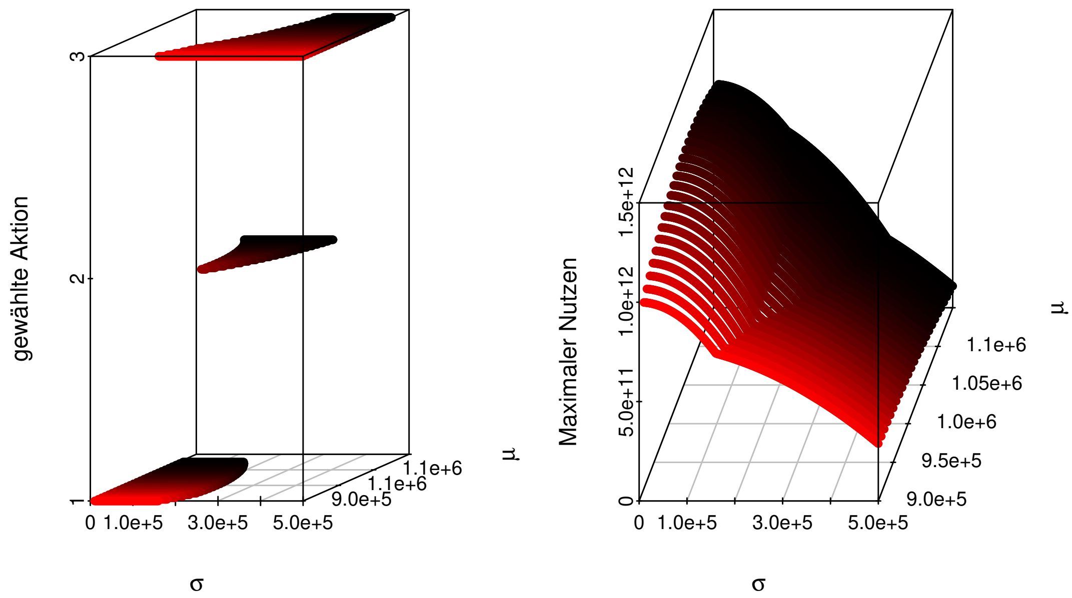

Abb. 6.4: Optimale Aktion im Sinne des Erwartungsnutzens in Abhängigkeit von σ und μ (links), maximal möglicher Nutzen in Abhängigkeit von σ und μ (rechts)

Abbildung 6.4 zeigt für die Situation, dass beide Parameter unsicher sind, die optimale Aktion und den entsprechenden Erwartungsnutzen in Abhängigkeit von den angenommenen Werten für μ und σ. Der Parameter μ ist dabei auf das Intervall $[9 \cdot 10^5, 1.15 \cdot 10^6]$ beschränkt, σ stammt wie oben aus $[10^4, 5 \cdot 10^5]$. Entsprechend der Graphik 6.1 sind auf dem linken Bild die für eine bestimmte μ-σ-Kombination optimale Aktion aus der Aktionenmenge $\{a_1, a_2, a_3\}$ des Beispiels 1 dargestellt, rechts die entsprechenden Erwartungsnutzen. Logischerweise gehören wieder alle Aktionen der Menge E-admissibler Aktionen an. Der Grad des Polynoms bei der Diskretisierung war dabei $N = 10$, die stetige Verteilung wurde also durch 10 Stützstellen ersetzt.

Mithilfe linearer Optimierung kann also in dem Fall, dass lediglich ein oder zwei Parameter einer feststehenden priori-Verteilung unsicher sind, graphisch oder auch numerisch leicht ein Überblick über die Menge E-admissibler Aktionen gewonnen werden. Aus einer möglicherweise sehr großen Anzahl zu wählender Aktionen können so diejenigen herausgefiltert werden, die für bestimmte Kombinationen der priori-Parameter die beste Wahl darstellen. Probleme können sich, wie in Kapitel 5 kurz angedeutet, bei der Genauigkeit der Lösung ergeben: sie ist unter anderem von dem gewählten N für die Diskretisierung abhängig. Je „komplizierter“ bzw. instabiler die Nutzenfunktion einer Aktion ist und je näher die Erwartungsnutzen verschiedener Aktionen zusammen liegen, desto höher sollte der Grad des Polynoms sein, mit dem die stetige Verteilung durch eine diskrete ersetzt wird.

Das Γ-Maximin Kriterium kann in dieser Unsicherheitssituation leicht durch Anwendung von implementierten Optimierungsverfahren für Funktionen angegangen werden: wie die lineare Optimierung aus Kapitel 5.3 ist auch das Berechnen des Erwartungsnutzens einer Aktion über die Diskretisierung nichts anderes als eine Funktion in Abhängigkeit verschiedener Parameter (N und die Parameter der Priori). Der folgende `R`-Code ist die Implementierung einer solchen Funktion, wobei die Priori der Normalverteilung angehört:

```
> e.wert = function(a = 0, b = 1, N = 10, u) {
+     gq = gauss.quad.prob(N, "normal", mu = a, sigma = b)
+     f = as.function(alist(x = , u(x)))
+     res = sum(f(gq$nodes) * gq$weights)
+     return(res)
+ }
```

Die Extremwerte dieser Funktion lassen sich mit implementierten Verfahren, wie der bereits in Kapitel 5.2 verwendeten `optimize`-Funktion, bestimmen: Sei im Kontext des schon mehrmals strapazierten Molekülbeispiels etwa bekannt, dass die Priori eine Normalverteilung mit $\mu = 10^6$ ist und die Standardabweichung σ zwischen 10^4 und $5 \cdot 10^5$ liegt.

Mithilfe der obigen Funktion und der Funktion `optimize` ergeben sich dann für die drei Aktionen a_1, a_2 und a_3 die folgenden minimalen Erwartungsnutzen:

$$\begin{aligned}
\min_\sigma(\mathbb{E}(u_1(\theta))) &= -7.9 \cdot 10^{12} \\
\min_\sigma(\mathbb{E}(u_2(\theta))) &= -4.03 \cdot 10^{12} \\
\min_\sigma(\mathbb{E}(u_3(\theta))) &= -3.74 \cdot 10^{11}.
\end{aligned}$$

Typischerweise (nicht notwendigerweise!) ergibt sich beim Γ-Maximin Kriterium eine einzelne Aktion, die optimal ist. In diesem Fall ist die Aktion a_3 zu wählen, da sie den größten Minimal-Erwartungsnutzen besitzt.

6.2.2 Variante 2: Unsicherheit über Typ der Priori-Verteilung

Nun betrachten wir die Situation, in der auch Unsicherheit bezüglich der Verteilungsfamilie herrscht. Als Information sei lediglich gegeben, dass die Werte der Verteilungsfunktion innerhalb bestimmter Grenzen liegen. Diese Grenzen wiederum seien mit Hilfe bekannter Verteilungen definiert. Im laufenden Beispiel würde sich die wahre Verteilung beispielsweise irgendwo in dem von den Normalverteilungen mit $\mu \in [9 \cdot 10^5, 1.1 \cdot 10^6]$ und $\sigma \in [10^4, 5 \cdot 10^5]$ begrenzten „Band" aus Verteilungsfunktionen befinden.

Wie lässt sich nun im Falle stetiger Verteilungen dieses Band so definieren, dass das anschließende Optimierungsproblem für die Erwartungsnutzen überschaubar bleibt und dass die oberen und unteren Grenzen der Wahrscheinlichkeiten einer F- oder sogar C-Wahrscheinlichkeit entsprechen? Die hier vorgestellte Lösung ist ein Algorithmus zur Erstellung derartiger oberer und unterer Grenzen, der sich auf die Diskretisierung einer stetigen Verteilung stützt:

Man diskretisiere zunächst eine wohlbekannte stetige Verteilung, die sich in etwa in der „Mitte" des zu erwartenden Bandes unterer und oberer Wahrscheinlichkeiten befindet, in N Stützstellen $x_1, \ldots, x_N$. Dadurch erhält man feste Intervalle, denen sich Wahrscheinlichkeiten zuordnen lassen:

$$P(x \in (x_{j-1}, x_j)) := p_j,$$

mit x_0 dem linken Ende des Trägers. Anschließend betrachte man Verteilungen aus der gleichen, leicht zugänglichen Verteilungsfamilie, mit allerdings veränderten, das ganze Spektrum möglicher Verteilungsfunktionen abdeckenden Parametern. Zur Bewertung setzt man sie in Bezug zur diskretisierten Referenzverteilung: für jede Verteilung im (festen) Intervall $(x_i, x_j),\ i < j$ der diskretisierten Referenzverteilung sei die Größe

$$p_{k_{i,j}} \quad := \quad P_k(x \in (x_i, x_j)) \tag{6.5}$$

definiert. k steht dabei für eine durch eine bestimmte Parameterkonstellation eindeutig festgelegte Verteilung aus der betrachteten Familie. Als Priori seien nun alle die (diskreten bzw. diskretisierten) klassischen Verteilungen zulässig, die die Bedingung

$$\pi(\{i, j\}) \;\in\; [\min_k p_{k_{i,j}}, \max_k p_{k_{i,j}}] := [L(p_{ij}), U(p_{ij})] \qquad \forall i, j,\;\; i < j \tag{6.6}$$

erfüllen. Somit werden sämtliche zusammenhängende Intervalle (x_i, x_j) (außer das Gesamtintervall (x_0, x_N), da gelten muss $\pi(\{0, N\}) = 1$) betrachtet, was durch die Nichtadditivität nichtklassischer Wahrscheinlichkeiten nötig ist, um das gesuchte Band durch die Struktur $\mathcal{M}$ auszudrücken. Die $L(p_{ij})$ und $U(p_{ij})$ genügen den Bedingungen der bereits angesprochenen F-Wahrscheinlichkeit nach K. Weichselberger ([20]). Allerdings erfüllen sie nicht die Kriterien 6.4 für eine C-Wahrscheinlichkeit, wie das folgende Zahlenbeispiel zeigt.

