Wissenschaftliche Berichte
ISSN 0947-8620

Globale Optimierung mit Ergebnisverifikation

– Eine Einführung und ein Überblick über neuere Entwicklungen –

Dietmar Ratz

Institut für Angewandte Mathematik
Universität Karlsruhe

Zusammenfassung

Ziel der globalen Optimierung ist es, unter einer im allgemeinen großen Zahl *lokaler* Optima das *globale* Optimum zu lokalisieren. Globale Optimierungsverfahren können dazu zwar auf lokale Verfahren zurückgreifen, benötigen aber zusätzlich globale Informationen. Ein exzellentes Hilfsmittel zur Ermittlung globaler Informationen ist die Intervallrechnung, da es beispielsweise eine einzige Intervall-Funktionsauswertung ermöglicht, Aussagen über die Lage der Funktionswerte für alle Punkte (im allgemeinen unendliche viele) innerhalb des Intervalls zu machen.

Neuere Intervallverfahren sind in der Lage, garantierte Einschließungen für alle Lösungen eines globalen Optimierungsproblems zu berechnen. Selbst Eindeutigkeitsnachweise können im Rahmen der numerischen Durchführung erbracht werden. In den meisten Fällen ist die Effizienz dieser sogenannten Verifikationsverfahren vergleichbar mit der von klassischen Verfahren. Zahlreiche Standardtestaufgaben aus der Literatur können sogar schneller gelöst werden als von herkömmlichen Verfahren.

Es wird zunächst eine Einführung in die Grundlagen und Prinzipien solcher Verifikationsverfahren zur globalen Optimierung gegeben. Im Anschluß wird ein Überblick über neuere Methoden gegeben und mittels zahlreicher Beispiele ihre Effizienz demonstriert.

1 Einleitung, Problemstellung und Motivation

Viele Probleme aus dem Bereich technisch-wissenschaftlicher Anwendungen können als globale nichtlineare Optimierungsprobleme formuliert werden. Im Gegensatz zur lokalen Optimierung, bei der eine optimale Lösung in der Nähe eines vorgegebenen Punktes gesucht wird, verlangt die globale Optimierung das Auffinden des „besten“ lokalen Optimums. Während das Gebiet der lokalen nichtlinearen Optimierung bereits seit vielen Jahren erforscht wird und entsprechend zahlreiche Theorien, numerische Verfahren und Veröffentlichungen vorliegen, steckt das Gebiet der globalen Optimierung noch in den Kinderschuhen. Wie wichtig eine zukünftige intensive Forschung auf diesem Gebiet sein wird unterstreicht die Tatsache, daß die meisten sogenannten *real-world*-Probleme von globalem und nicht etwa lokalem Charakter sind (vgl. z. B. [18] und [23]).

Wir beschränken uns in diesem Artikel auf die Problemstellung der Minimierung, die aber natürlich leicht (durch Negation der Zielfunktion) in das entsprechende Maximierungsproblem überführt werden kann.

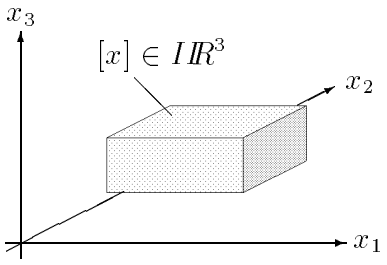


Abbildung 1: Ein dreidimensionaler Quader (Intervallvektor)

Gegeben seien also eine stetige Funktion $f : D \to I\!R$ und ein n-dimensionaler *Quader* $[x] \subseteq D \subseteq I\!R^n$ mit

$$[x] = ([x]_1, \ldots, [x]_n) \quad \text{und} \quad [x]_i = [\underline{x}_i, \overline{x}_i]$$

den wir auch *Box* oder *Intervallvektor* nennen (vgl. Abbildung 1). Gesucht sind das *globale Minimum*

$$f^* = \min_{x \in [x]} f(x)$$

und *alle globale Minimalstellen* in $[x]$, d. h. die Menge

$$X^* = \{x \in [x] \mid f(x) = f^*\}.$$

Im allgemeinen existiert eine große Zahl lokaler Minima, und ein globales Optimierungsverfahren kann zwar auf bekannte lokale Verfahren zurückgreifen, benötigt aber zusätzlich globale Informationen um garantierte Aussagen über das tatsächliche Erreichen des globalen Minimums machen zu können. Setzt man nur lokale Methoden ein, so muß nämlich am Ende des Verfahrens eine der lokalen Lösungen zur globalen Lösung gemacht werden, ohne zu wissen, ob die „richtige“ lokale Lösung auch tatsächlich unter allen gefundenen ist.