Seien dabei die $L(p_{ij})$ und $U(p_{ij})$ durch zwei Verteilungen definiert. Man betrachte die Intervalle $A = (x_{i-2}, x_i)$, $B = (x_{i-1}, x_{i+1})$ und $C = (x_{i-1}, x_i)$. Für eine C-Wahrscheinlichkeit muss $L(A \cup B) + L(A \cap B) \geq L(A) + L(B)$ erfüllt sein, hier also $L(A \cup B) + L(C) \geq L(A) + L(B)$. Die Werte für die Verteilungen und die darus resultierenden $L(p_{ij})$ seien

$$\begin{array}{llcll}
\text{Verteilung 1:} & P(A \cup B) & = & 0.4 & \longrightarrow \quad L(A \cup B) \\
& P(A) & = & 0.3 & \longrightarrow \quad L(A) \\
& P(B) & = & 0.3 & \longrightarrow \quad L(B) \\
& P(C) & = & 0.2 & \\
\\
\text{Verteilung 2:} & P(A \cup B) & = & 0.7 & \\
& P(A) & = & 0.35 & \\
& P(B) & = & 0.35 & \\
& P(C) & = & 0.05 & \longrightarrow \quad L(C)
\end{array}$$

Somit ergibt sich

$$L(A \cup B) + L(C) = 0.4 + 0.05 \; < \; 0.3 + 0.3 = L(A) + L(B),$$

was im Widerspruch zur Bedingung für C-Wahrscheinlichkeiten steht.

Das gesuchte Band lässt sich also durch die Verteilungsfunktionen aus einer bekannten, in Frage kommenden Verteilungsfamilie finden und wird von F-Wahrscheinlichkeiten begrenzt. Dabei spielen die Gewichte der diskretisierten Verteilung nur mittelbar eine Rolle, wichtiger ist in diesem Algorithmus die Lage der Stützpunkte. Sie bestimmt die Intervalle (x_i, x_j) und ist das einzige Charakteristikum, das von dieser Referenzverteilung an die übrigen Verteilungen weitergegeben wird: Die mittlere Verteilung nimmt also lediglich durch ihren Träger Einfluss auf p_{k_j} und somit auf die $L(p_j)$ und $U(p_j)$.[3]

[3] Ein weiterer Punkt ist bei dieser Art der Beschreibung des Bandes zu beachten: Die

Dargestellt ist die Vorgehensweise zum Finden der oberen und unteren Grenzen ansatzweise in Abbildung 6.5 mithilfe einer mittleren Referenzverteilung, sowie zwei anderen Verteilungsfunktionen, die die gleiche Gestalt aufweisen, jedoch nach links bzw. rechts verschoben sind.

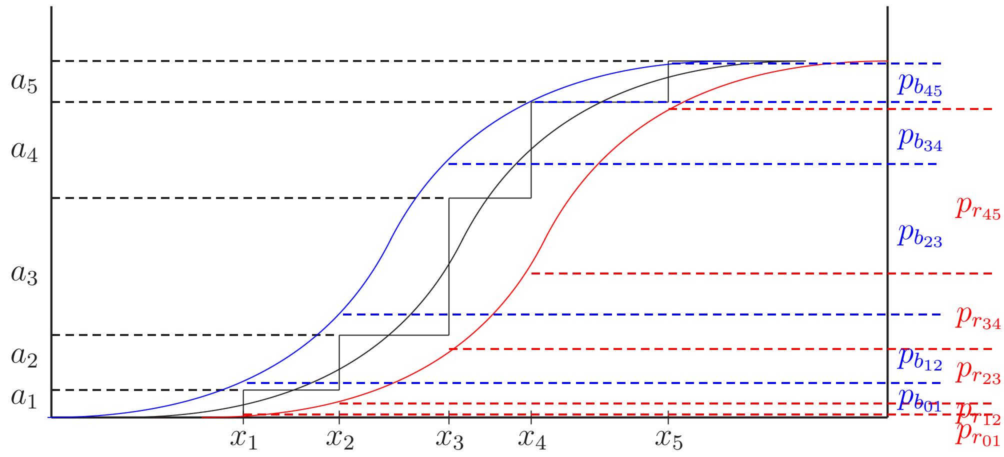

Abb. 6.5: Finden von oberen und unteren Wahrscheinlichkeiten (Skizze)

Die schwarze, mittlere Kurve stellt die stetige Referenzverteilung dar, die Stufen ihre Diskretisierung mit den Stützstellen x_j. Die Wahrscheinlichkeiten p_j sind links dargestellt. Zwischen den beiden anderen Verteilungen (rot und blau) ist das gesuchte Band, in dem sich alle hier möglichen Verteilungen befinden. Aus den klassischen Wahrscheinlichkeiten $p_{k_{ij}}$, die rechts für $i = j-1$ dargestellt sind, leiten sich die oberen und unteren Grenzen $L(p_j)$ und $U(p_j)$ für die $\pi(\{i,j\})$ ab.

Betrachtet man nun unter diesen Voraussetzungen wieder das Optimierungsproblem des Erwartungsnutzens, so muss man in (5.8) zusätzlich beachten, dass die $\pi(\{\theta_j\}) := \pi(\{j-1,j\})$ nicht einer vorgegebenen Verteilung an-

oberen und unteren Wahrscheinlichkeiten werden lediglich über alle zusammenhängenden Intervalle hinweg definiert. Es wird also nicht die gesamte Potenzmenge betrachtet. Die dabei möglicherweise verlorengehende Information wird aus Effizienzgründen in Kauf genommen: Bei der Potenzmenge müssten an dieser Stelle $2^{(NM)} - 2$ Teilmengen betrachtet werden, nimmt man die hier beschriebene Methode, so bleiben $\frac{1}{2} \cdot (NM+1)NM - 1$ zu betrachtende Intervalle (bei einer Aufteilung des Trägers in $N \cdot M$ Intervalle).

gehören, sondern lediglich die Bedingung (6.6) erfüllen müssen. Neben den Nebenbedingungen an λ

$$\begin{aligned} \sum_i \lambda(a_i) &= 0 \\ \lambda(a_i) &\geq 0 \quad \forall i. \end{aligned}$$

müssen also auch die Forderungen an $\pi(\cdot)$ beachtet werden. Eine Möglichkeit, das zu berücksichtigen, bietet ein von Utkin und Augustin([17]) entwickelter, hier leicht modifizierter Algorithmus zur Berechnung E-admissibler Aktionen unter komplexer Unsicherheit. Er zeigt einen Weg, wie man bei einer diskreten, aber unsicheren Zustandsverteilung mithilfe linearer Optimierung feststellen kann, ob eine Aktion a_i E-admissibel ist oder nicht.

Dazu betrachte man die Menge aller Wahrscheinlichkeitsmaße aus der Struktur $\mathcal{M}$, für die eine Aktion a_i optimal ist:

$$\Pi_i = \left\{ \pi(\cdot) \in \mathcal{M} \middle| \sum_{j=1}^{N} u_i(\theta_j)\pi(\{\theta_j\}) \geq \sum_{j=1}^{N} u_l(\theta_j)\pi(\{\theta_j\}), \quad \forall l = 1, \ldots, n \right\}$$

Bei der Optimierung führe man zunächst die Hilfsvariable z ein und löse anschließend das Problem:

$$\begin{aligned} z &\longrightarrow \max_{(\pi^T, z)^T} \\ \sum_{j=1}^{N} u_i(\theta_j)\pi(\{\theta_j\}) &\geq \sum_{j=1}^{N} u_l(\theta_j)\pi(\{\theta_j\}) \qquad (6.7) \\ \forall l = 1, \ldots, n & \\ \sum_{j=1}^{N} \pi(\{\theta_j\}) = z,\ z \leq 1, & \qquad \pi(\{\theta_j\}) \geq 0,\ j = 1, \ldots, N \\ \underline{b}_{s,t} \leq \sum_{j=1}^{N} f_{s,t}(\{\theta_j\})\pi(\{\theta_j\}) \leq \overline{b}_{s,t}, & \qquad s < t, \qquad (6.8) \\ \text{mit} \quad f_{s,t}(\{\theta_j\}) &= \begin{cases} 1 & \text{für } s \leq j \leq t \\ 0 & \text{sonst} \end{cases} . \end{aligned}$$

Die $\underline{b}_{s,t}$ und $\overline{b}_{s,t}$ sind dabei die unteren bzw. oberen Schranken für $\pi(\{r, t\})$. Insbesondere für die Wahrscheinlichkeiten $\pi(\{\theta_j\})$ (mit $\theta_j := (x_{j-1}, x_j)$) stellen sie bei $s = t - 1$ also die unteren und oberen Wahrscheinlichkeiten $L(\theta_j)$

und $U(\theta_j)$ dar, die mithilfe des vorhergehenden Algorithmus gefunden wurden. Die Nebenbedingung (6.7) besagt, dass der Nutzen der Aktion a_i mindestens so hoch ist wie der jeder anderen der n möglichen Aktionen unter einem $\pi(\theta)$. Dann und nur dann, wenn der optimale Wert von $z = 1$ ist, stellt a_i eine E-admissible Aktion dar, d.h. Π_i ist nicht leer.

Allgemein gilt für die lineare Optimierung in dem Paket `lpSolve`: $c'x \to \max$ unter den Nebenbedingungen $Ax \leq b, x \geq 0$. Auf den beschriebenen Algorithmus angewandt, ergibt dies für die Aktion a_i das folgende Optimierungsproblem:

$$\begin{aligned} c &:= (1, 1, \ldots, 1) \\ x &:= (\pi(\{\theta_1\}), \ldots, \pi(\{\theta_N\})) \\ c'x &= z \end{aligned}$$

Sei $d_{l_k j} = u_k(\theta_j) - u_l(\theta_j)$ die Differenz der Nutzen zwischen Aktion a_k und der zu untersuchenden Aktion a_l bei Umweltzustand θ_j, wobei $k \in \{1, 2, \ldots, n\} \setminus \{l\}$. Die Ungleichung $Ax \leq b$ der Nebenbedingungen hat somit folgende Gestalt:

$$
\begin{pmatrix}
d_{l_1 1} & d_{l_1 2} & & \dots & & d_{l_1 N} \\
\vdots & \vdots & & & & \vdots \\
d_{l_n 1} & d_{l_n 2} & & \dots & & d_{l_n N} \\
1 & 1 & & \dots & & 1 \\
-1 & 0 & & \dots & & 0 \\
0 & -1 & & \dots & & 0 \\
 & & & \ddots & & \\
0 & 0 & & \dots & & -1 \\
1 & 0 & & \dots & & 0 \\
0 & 1 & & \dots & & 0 \\
 & & & \ddots & & \\
0 & 0 & & \dots & & 1 \\
1 & 1 & 0 & \dots & & 0 \\
0 & 1 & 1 & \dots & & 0 \\
 & & & \ddots & & \\
0 & 0 & & \dots & 1 & 1 \\
-1 & -1 & 0 & \dots & & 0 \\
 & & & \vdots & & \\
0 & -1 & -1 & \dots & -1 & -1
\end{pmatrix}
\cdot
\begin{pmatrix}
\pi(\{\theta_1\}) \\
\pi(\{\theta_2\}) \\
\vdots \\
\pi(\{\theta_N\})
\end{pmatrix}
\leq
\begin{pmatrix}
0 \\
\vdots \\
0 \\
1 \\
-\underline{b}_{0,1} \\
-\underline{b}_{1,2} \\
\vdots \\
-\underline{b}_{N-1,N} \\
\overline{b}_{0,1} \\
\overline{b}_{1,2} \\
\vdots \\
\overline{b}_{N-1,N} \\
-\underline{b}_{0,2} \\
-\underline{b}_{1,3} \\
\vdots \\
-\underline{b}_{N-2,N} \\
\overline{b}_{0,2} \\
\vdots \\
\overline{b}_{1,N}
\end{pmatrix}
$$

Somit wird die Matrix der Nebenbedingungen für eine große Anzahl an Stützstellen schnell sehr groß. Die genaue Anzahl der Nebenbedingungen in der Matrix liegt bei $MN(MN+1)+n-1$ bei n zu untersuchenden Aktionen und MN Stützstellen. Auch die Erstellung der $\underline{b}_{i,j}$ und $\overline{b}_{i,j}$ wird mit $\left(\frac{(MN+1)MN}{2}-1\right)\cdot\ell-1$ zu betrachtenden Intervallen (bei ℓ möglichen Priori-Parameterkonstellationen) schnell unübersichtlich und irgendwann auch für den Rechner ein Problem. Aus diesem Grund wird auch nicht die gesamte Potenzmenge der Intervalle betrachtet (siehe Fußnote 3). Die Anzahl der Stützstellen bei der Diskretisierung sowie die Anzahl in Erwägung zu ziehender Priori-Parameter sollte nicht zu groß gewählt werden, um numerisch instabile Ergebnisse zu vermeiden.

Die Lösungen des linearen Optimierungsproblems liefern wiederum bereits implementierte Verfahren. Alle $x = (\pi(\{\theta_1\}), \ldots, \pi(\{\theta_N\}))$, die die Nebenbedingungen erfüllen, bilden, wie in Kapitel 5.3 erwähnt, einen konvexen, N-dimensionalen Polyeder. Dieser stimmt im Falle einer E-admissiblen Aktion in Teilen mit der Struktur $\mathcal{M}$ überein, die eine Dimension weniger besitzt. Schematisch dargestellt ist der Zusammenhang von Struktur und dem Simplex der Lösungen des linearen Optimierungsproblems in der folgenden Skizze für $|\Theta| = 2$ und entsprechende obere und untere Grenzen für $\pi(\{\theta_1\})$ und $\pi(\{\theta_2\})$:

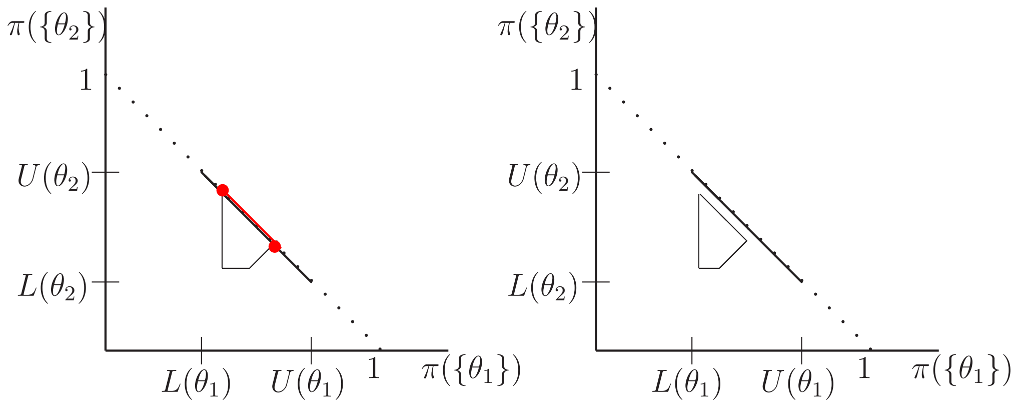

Abb. 6.6: Zusammenhang von Struktur und Simplex der Optimierung: E-admissible Aktion (links), nicht E-admissible Aktion(rechts)

Das hier 2-dimensionale Simplex der möglichen Lösungen für das Optimierungsproblem wird im linken Bild auf einer Seite von der Struktur, die lediglich eindimensional ist, begrenzt. Auf dem rechten Bild dagegen berührt das Simplex die Struktur überhaupt nicht, dort handelt es sich um eine nicht E-admissible Aktion.

Zur Erklärung: Das Simplex der linearen Optimierung und die Struktur müssen zunächst einmal die selben Bedingungen erfüllen: Die $\pi(\{i,j\})$ müssen zwischen $\overline{b}_{ij}$ und $\underline{b}_{i,j}$ liegen. Für die Struktur gilt jedoch außerdem $\pi(\{\theta_1\}) + \pi(\{\theta_2\}) = 1$ (daraus folgt, dass die Struktur hier zweidimensional ist), während im Simplex des Optimierungsproblems $\pi(\{\theta_1\}) + \pi(\{\theta_2\}) \leq 1$ und die Bedingung (6.7) erfüllt sein müssen. Im linken Bild werden auch von Werten in der Struktur die Bedingungen (6.7) erfüllt und somit gibt es klassische

Wahrscheinlichkeiten, für die die entsprechende Aktion a_i optimal ist. Im Bild rechts reicht das Simplex aller möglichen Lösungen für $Ax \leq b$ nicht an die Struktur heran, die Aktion ist also nicht E-admissibel.

Um alle E-admissiblen Aktionen zu bestimmen, werden $n = |\mathbb{A}|$ derartige lineare Optimierungen benötigt. Die Funktion `noncl.exp.value` stellt zusammen mit der Hilfsfunktion `eadmissible` (beide Funktionen siehe Anhang) die entsprechende Implementierung dar: Für eine gegebene Menge $\mathbb{A}$ an Nutzenfunktionen, einer mittleren Referenzverteilung für die Priori, sowie einer Reihe an möglichen Parametern für diese Verteilungsfamilie, berechnen die Funktionen die Menge an E-admissiblen Aktionen aus der Menge $\mathbb{A}$. Verwendet werden dabei die vorgestellten Algorithmen zur Erstellung von oberen und unteren Wahrscheinlichkeiten, sowie zum Finden E-admissibler Aktionen. Es kann in der Funktion zudem gewählt werden zwischen der Diskretisierung nach Theorem 1 und der mit einer Aufteilung des Trägers in M Intervalle, also Theorem 2.

Es ist mithilfe linearer Optimierung also leicht möglich, zu bestimmen, ob eine Aktion E-admissibel ist oder nicht. Auch für das Γ-Maximin Kriterium kann die lineare Optimierung eingesetzt werden. Im Gegensatz zur E-Admissibilität gibt es hier auch typischerweise eine eindeutige Lösung, sprich eine einzelne optimale Aktion, die zu wählen ist. Eine Möglichkeit, die Γ-Maximin Aktion zu finden, ist die, $n = |\mathbb{A}|$ lineare Optimierungen durchzuführen, um jede Aktion zu minimieren. Anschließend sucht man diejenige Aktion, die hier den höchsten Wert aufweist. Durch ein von Utkin und Augustin in [17] beschriebenes Vorgehen kann man sich jedoch dabei sogar auf ein einziges Optimierungsproblem beschränken.[4]

Für eine Γ-Maximin Aktion muss das Problem

$$\min_{\pi \,\in\, \mathcal{M}} \sum_{i=1}^{N} \left(\sum_{s=1}^{n} u_s(\theta_i)\lambda(a_s) \cdot \pi_j \right) \longrightarrow \max_{\lambda}$$

unter der Nebenbedingung $\sum_i \lambda(a_i) = 1$ gelöst werden.