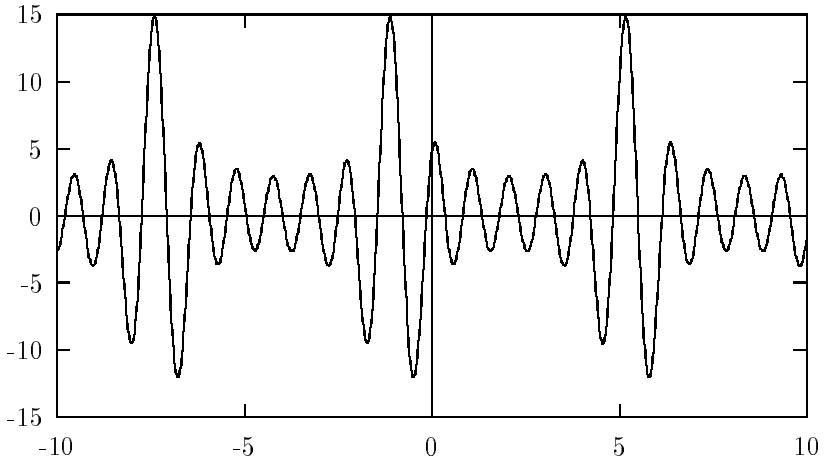


Abbildung 2: Funktion von Shubert

Klassische numerische Verfahren (approximative Verfahren) zur globalen Minimierung verwenden üblicherweise ein möglichst „dicht“ im Optimierungsgebiet verteiltes Raster

von Testpunkten, die als Startpunkte für lokale Iterationsverfahren (Abstiegsverfahren) dienen. Entsprechend problematisch gestaltet sich deshalb die Lösung von Problemen mit einer großen Zahl lokaler Minima. Wir wollen dies anhand der Funktion von Shubert (Abbildung 2) verdeutlichen, bei der viele Startpunkte und damit lokale Iterationen notwendig werden um die drei globalen Minimalstellen mittels approximativer Verfahren zu lokalisieren.

Abbildung 3 verdeutlicht eine weitere Problematik bei klassischen Optimierungsverfahren, wenn das globale Minimum in einem scharfen Peak liegt. Hier kann zum Beispiel der Fall eintreten, daß das eigentliche globale Minimum durch die (mit $\times$ gekennzeichnete) Iterationsfolge eines Abstiegsverfahrens quasi „übersprungen“ und das in der Nähe liegende lokale Minimum 0 als vermeintliches globales Minimum ausgegeben wird.

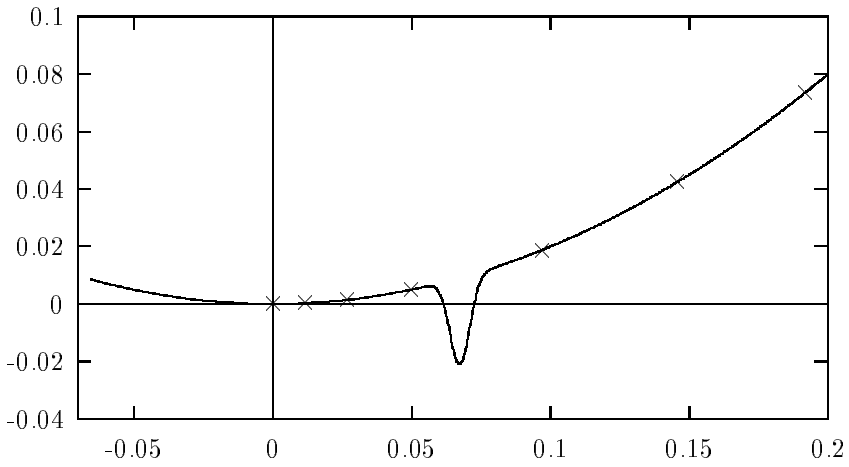


Abbildung 3: Funktion mit Peak

Die Notwendigkeit globaler Informationen, die sich somit offensichtlich stellt, wird durch den nachfolgenden Satz aus [23] nochmals unterstrichen.