Das Vorgehen von Utkin und Augustin nutzt nun aus, dass im primalen und im dualen Problem die Zielfunktionen am Optimum die gleichen Werte besitzen. Das lineare Optimierungsproblem hat somit folgende Gestalt:

$$\max_{c,C,D,\lambda} \{k + \underline{B}C - \overline{B}D\} \tag{6.9}$$

[4] Der Algorithmus hat außerdem den Vorteil, dass er die gemischte Erweiterung des Aktionenraums betrachtet: beim Γ-Maximin Kriterium können auch randomisierte Aktionen optimal sein (siehe [1]).

unter den Bedingungen $k \in \mathbb{R},\ C, D \in \mathbb{R}_+^r,\ \lambda \cdot \mathbf{1} = 1$ und

$$k + F_j(C - D) \leq \sum_{s=1}^{n} u_s(\theta_j)\lambda(a_s).$$

Dabei wurden die folgenden Abkürzungen verwendet:

$$\begin{array}{rcl} \underline{B} & := & (\underline{b}_{0,1}, \ldots, \underline{b}_{1,N}) \ \in \ \mathbb{R}_+^r \\ \underline{B} & := & (\overline{b}_{0,1}, \ldots, \overline{b}_{1,N}) \ \in \ \mathbb{R}_+^r \\ F_j & := & (f_{0,1}(\{\theta_j\}), \ldots, f_{1,N}(\{\theta_j\})) \ \in \ \mathbb{R}_+^r \\ & & \text{mit } f_{s,t}(\{\theta_j\}) = \begin{cases} 1 & \text{für } s \leq j \leq t \\ 0 & \text{sonst} \end{cases} \end{array}$$

In der Form $c'x \to \max$ unter $Ax \leq b$ sieht dieses Optimierungsproblem hier wie folgt aus:

$$\begin{array}{rcl} c & = & (1, \underline{B}, -\overline{B}, 1, \ldots, 1) \\ x & = & (k, C, D, \lambda) \end{array}$$

Für die Ungleichung der Nebenbedingungen gilt:

$$\begin{pmatrix} 1 & F_1 & -F_1 & -u_1(\theta_1) & \ldots & -u_n(\theta_1) \\ 1 & F_2 & -F_2 & -u_1(\theta_2) & \ldots & -u_n(\theta_2) \\ & \vdots & \vdots & \vdots & & \vdots \\ 1 & F_N & -F_N & -u_1(\theta_N) & \ldots & -u_n(\theta_N) \\ 0 & \ldots & 0 & 1 & \ldots & 1 \\ 0 & \ldots & 0 & -1 & \ldots & -1 \end{pmatrix} \cdot \begin{pmatrix} k \\ C \\ D \\ \lambda \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ -1 \end{pmatrix}$$

Die optimale Aktion nach dem Γ-Maximin Kriterium mithilfe der beschriebenen Optimierung liefert wiederum die von mir implementierte Funktion `noncl.exp.value`, dieses Mal zusammen mit der Hilfsfunktion `maximin` (siehe Anhang): zusätzlich zur Menge der E-admissiblen Aktionen gibt sie die Γ-Maximin Aktion mithilfe der oben beschriebenen Optimierung zurück.

Bsp. 5 (Fortsetzung: Beispiel 1):

Im Folgenden werde ich im Kontext des laufenden Beispiels mit den drei möglichen Aktionen a_1, a_2 und a_3 und deren Nutzenfunktionen $u_i(\theta)$ die beschriebenen Algorithmen in Unsicherheitssituationen anwenden. In Erinnerung gerufen seien dazu die Nutzenfunktionen der drei Aktionen

$$\begin{aligned} u_1(\theta) &= -10\ \theta^2 + 2.1 \cdot 10^7\theta - 9.8 \cdot 10^{12}, \\ u_2(\theta) &= -5\ \theta^2 + 1.1 \cdot 10^7\theta - 5 \cdot 10^{12} - 3 \cdot 10^{10} \quad \text{und} \\ u_3(\theta) &= -2\ \theta^2 + 3.2 \cdot 10^6\theta + log(|\theta|)^{10} - 7 \cdot 10^{11}, \end{aligned}$$

und ihre jeweiligen Verläufe (Abbildung 5.1). Über die Verteilung der Zustände sei dabei bekannt, dass sie sich dieses Mal irgendwo in dem aus Normalverteilungen begrenzten Band mit $\mu \in [9.7 \cdot 10^5, 1.03 \cdot 10^6]$ und $\sigma \in [10^5, 2 \cdot 10^5]$ befindet, jedoch nicht auf die Familie der Normalverteilung eingeschränkt ist.

Zunächst gilt es, die oberen und unteren Grenzen für das Band der Verteilungen zu finden. Als mittlere Referenzverteilung bietet sich eine Normalverteilung um $\mu = 10^6$ mit $\sigma = 1.5 \cdot 10^5$ an. Sie wird mit einer vorgegebenen Anzahl N an Stützstellen diskretisiert. Anschließend werden die Normalverteilungen, die sich innerhalb der vermuteten Grenzen befinden, zum Finden der $\underline{b}_{i,j}$ und $\overline{b}_{i,j}$ benutzt. In diesem Beispiel verwende ich bei der Diskretisierung Theorem 1 mit $N = 30$ Stützstellen, sowie anschließend alle Normalverteilungen $N(\mu, \sigma)$, für die gilt $\mu \in [9.7 \cdot 10^5, 9.8 \cdot 10^5, \ldots, 1.03 \cdot 10^6]$ und $\sigma \in [10^5, 1.1 \cdot 10^5, \ldots, 2 \cdot 10^5]$.

Ein Aufruf der bereits beschriebenen Funktion `noncl.exp.value` mit diesen Angaben liefert beim Kriterium der E-Admissibilität für λ den Vektor $(1, 1, 0)$ zurück: Die Aktionen a_1 und a_2 sind also unter diesen Vorgaben E-admissibel, da ihr optimaler Wert für z gleich 1 ist, Aktion a_3 ist dagegen nicht E-admissibel. Aktion a_1 ist die Aktion nach dem Γ-Maximin Kriterium ($\lambda = (1, 0, 0)$) mit einem errechneten minimalen Erwartungsnutzen von $8.05 \cdot 10^{11}$.

Ob tatsächlich Aktion a_1 gewählt wird, hängt von den individuellen Risikopräferenzen des Entscheiders ab. Wie schon erwähnt, spiegelt das Γ-Maximin Kriterium eine äußerst pessimistische Einschätzung (beispielsweise in einer Situation mit einem feindlichen Gegenspieler) wider. Um zu einer endgültigen Entscheidung zu kommen, können neben diesen beiden auch weitere Kriterien herangezogen werden (siehe dazu [18]).

Genauigkeit der Ergebnisse

Bis hierher wurde im Beispiel zur Diskretisierung Luceños erstes Theorem verwendet. Da das Finden der $\underline{b}_{i,j}$ und $\overline{b}_{i,j}$ aber auf den Verteilungsfunktionen basiert, bietet sich an dieser Stelle sogar eher das zweite Theorem an, das eine neue Zufallsvariable kreiert, die, was die Verteilung betrifft, der ursprünglichen ähnlicher ist als eine mit Theorem 1 diskretisierte Variable (siehe Kapitel 3). Tatsächlich ergeben sich in diesem Fall etwas andere Werte. Den Grund hierfür kann man an den Graphiken in 6.7 erkennen.

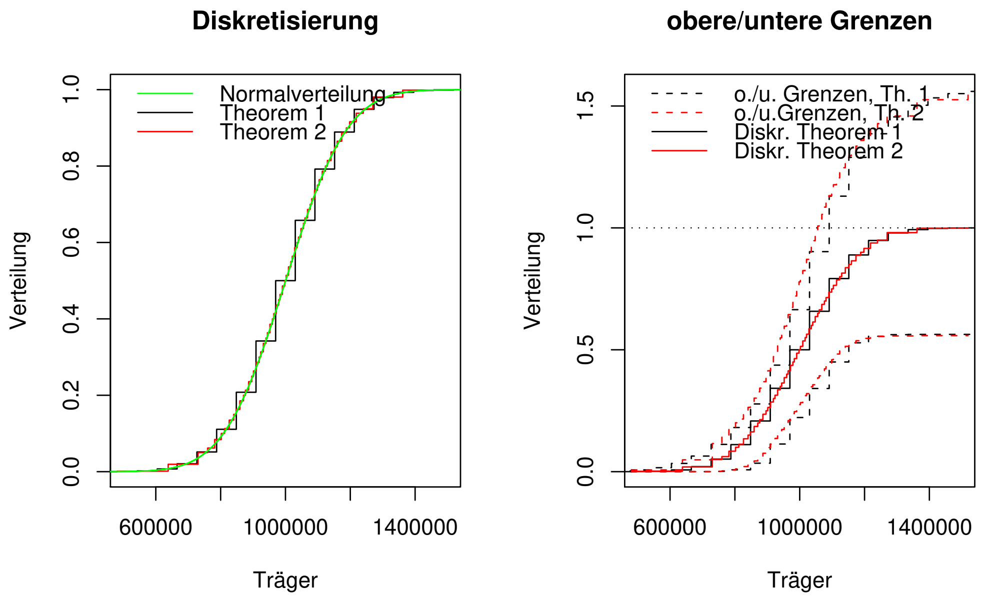

Abb. 6.7: Diskretisierung mit Theorem 1 bzw. Theorem 2 (links). Obere/untere Grenzen für Wahrscheinlichkeiten $\pi(\{\theta_j\})$ berechnet mit beiden Theoremen (rechts).

Die Abbildung zeigt die Unterschiede der beiden Techniken bei der Diskretisierung und dem anschließenden Finden oberer und unterer Grenzen: Während bei Theorem 1 deutlich der Unterschied zur Verteilungsfunktion der Originalverteilung zu sehen ist, ist die Kurve bei Theorem 2 kaum mehr von der der stetigen Verteilung zu unterscheiden (Bild links). Bei Theorem 1 war dabei $N = 60$, beim zweiten Theorem wurden mit $N = 3$ und $M = 20$ ebenfalls 60 Stützstellen verwendet, wobei hier zur Wahl der Intervalle die Verteilungsfunktion verwendet wurde.

Das rührt daher, dass bei der einfachen Gaußschen Quadratur die Stützpunkte über den ganzen Träger mithilfe eines einzigen Polynoms festgelegt werden, das zwar die Gewichte entsprechend verteilt, aber dennoch dafür sorgt, dass es auch relativ viele Stützstellen in den „unwichtigen“ äußeren Bereichen des Trägers gibt. In das Finden der $\underline{b}_{i,j}$ und $\overline{b}_{i,j}$ fließt, wie erläutert, direkt jedoch nur die Lage der Stützstellen und nicht deren Gewichte ein. Die Approximation in den inneren Bereichen ist also relativ grob und dadurch hier weniger geeignet als die bei Theorem 2. Denn bei der Diskretisierung mit vorheriger Aufteilung in Intervalle werden, wenn die Intervalle geschickt gewählt wurden, speziell in mittleren Bereichen sehr viele Stützstellen erzeugt und so wird für eine sehr gute Annäherung an die stetige Kurve gesorgt. Somit unterscheiden sich am Ende auch die oberen und unteren Grenzen $\overline{b}_{i,j}$ und $\underline{b}_{i,j}$, wobei die Methode mit Intervallen die bessere, weil feinere Begrenzung liefert. Im Bild rechts kann man diese Unterschiede in den oberen und unteren Grenzen erkennen, dargestellt sind die $\underline{b}_{j-1,j}$ und $\overline{b}_{j-1,j}$.

In der folgenden Graphik 6.8 sieht man die Auswirkungen dieser Unterschiede. Sie zeigt die Ergebnisse der linearen Optimierungen für das Γ-Maximin Kriterium mit beiden Theoremen.

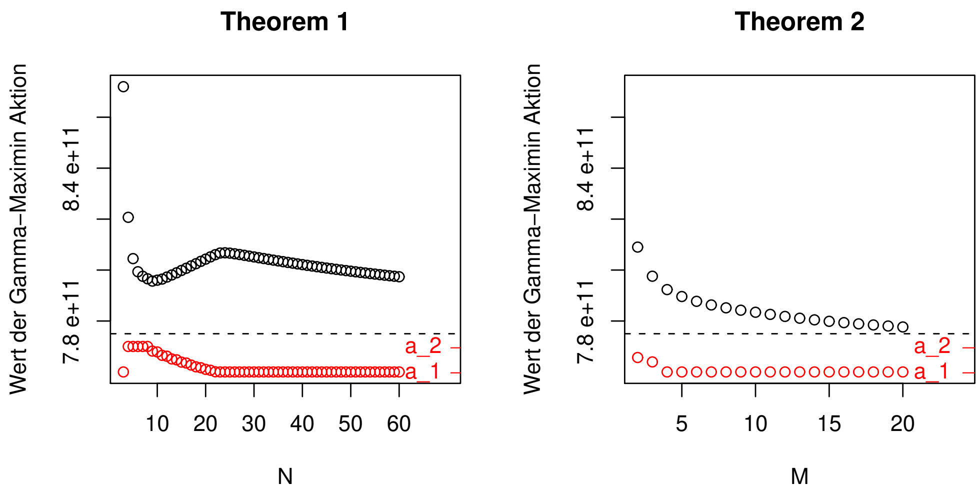

Abb. 6.8: Γ-Maximin Aktion, berechnet mit Theorem 1 bzw. Theorem 2. Oberer Teil der Graphik: Erwartungswert der Γ-Maximin Aktion, unterer Teil: welche Aktion ist Γ-Maximin?

Dargestellt sind jeweils in der oberen Kurve die berechneten minimalen Erwartungsnutzen der Γ-Maximin Aktion. Der untere Teil der Graphik stellt diese Aktion dar. Bei den Ergebnissen von Theorem 1 ist dabei zu sehen, dass sich zu Beginn die Ergebnisse für die optimale Aktion erst langsam einpendeln: zunächst ist Aktion a_1 optimal (unten), anschließend a_2 (oben), danach eine Kombination der beiden (die optimale Aktion würde also aus der gemischten Erweiterung stammen, was durchaus möglich ist (siehe [1])), am Ende bleibt das Ergebnis stabil bei a_1. Entsprechend verhalten sich auch die berechneten Erwartungswerte: ab einem N von 20 zeigen sie zwar einen Trend nach unten, sind aber trotz allem relativ stabil.

Auch beim zweiten Theorem (hier wurde $N = 3$ verwendet) schwanken die Ergebnisse für die optimale Aktion zu Beginn. Für sehr kleine M stammt die entsprechende Lösung auch hier aus der gemischten Erweiterung. Relativ schnell ist a_1 dann jedoch die eindeutige Γ-Maximin Aktion. Entsprechend schwanken die Erwartungswerte weniger als bei Theorem 1, sondern sind hier von Anfang an stabil mit einem Trend nach unten.

Auffällig ist, dass zwar die optimale Aktion am Ende mit beiden Verfahren die gleiche ist, das Niveau des berechneten Erwartungsnutzens jedoch etwas, wenn auch nur leicht, versetzt liegt. Aus den bereits erläuterten Gründen sind dabei prinzipiell die Ergebnisse der mit Theorem 2 durchgeführten Diskretisierung zu bevorzugen: Bei ihr ist die Begrenzung der Verteilungen in den entscheidenden Bereichen weniger grob, wodurch sich die Ergebnisse auch schneller stabilisieren. Die Diskretisierung mit dem ersten Theorem wird zwar mit steigendem N auch feiner und somit besser, für gute Ergebnisse, also feine Begrenzungen, wird jedoch ein sehr hohes N gefordert, was nur mit einigem Rechenaufwand möglich ist.

Dass mit der Diskretisierung generell jedoch nur eine Approximation der Wirklichkeit erreicht werden kann, wird daran deutlich, dass die berechneten Erwartungswerte auch für hohe M bzw. N einen leichten Trend zeigen: wird die Diskretisierung also feiner, so verändern sich die Ergebnisse noch graduell. Deshalb sollten Entscheidungen, die knapp ausfallen, prinzipiell mit veränderten Vorgaben, sprich unterschiedlichen M und N, mehrmals berechnet werden, um die Ergebnisse zu validieren.

7. Zusammenfassung und Ausblick

Im Laufe der vorliegenden Arbeit habe ich Verfahren und Anwendungen in der Statistik beleuchtet, in denen die Ergebnisse von Luceños Theoremen gewinnbringend eingesetzt werden können. Die Liste dieser Anwendungsgebiete ist dabei natürlich keineswegs vollständig. Luceño selbst gibt in seinem Aufsatz noch andere Möglichkeiten der Anwendung an, wie z.B. im Bereich statistischer Transformation oder für den sogenannten „process capability index“, die ich hier nicht behandelt habe und die sich für weitere Untersuchungen anbieten.

Kapitel 3 dieser Diplomarbeit zeigt für das Beispiel der Standardnormalverteilung die Güte der diskreten Approximation nach Theorem 2. Über alle Tests hinweg wird dabei deutlich: stetige und diskrete Verteilung sind ab einer gewissen Anzahl an Intervallen, in die der Träger aufgeteilt wird, durch Stichproben kaum mehr voneinander zu unterscheiden. Diese Anzahl ist allerdings auch stark von den Eigenheiten dieser Intervalle abhängig: je besser diese an die stetige Verteilung angepasst sind, desto eher gelingt eine gute Approximation. Im Gegensatz zu der ebenfalls vorgestellten naiven Diskretisierungsmethode hat die nach Luceño zusätzlich den Vorteil, dass zumindest die ersten $2N-1$ Momente der beiden Verteilungen gleich sind.

Ziel des vierten Kapitels ist es, typische Situationen für numerische Integration in der Statistik darzustellen und die Anwendung der Gaußschen Quadratur in diesem Kontext zu zeigen. Das Hauptaugenmerk liegt dabei auf dem Vergleich mit der standardisierten `integrate`-Methode in `R`. Es wird deutlich, dass die „von Hand“-Methode nach Gauß dabei in bestimmten Situationen durchaus Vorteile haben kann. Die Liste der Beispiele ist dabei kurz, liegt doch der Schwerpunkt dieser Diplomarbeit auf den Kapiteln 5 und 6, also der Anwendung der Gaußschen Quadratur in der Entscheidungstheorie. Die Gauß-Laguerre und Gauß-Legendre Polynome, die bei Integralen mit der Gamma- bzw. Gleichverteilung benötigt werden, habe ich aus Platzgründen bei den Beispielen übergangen. Vor allem ein Vergleich der numerischen Integrationsergebnisse bei der Gleichverteilung zwischen den Gauß-Legendre Polynomen und der standardisierten `integrate`-Funktion, die mit den `QUADPACK`-Prozeduren auch auf Stützstellen dieser Polynome zurückgreift, steht noch aus.