Satz 1 *Ein globaler Optimierungsalgorithmus konvergiert genau dann gegen das globale Minimum einer stetigen Funktion f, wenn die Folge der durch den Algorithmus generierten Testpunkte überall dicht im kompakten Minimierungsbereich $[x]$ liegt.*

Beweis: Siehe [23].

Ein exzellentes Hilfsmittel zur Ermittlung globaler Informationen über Teilbereiche des Optimierungsgebietes ist die *Intervallrechnung*, denn Intervalle stellen ein ganzes Kontinuum dar, d. h. sie repräsentieren im allgemeinen unendlich viele Punkte, nämlich alle Punkte, die auf und zwischen den beiden Intervallgrenzen liegen. Unter Verwendung einer einzigen intervallarithmetischen Funktionsauswertung können beispielsweise Aussagen über die Lage der Funktionswerte für alle Punkte innerhalb eines Intervalls gemacht werden, denn die Intervallauswertung liefert eine Obermenge des Wertebereichs über dem Intervall. Somit ersetzt eine Intervallauswertung (unendlich) viele reelle Funktionsauswertungen. Bei der praktischen Durchführung auf dem Rechner werden auch alle auftretenden

Rundungsfehler mit erfaßt, so daß garantierte Fehlerschranken automatisch mitgeliefert werden.

Wenn wir mit diesem Hilfsmittel beispielsweise unsere bereits oben erwähnte Funktion mit dem Peak (Abbildung 3) behandeln wollen, so bringen bereits die drei Intervallauswertungen $[f_u]$, $[f_v]$ und $[f_w]$ von f über den Intervallen $[u]$, $[v]$ und $[w]$ wertvolle globale Informationen, die den approximativ berechneten Wert 0 in Frage stellen oder aber den zu untersuchenden Bereich verkleinern können. Schaut man sich die Funktionsintervalle am rechten Rand von Abbildung 4 an, so kann $[w]$ auf keinen Fall das globale Minimum enthalten, da die Funktionswerte in $[u]$ sämtlich kleiner sind. Außerdem deutet die Intervallauswertung über $[v]$ mit Funktionswerten kleiner als 0 darauf hin, daß dort das globale Minimum liegen könnte.

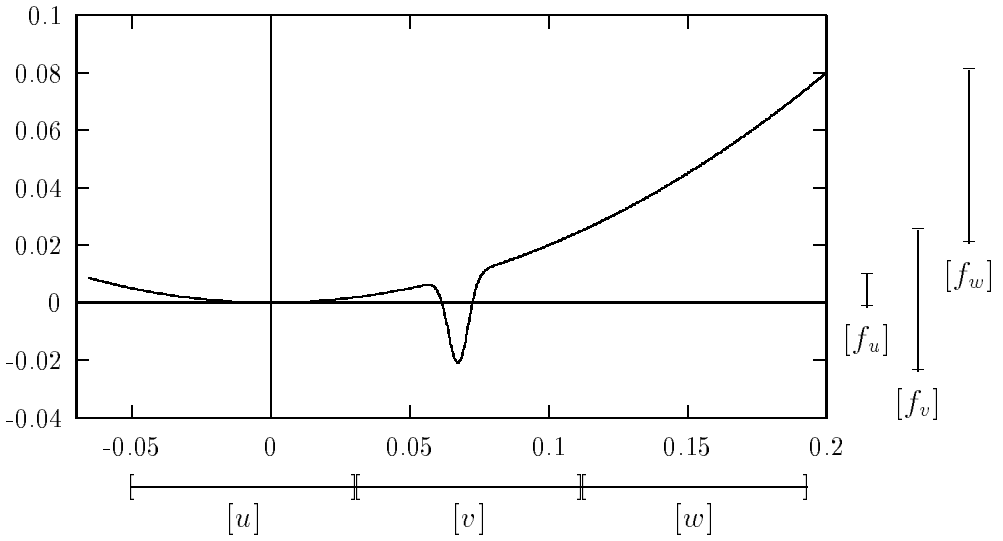


Abbildung 4: Intervallauswertungen für die Funktion mit Peak

2 Intervallarithmetik

Wir wollen nun einige Grundbegriffe der Intervallrechnung einführen und einige für das weitere Verständnis notwendige Eigenschaften erläutern. Eine ausführlichere Darstellung der Intervallrechnung findet sich zum Beispiel in [1] und in [6].