Was die Entscheidungstheorie betrifft, so zeigen sich in den Kapiteln 5 und 6 vielfältige Anwendungsmöglichkeiten der Gaußschen Quadratur bzw. der Diskretisierung nach Luceño. So kann sich beispielsweise, speziell bei datengestützten Entscheidungsproblemen mit sehr vielen Aktionen, ein Effizienzgewinn im Sinne schnellerer Berechnung durch Anwendung fester Gauß-Stützstellen bemerkbar machen. Diese Stützstellen können auch bei Optimierungen unter linearen Nebenbedingungen verwendet werden. In der Situation unter komplexer Unsicherheit habe ich diese Stützstellen zum Finden oberer und unterer (diskreter) Grenzen für die (klassischen) Zustandswahrscheilichkeiten benutzt. Der Algorithmus zum Finden dieser Grenzen stellt dabei allerdings nur eine Möglichkeit von möglicherweise vielen dar, wie bei komplexer Unsicherheit das Problem einer stetigen Zustandsverteilung angegangen werden kann. Angesprochen seien hier Methoden, die auf dem Betrachten der gesamten Potenzmenge der Intervalle beruhen und speziell Methoden, die zu C-Wahrscheinlichkeiten als Grenzen führen.

Als Entscheidungskriterien in der Situation unter komplexer Unsicherheit habe ich das Kriterium der E-Admissibilität, sowie das Γ-Maximin Kriterium betrachtet. Neben diesen gibt es noch weitere, wie beispielsweise das der Maximalität oder der Intervall Dominanz. Die Algorithmen zur Identifizierung einer E-admissiblen Aktion bzw. der Γ-Maximin Aktion stammen dabei beide aus einem Aufsatz von Utkin und Augustin aus dem Jahr 2005 ([17]). Dabei ist das Verfahren der linearen Optimierung zum Finden einer Γ-Maximin Aktion nur eine von zwei darin genannten Möglichkeiten. Somit bietet sich eine Fülle weiterer Untersuchungen zur Anwendung dieser Diskretisierungsmethode in Entscheidungssituationen unter komplexer Unsicherheit an.

Ziel der Diplomarbeit war es, die Bedeutung der Diskretisierungstechnik für verschiedene Bereiche der Statistik zu veranschaulichen und diese Anwendungen zu implementieren. Die vorliegenden Untersuchungsergebnissen zeigen dabei, dass die Diskretisierungsmethode Luceños in vielen Bereichen der Statistik von Nutzen sein kann. Aufbauend oder auch gänzlich losgelöst von dieser Arbeit bieten sich zu diesem Thema weitere, wie die in diesem Kapitel erwähnten, Untersuchungen an.

Literaturverzeichnis

[1] Augustin T., 2002, *Expected utility within a generalized concept of probability - a comprehensive framework for decision making under ambiguity*, Stat. Pap. 43, 5-22.

[2] Carroll R.J., Ruppert D., Stefanski L.A., Crainiceanu C., 2006, *Measurement Error in Nonlinear Models: A Modern Perspective, Second Edition*, Chapman&Hall, London.

[3] Deuflhard P., Hohmann A., 1993, *Numerische Mathematik I - eine algorithmisch orientierte Einführung*, de Gruyter, Berlin.

[4] Ellsberg D., 1961, *Risk, ambiguity and the Savage axioms*, Q. J. Econ. 75, 643-669.

[5] Gautschi W., 1968, *Construction of Gauss-Christoffel Quadrature Formulas*, Math. Comp. 22, 251-270.

[6] Gautschi W., 1970, *On the Construction of Gaussian Quadrature Rules from Modified Moments*, Math. Comp. 24, 245-260.

[7] Gautschi W., 1994 *Algorithm 726: ORTHPOL – A Package of Routines for Generating Orthogonal Polynomials and Gauss-Type Quadrature Rules*, ACM Trans. Math. Software 20 (1), 21-62.

[8] Kenney J. F., Keeping E. S., 1951 *Mathematics of Statistics, Part 2*, 2. Ausgabe, D. Van Nostrand Company, INC, New York.

[9] Luceño A., 1999, *Discrete approximations to continous univariate distributions – an alternative to simulation*, J.R. Statist. Soc. B 61, Part 2, 345-352.

[10] Patel Jagdish K, Read Campbell B., 1982, *Handbook of the Normal Distribution*, Marcel Dekker, New York.

[11] Piessens R., deDoncker-Kapenga E., Uberhuber U., Kahaner D., 1983, *Quadpack: a Subroutine Package for Automatic Integration* Springer Berlin Heidelberg.

[12] Press et al., 1987, *Numerical Recipes*, Cambridge University Press.

[13] Rüger B., 1999, *Test- und Schätztheorie, Band I: Grundlagen*, Oldenbourg Verlag, München.

[14] Rüger B., 2002, *Test- und Schätztheorie, Band II: Statistische Tests*, Oldenbourg Verlag, München.

[15] Sack R.A., Donavan A.F., 1971/72, *Numerische Mathematik*, vol. 18, S. 465-478.

[16] Stoer J., Bulirsch R., 1999, *Numerische Mathematik I*, Springer Berlin Heidelberg.

[17] Utkin L.V., Augustin T., 2005, *Powerful algorithms for decision making under partial prior information and general ambiguity attitudes* in: Cozman F.G., Nau R., Seidenfeld T.(Hrg.), ISIPTA '05, Proceedings of the Fourth International Symposium on Imprecise Probabilities and their Applications, Pittsburgh, PA, USA, SIPTA, Manno (CH), 2005, 349-358.

[18] Troffaes M., 2004 *Decision making with imprecise probabilities: A short review.* In F. G. Cozman (Hrg), SIPTA Newsletter, Dezember 2004, 4-7.

[19] Weichselberger A., 1993, *Ein neuer nichtparametrischer Anpassungstest zur Beurteilung der Lage von Verteilungen*, Vandenhoeck & Ruprecht, Göttingen.

[20] Weichselberger K., 2001, *Elementare Grundbegriffe einer allgemeinen Wahrscheinlichkeitsrechnung I. Intervallwahrscheinlichkeit als umfassendes Konzept.* Physica, Heidelberg. Unter Mitarbeit von T. Augustin und A. Wallner.

[21] Wheeler J.C., 1974, *Rocky Mountain*, J.Math. 4, 287-296.

Nützliche Links

[22] `http://www.netlib.org/quadpack/`: `Quadpack` Algorithmen zur numerischen Integration.

[23] http://www.library.cornell.edu/nr/cbookcpdf.html: Online-Version der *Numerical Recipes* für C.

[24] http://www.netlib.org/lapack/: LAPACK-Paket mit (in R verwendeten) Funktionen zur Eigenwertzerlegung.

[25] http://lpsolve.sourceforge.net/5.5/: Mixed Integer Programming (MIP) Solver (lp_solve 5.5 solver) zur linearen Optimierung unter Nebenbedingungen.

[26] http://www.sipta.org/: Homepage der „Society for Imprecise Probability“.

Abbildungsverzeichnis

Anhang

R-Quellcode

Der folgende und weiterer in dieser Diplomarbeit verwendeter R-Code ist auf der beigefügten CD zu finden.

Funktionen `discretization` und `polynom` zum Berechnen der Gausschen Quadratur ohne klassische Gewichte

Hauptfunktion

Eingabe:

- `a`: linker Rand des Trägers
- `b`: rechter Rand des Trägers
- `N`: Anzahl der Stützstellen und Grad des Polynoms
- `w`: Gewichts- bzw. Dichtefunktion

Ausgabe:

- `nullstellen`: Stützstellen der Gausschen Quadratur
- `dichte`: Gewichte der Gausschen Quadratur

Benötigt:

Die Hilfsfunktion `polynom`.

```
> discretization <- function(a, b, N, w = as.function(
+     alist(x = , 1/(sqrt(2 *pi)) * exp((-1/2) * x^2)))) {
+     p <- c(0, as.function(alist(x = , 1)))
+     delta <- rep(0, N)
+     gamma <- rep(0, N)
```

```
+     for (i in 1:N) {
+         pol <- polynom(p[[i]], p[[i + 1]],
+             w = as.function(alist(x = , w(x))),
+             a = a, b = b)
+         p <- c(p, pol[[3]])
+         gamma[i] <- pol[[1]]
+         delta[i] <- pol[[2]]
+     }
+     J_oben1 <- diag(sqrt(gamma[2:N]))
+     J_oben2 <- cbind(rep(0, N - 1), J_oben1)
+     J_oben3 <- rbind(J_oben2, t(rep(0, N)))
+     J_unten2 <- cbind(J_oben1, rep(0, N - 1))
+     J_unten3 <- rbind(t(rep(0, (N))), J_unten2)
+     J <- diag(delta[1:N]) + J_oben3 + J_unten3
+     E <- eigen(J, symmetric = TRUE)
+     nullstellen <- E[[1]]
+     dichte <- E[[2]][1, ]^2
+     return(c(nullstellen, dichte))
+ }
```

Hilfsfunktion

Eingabe:

`p1`:	Polynom (Grad: N-2)
`p2`:	Polynom (Grad: N-1)
`a,b`:	Trägergrenzen
`w`:	Gewichtsfunktion

Ausgabe:

`gamma`:	γ_N^2
`delta`:	δ_N
`p_neu`:	Polynom (Grad: N)

```
> polynom <- function(p1, p2, a, b, w) {
+     skalarprodukt_zähler_delta <- as.function(alist(x = ,
+         x * p2(x)^2 * w(x)))
+     skalarprodukt_nenner_delta <- as.function(alist(x = ,
+         p2(x)^2 * w(x)))
+     delta <- as.numeric(integrate(skalarprodukt_zähler_delta,
+         a, b)[1])/as.numeric(integrate(skalarprodukt_nenner_delta,
+         a, b)[1])
+     if (mode(p1) == "numeric") {
+         gamma <- 0
+         p1 <- as.function(alist(x = , 0))
+     }
+     else {
+         skalarprodukt_nenner_gamma <- as.function(alist(x = ,
+             p1(x)^2 * w(x)))
+         skalarprodukt_zähler_gamma <- as.function(alist(x = ,
+             p2(x)^2 * w(x)))
+         gamma <- as.numeric(integrate(skalarprodukt_zähler_gamma,
+             a, b)[1])/as.numeric(integrate(
+             skalarprodukt_nenner_gamma,a, b)[1])
+     }
+     p_neu <- as.function(alist(x = , (x - delta) * p2(x) - gamma *
+     p1(x)))
+     return(c(gamma, delta, p_neu))
+ }
```

Funktion `gauss.quad.prob.inter` zur Berechnung der Gausschen Quadratur mit Hilfe einer Aufteilung des Trägers in Intervalle (Theorem 2)

Eingabe:

`N`:	Anzahl der Stützstellen in jedem Intervall und Grad des Polynoms
`M`:	Anzahl der Intervalle
`a,b`:	• Parameter der angegebenen Verteilung (normal: a=mean, b=sigma; beta: a=shape, b=scale; gamma: a=shape, b=scale; unif: a=a, b=b) oder • linker und rechter Rand der äquidistanten Partition
`w`:	Dichtefunktion
`angepasst`:	Partition wird an die Dichtefunktion angepasst
`typ`:	Um welche Verteilung handelt es sich (”normal”, ”beta”, ”gamma”, ”unif”)
`part`:	wird eine Partition des Trägers vorgegeben, ja oder nein?
`pa`:	Vektor mit dieser vorgegebenen Partition (Länge: M+1)

Ausgabe:

`nodes`:	Stützstellen der Gausschen Quadratur
`weights`:	Gewichte der Gausschen Quadratur
`intervals`:	die M Intervalle

Benötigt:

Die Funktionen `discretization` und `polynom`

```
> gauss.quad.prob.inter <- function(N, M, a, b, w = as.function(alist(x = ,
+     1/(sqrt(2 * pi)) * exp((-1/2) * x^2))), angepasst = FALSE,
+     typ = "normal", part = FALSE, pa = c(0)) {
+     if (angepasst == TRUE) {
+         pa <- seq(0, 1, 1/M)
```

```
+         if (typ == "normal") {
+             tr <- qnorm(pa, a, b)
+             if (tr[1] == -Inf) {
+                 tr[1] = tr[2] - 10 * b
+             }
+             if (tr[length(tr)] == Inf) {
+                 tr[length(tr)] = tr[(length(tr) - 1)] + 10 * b
+             }
+         }
+         if (typ == "beta") {
+             tr <- qbeta(pa, a, b)
+         }
+         if (typ == "gamma") {
+             if (a == 1) {
+                 pa[length(pa)] <- 0.999999
+             }
+             tr <- qgamma(pa, a, b)
+         }
+         if (typ == "unif") {
+             tr <- qunif(pa, a, b)
+         }
+     }
+     if (part == TRUE) {
+         tr <- pa
+         l <- length(pa)
+         if (l != M + 1)
+             stop("M und Länge des Trägers passen nicht zusammen")
+     }
+     if (angepasst == FALSE && part == FALSE) {
```

```
+         schritt <- (b - a)/M
+         tr <- seq(a, b, by = schritt)
+     }
+     y <- rep(0, (N * M))
+     p <- rep(0, (N * M))
+     for (i in 1:M) {
+         if (angepasst == TRUE) {
+             if (typ == "normal") {
+                 int = pnorm(tr[i + 1], mean = a, sd = b) -
+                       pnorm(tr[i], mean = a, sd = b)
+             }
+             else {
+                 if (typ == "beta") {
+                   int = pbeta(tr[i + 1], a, b) - pbeta(tr[i], a, b)
+                 }
+                 else {
+                   if (typ == "gamma") {
+                     int = pgamma(tr[i + 1], a, b) - pgamma(tr[i], a, b)
+                   }
+                   else {
+                     if (typ == "unif") {
+                       int = punif(tr[i + 1], a, b) - punif(tr[i], a, b)
+                     }
+                   }
+                 }
+             }
+         }
+         else {
+             int <- as.numeric(integrate(w, tr[i], tr[i + 1])[1])
```

```
+         }
+         gauss <- discretization(tr[i], tr[i + 1], N, w = as.function(
+                                 alist(x = , w(x)/int)))
+         y[(i * N):((i * N) - (N - 1))] <- gauss[1:N]
+         p[(i * N):((i * N) - (N - 1))] <- (gauss[(N + 1):(2 * N)] * int)
+     }
+     return(list(nodes = y, weights = p, intervals = tr))
+ }
```

Funktion bayes zur Berechnung des Erwartungswertes nach Bayes im Bernoulli-Experiment mit Beta-Priori

Eingabe:

N:	Grad des Polynoms
a,b:	Parameter der Beta-Priori
y:	Anzahl Erfolge im Bernoulli-Experiment
n:	Anzahl Versuche im Bernoulli-Experiment

Ausgabe:

`gauss`:	Erwartungswert nach Gauss
`int`:	Erwartungswert mit `integrate`
`True`:	„wahrer“ Wert: $\frac{y}{n}$
`analytical`: analytisch berechneter Wert	

Benötigt:

Die Funktion `gauss.quad.prob` aus dem Packet `statmod`.

```
> bayes <- function(N = 10, a = 1, b = 1, y, n) {
+     if (y > n)
+         stop("Anzahl Treffer größer als Anzahl Versuche")
+     gqp <- gauss.quad.prob(N, "beta", alpha = a, beta = b)
```

```
+     zähler <- sum(gqp$weights * gqp$nodes^(y + 1) *
+        (1 - gqp$nodes)^(n - y))
+     nenner <- sum(gqp$weights * gqp$nodes^(y) *
+        (1 - gqp$nodes)^(n - y))
+     EW_gauss <- zähler/nenner
+     try(f1 <- as.function(alist(x = , x^(y + a) * (1 - x)^(n -
+         y + b - 1) * gamma(a + b)/(gamma(a) * gamma(b)))))
+     try(f2 <- as.function(alist(x = , x^(y + a - 1) * (1 - x)^(n -
+         y + b - 1) * gamma(a + b)/(gamma(a) * gamma(b)))))
+     try(int1 <- integrate(f1, 0, 1))
+     try(int2 <- integrate(f2, 0, 1))
+     try(EW_int <- int1[[1]]/int2[[1]])
+     an = (y+a)/(n+b+a)
+     return(c(gauss = EW_gauss, int = EW_int, True = y/n,
+         analytical = an))
+ }
```

Funktion `linopt` zum Finden der optimalen Aktion mithilfe linearer Optimierung

Eingabe:

a:	μ bei Normalverteilung, l bei Gleichverteilung, α bei Gammaverteilung und Betaverteilung
b:	σ bei Normalverteilung, u, bei Gleichverteilung, β bei Gammaverteilung und Betaverteilung
N:	Grad des Polynoms
typ:	"normal", "uniform", "gamma", "beta"
Fu:	$(n \times 1)$-Matrix mit den Nutzenfunktionen der n Aktionen

Ausgabe:

`res`: optimale Aktion und dazugehöriger Erwartungsnutzen

Benötigt:

Die Funktion `gauss.quad.prob` aus dem Paket `statmod`, sowie die Funktion `lp` aus dem Paket `lpSolve`.

```
> linopt = function(N = 10, a = 0, b = 1, typ = "normal", Fu) {
+     gq = gauss.quad.prob(N, dist = typ, mu = a, sigma = b, l = a,
+         u = b, alpha = a, beta = b)
+     S = nrow(Fu)
+     A = matrix(rep(1, S), nrow = 1)
+     k = c(1)
+     for (j in 1:S) {
+         d1 = c(sum(Fu[[j, 1]](gq$nodes) * gq$weights))
+         if (j == 1) {
+             d = d1
+         }
+         else {
+             d = c(d, d1)
+         }
+     }
+     res1 = (lp(direction = "max", d, A, "=", k))$solution
+     res2 = (lp(direction = "max", d, A, "=", k))$objval
+     for (i in 1:S) {
+         if (res1[[i]] != 0)
+             res3 = i
+     }
+     res = list(res3,res2)
+     res
+ }
```

Funktionen `noncl.exp.value`, `eadmissible` und `maximin` zum Identifizieren E-admissibler und Γ-Maximin Aktionen bei stetigem Zustandsraum

Hauptfunktion

Eingabe:

`U`:	Matrix mit Nutzenfunktionen
`typ`:	Welcher Typ Verteilung soll diskretisiert werden?
`a,b`:	Parameter der angegebenen Verteilung (normal: a=mean, b=sigma; beta: a=shape1, b=shape2; gamma: a=shape, b=rate; unif: a=a, b=b)
`a_seq`:	Sequenz möglicher anderer Werte für Parameter a
`b_seq`:	Sequenz möglicher anderer Werte für Parameter b
`N`:	Grad Polynom
`theorem1`:	nach Theorem 1 oder 2 diskretisieren? (1: TRUE, 2: FALSE)
`M`:	Anzahl Intervalle, für Theorem 2 benötigt
`w`:	Dichtefunktion, für Theorem 2 benötigt

Ausgabe:

`eadmissible`:	Vektor mit Charakterisierung der Aktionen 1: e-admissibel, sonst: nicht e-admissibel
`maximin`:	Vektor mit Charakterisierung der Aktionen für Γ-Maximax Kriterium ($\rightarrow \lambda$)
`value`:	Wert der Γ-Maximin Aktion
`discrete`:	Träger der diskretisierten Verteilung
`lower`:	untere Grenzen der Wahrscheinlichkeiten
`upper`:	obere Grenzen der Wahrscheinlichkeiten

Benötigt:

Die Funktionen `gauss.quad.prob.inter`, `discretization`, sowie `polynom`. Außerdem die Hilfsfunktionen `eadmissible` und `maximin`.

```
> noncl.exp.value = function(U, typ = "normal", a, b, a_seq = 0,
+     b_seq = 0, N, theorem1 = T, M = 1, w = as.function(alist(x = ,
```

```
+        1/(sqrt(2 * pi)) * exp((-1/2) * x^2)))) {
+    if (theorem1 == T) {
+        gq = gauss.quad.prob(N, "normal", mu = a, sigma = b)
+    }
+    else {
+        gq = gauss.quad.prob.inter(N, M, a, b, w, angepasst = T,
+            typ)
+    }
+    w = gq$weights
+    x = gq$nodes
+    m = nrow(U)
+    N = N * M
+    l1 = length(a_seq)
+    l2 = length(b_seq)
+    if (l1 == 1)
+        a_seq = a
+    if (l2 == 1)
+        b_seq = b
+    A_sigma = matrix(rep(0, l1 * l2 * (N - 1) * N), ncol = (N))
+    z = 1
+    if (typ == "normal") {
+        for (i in a_seq) {
+            for (j in b_seq) {
+                for (k in (N - 1):1) {
+                  A_sigma[z + l1 * l2 * (N - k - 1), ] = (pnorm(x,
+                    mean = i, sd = j) - c(rep(0, N - k), pnorm(x[1:k],
+                    mean = i, sd = j)))
+                }
+                z = z + 1
```

```
+             }
+         }
+     }
+     if (typ == "beta") {
+         for (i in a_seq) {
+             for (j in b_seq) {
+                 for (k in (N - 1):1) {
+                   A_sigma[z + l1 * l2 * (N - k - 1), ] = (pbeta(x,
+                     shape1 = i, shape2 = j) - c(rep(0, N - k),
+                     pbeta(x[1:k], shape1 = i, shape2 = j)))
+                 }
+                 z = z + 1
+             }
+         }
+     }
+     if (typ == "gamma") {
+         for (i in a_seq) {
+             for (j in b_seq) {
+                 for (k in (N - 1):1) {
+                   A_sigma[z + l1 * l2 * (N - k - 1), ] = (pgamma(x,
+                     shape = i, rate = j) - c(rep(0, N - k), pgamma(x[1:k],
+                     shape = i, rate = j)))
+                 }
+                 z = z + 1
+             }
+         }
+     }
+     if (typ == "unif") {
+         for (i in a_seq) {
```

```
+           for (j in b_seq) {
+               for (k in (N - 1):1) {
+                 A_sigma[z + l1 * l2 * (N - k - 1), ] = (punif(x,
+                   min = i, max = j) - c(rep(0, N - k), punif(x[1:k],
+                   min = i, max = j)))
+               }
+               z = z + 1
+           }
+       }
+   }
+   pu = matrix(rep(0, (N - 1) * N), ncol = N)
+   po = matrix(rep(0, (N - 1) * N), ncol = N)
+   for (i in 1:(N - 1)) {
+       pu[i, i:N] = apply(A_sigma[(1 + l1 * l2 * (i - 1)):(l1 *
+           l2 * i), i:N], 2, "min")
+       po[i, i:N] = apply(A_sigma[(1 + l1 * l2 * (i - 1)):(l1 *
+           l2 * i), i:N], 2, "max")
+   }
+   Nu = matrix(rep(0, m * N), ncol = N)
+   I = seq(1:m)
+   for (i in I) {
+       Nu[i, ] = U[[i, 1]](x)
+   }
+   eadmis = rep(0, m)
+   for (i in I) {
+       hilf = eadmissible(Nu, x, aktion = i, pu, po, N)
+       eadmis[i] = hilf
+   }
+   minim = minimax(Nu, x, pu, po)
```

```
+     maxmin_aktion = minim[1:m]
+     maxmin_wert = minim[(m + 1)]
+     return(list(eadmissible = eadmis, maximin = maxmin_aktion,
+         value = maxmin_wert, discrete = x, lower = pu, upper = po))
+ }
```

Hilfsfunktion eadmissible

Eingabe:

Nu:	Matrix der Werte der Nutzenfunktionen der Aktionen (Dimension: $m \times n$)
aktion:	Welche Aktion aus Nu soll betrachtet werden?
x:	Stützstellen der Gauss-Quadratur
pu:	untere Grenzen der Wahrscheinlichkeiten
po:	obere Grenzen der Wahrscheinlichkeiten
N:	Grad Polynom

Ausgabe:

res:	1: Aktion ist e-admissibel, sonst: Aktion ist nicht e-admissibel

```
> eadmissible = function(Nu, aktion = 1, x, pu, po, N) {
+     m = nrow(Nu)
+     A = matrix(rep(0, N * (m - 1)), ncol = N)
+     if (aktion != 1) {
+         vertausch = diag(m)
+         vertausch[1, 1] = 0
+         vertausch[aktion, aktion] = 0
+         vertausch[1, aktion] = 1
+         vertausch[aktion, 1] = 1
+         Nu = vertausch %*% Nu
+     }
+     I = seq(1:(m - 1))
```

```
+     for (i in I) {
+         A[i, ] = Nu[1, ] - Nu[(i + 1), ]
+     }
+     A = rbind(A, rep(1, N))
+     A = rbind(A, diag(N))
+     A = rbind(A, diag(N))
+     for (i in 2:(N - 1)) {
+         for (j in N:i) {
+             A = rbind(A, c(rep(0, (N - j)), rep(1, i), rep(0,
+                 (j - i))))
+         }
+     }
+     for (i in 2:(N - 1)) {
+         for (j in N:i) {
+             A = rbind(A, c(rep(0, (N - j)), rep(1, i), rep(0,
+                 (j - i))))
+         }
+     }
+     d = c(rep(0, (m - 1)), 1, pu[1, ], po[1, ])
+     for (i in 2:(N - 1)) {
+         d2 = pu[i, i:N]
+         d = c(d, d2)
+     }
+     for (i in 2:(N - 1)) {
+         d2 = po[i, i:N]
+         d = c(d, d2)
+     }
+     k = rep(1, N)
+     zähler = sum(2:(N - 1))
```

```
+     res = (lp(direction = "max", k, A, c(rep(">=", (m - 1)),
+         "<=", rep(">=", N), rep("<=", N), rep(">=", zähler),
+         rep("<=", zähler)), d))$objval
+     return(res)
+ }
```

Hilfsfunktion `maximin`

Eingabe:

`Nu`:	Matrix der Werte der Nutzenfunktionen der Aktionen (Dimension: $m \times n$)
`x`:	Stützstellen der Gauss-Quadratur
`pu`:	untere Grenzen der Wahrscheinlichkeiten
`po`:	obere Grenzen der Wahrscheinlichkeiten

Ausgabe:

`minimax_aktion`:	Vektor mit Charakterisierung der Aktionen für Γ-Maximax Kriterium ($\rightarrow \lambda$)
`minimax_wert`:	Wert der Γ-Maximin Aktion

```
> minimax = function(Nu, x, pu, po) {
+     n = length(x)
+     m = nrow(Nu)
+     Fj = diag(n)
+     for (i in 2:n) {
+         for (j in 1:(n - i + 1)) {
+             L = matrix(c(rep(0, (j - 1)), rep(1, i), rep(0, (n -
+                 i - j + 1))), nrow = 1)
+             Fj = rbind(Fj, L)
+         }
+     }
+     l = (n + 1) * n/2
+     z = rep(1, n)
```

```
+     A = cbind(z, t(Fj), t(-Fj), t(-Nu))
+     e = matrix(c(rep(0, (1 + 2 * l)), rep(1, m)), nrow = 1)
+     A = rbind(A, e)
+     d = c(rep(0, n), 1)
+     I = 1:m
+     res_max = rep(0, m)
+     res_min = rep(0, m)
+     Bu = pu[1, ]
+     Bo = po[1, ]
+     for (i in 2:(n - 1)) {
+         Bu = c(Bu, pu[i, i:n])
+         Bo = c(Bo, po[i, i:n])
+     }
+     k = c(1, Bu, 0, -Bo, -1, rep(1, m))
+     res_min = lp(direction = "max", k, A, c(rep("<=", n), "="),
+         d)
+     minmax_aktion = res_min$solution[(2 * l + 2):(2 * l + 1 +
+         m)]
+     minmax_wert = res_min$objval
+     return(c(minmax_aktion, minmax_wert))
+ }
```

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe.

München, den 04. Oktober 2006

(Michael Obermeier)