Die kompakte Menge

$$[a] := [\underline{a}, \overline{a}] := \{x \in I\!R \mid \underline{a} \leq x \leq \overline{a}\}$$

mit $\underline{a}, \overline{a} \in I\!R$ heißt reelles *Intervall*. Man nennt $\underline{a} = \inf[a]$ das *Infimum* oder die *Unterschranke* von $[a]$ und $\overline{a} = \sup[a]$ das *Supremum* oder die *Oberschranke* von $[a]$. Ein Intervall $[a]$ heißt *Punktintervall*, wenn gilt $\underline{a} = \overline{a}$. Die Menge aller reellen Intervalle bezeichnet man mit $I\!I\!R = \{[a] \mid \underline{a} \leq \overline{a}\}$. Die reellen Zahlen $m([a]) = \frac{1}{2}(\underline{a} + \overline{a})$ und $d([a]) = \overline{a} - \underline{a}$ heißen *Mittelpunkt* und *Durchmesser* von $[a]$.

Für zwei Intervalle $[a]$ und $[b]$ gilt

$$\begin{aligned} [a] \subseteq [b] &\iff \underline{b} \le \underline{a} \wedge \overline{a} \le \overline{b}, \\ [a] \overset{\circ}{\subset} [b] &\iff \underline{b} < \underline{a} \wedge \overline{a} < \overline{b}. \end{aligned}$$

Die *Verbandsoperationen* $\cap$ (Schnitt) und $\cup$ (Intervallhülle) sind für zwei Intervalle $[a]$ und $[b]$ definiert durch

$$\begin{aligned} [a] \cap [b] &:= [\max\{\underline{a}, \underline{b}\}, \min\{\overline{a}, \overline{b}\}], \\ [a] \underline{\cup} [b] &:= [\min\{\underline{a}, \underline{b}\}, \max\{\overline{a}, \overline{b}\}]. \end{aligned}$$

Dabei ist der Schnitt nur definiert, falls $\max\{\underline{a}, \underline{b}\} \le \min\{\overline{a}, \overline{b}\}$.

Die intervallarithmetischen Operationen $\circ \in \{+, -, \cdot, /\}$ werden definiert durch

$$[a] \circ [b] := \{a \circ b \mid a \in [a],\, b \in [b]\}.$$

Sie können auf Operationen mit den Intervallgrenzen zurückgeführt werden:

$$\begin{aligned} [a] \;+\; [b] &= [\underline{a} + \underline{b},\ \overline{a} + \overline{b}], \\ [a] \;-\; [b] &= [\underline{a} - \overline{b},\ \overline{a} - \underline{b}], \\ [a] \;\cdot\; [b] &= [\min\{\underline{a}\,\underline{b}, \underline{a}\,\overline{b}, \overline{a}\,\underline{b}, \overline{a}\,\overline{b}\},\ \max\{\underline{a}\,\underline{b}, \underline{a}\,\overline{b}, \overline{a}\,\underline{b}, \overline{a}\,\overline{b}\}], \\ [a] \;/\; [b] &= [a] \cdot [1/\overline{b},\ 1/\underline{b}] \quad \text{für } 0 \notin [b]. \end{aligned}$$

Beispiele:

$$\begin{aligned} [1,2] \;+\; [3,4] &= [4,6], \\ [1,2] \;-\; [1,2] &= [-1,1], \\ [-1,2] \;\cdot\; [4,6] &= [-6,12]. \end{aligned}$$

Das zweite Beispiel zeigt, daß für die Subtraktion im allgemeinen $[a] - [a] \neq [0,0]$ gilt. Addition und Multiplikation sind kommutativ und assoziativ, für Intervalle gilt jedoch nur die sogenannte *Subdistributivität*

$$[a] \cdot ([b] + [c]) \subseteq [a] \cdot [b] + [a] \cdot [c],$$

bei der für $[a], [b], [c] \in I\!I\!R$ nur in Ausnahmefällen Gleichheit gilt (z. B. wenn $\underline{b} \ge 0$ und $\underline{c} \ge 0$).

Beispiel:

$$\begin{aligned} &[-1,1] \cdot ([1,2] + [-2,-1]) = [-1,1] \cdot [-1,1] = [-1,1] \\ &[-1,1] \cdot [1,2] + [-1,1] \cdot [-2,-1] = [-2,2] + [-2,2] = [-4,4] \end{aligned}$$

Die zentrale Eigenschaft der Intervalloperationen ist die *Inklusionsisotonie* oder *Einschließungseigenschaft*

$$\begin{aligned} a \in [a] \wedge b \in [b] &\implies a \circ b \in [a] \circ [b] \\ [a] \subseteq [c] \wedge [b] \subseteq [d] &\implies [a] \circ [b] \subseteq [c] \circ [d] \end{aligned}$$

für alle $\circ \in \{+, -, \cdot, /\}$ mit $a, b \in I\!R$ und $[a], [b], [c], [d] \in I\!I\!R$.

In ähnlicher Form ist es möglich die Elementarfunktionen $\varphi : D \subset I\!R \to I\!R$ (wie z. B. sin, cos, exp etc.) für Intervalle zu definieren durch

$$\varphi([x]) := \{\varphi(x) \mid x \in [x]\}.$$

Auch diese können teilweise wiederum auf Operationen für die Intervallgrenzen zurückgeführt werden.

Beispiele:
$$\begin{aligned}
\varphi([x]) &= [\varphi(\underline{x}), \varphi(\overline{x})], \quad \varphi \in \{\arctan, \operatorname{arsinh}, \ln, \sinh\}, \\
\varphi([x]) &= [\varphi(\overline{x}), \varphi(\underline{x})], \quad \varphi \in \{\operatorname{arccot}, \operatorname{arcoth}\}, \\
\sqrt{[x]} &= [\sqrt{\underline{x}}, \sqrt{\overline{x}}], \\
\mathrm{e}^{[x]} &= [\mathrm{e}^{\underline{x}}, \mathrm{e}^{\overline{x}}].
\end{aligned}$$

Die zentrale Eigenschaft der Inklusionsisotonie lautet für die Elementarfunktionen

$$[a] \subseteq [b] \Longrightarrow \varphi([a]) \subseteq \varphi([b]).$$

Wir wollen drei wichtige Sachverhalte im Zusammenhang mit der Intervallarithmetik unterstreichen:

- Der Aufwand für eine Intervalloperation ist etwa doppelt so groß wie für die entsprechende reelle Operationen.
- Beim Rechnen mit Intervallen auf einer Rechenanlage ist zu beachten, daß alle Operationen mit *Außenrundung* ausgeführt werden müssen, um alle Rundungsfehler mit zu erfassen.
- Die reellen Intervalle müssen auf *Maschinenintervalle* abgebildet werden. Dazu werden die Intervallgrenzen durch gerichtete Rundungen (Außenrundung) auf Maschinenzahlen abgebildet. Man beachte jedoch, daß ein Intervall auf dem Rechner zwar Maschinenzahlen als Unter- und Obergrenzen besitzt, daß es aber trotzdem auch *alle reellen* Werte zwischen den Grenzen umfaßt! Das Maschinenintervall stellt somit auf dem Rechner ein Kontinuum dar.

Unter Einsatz der Intervallarithmetik ist es nun auf einfache Art und Weise möglich, den *Wertebereich* $W_f([x]) = \{f(x) \mid x \in [x]\}$ einer Funktion $f : \mathbb{R} \to \mathbb{R}$ über einem Intervall $[x]$ einzuschließen. Ersetzt man im entsprechenden Funktionsausdruck alle reellen Größen durch entsprechende Intervallgrößen und alle auftretenden Operationen durch die entsprechenden Intervalloperationen, so erhält man die *intervallmäßige Auswertung* $f([x])$ von f über $[x]$. Man nennt dies auch *natürliche Intervallerweiterung*.

Es gilt stets $W_f([x]) \subseteq f([x])$, d. h. die intervallmäßige Auswertung (auf dem Rechner einschließlich Außenrundung) liefert eine Obermenge (Einschließung) des Wertebereichs. Damit ist es prinzipiell möglich mit einer einzigen Intervallauswertung die Garantie dafür zu liefern, daß eine Funktion f keine Nullstelle bzw. keine negative Werte hat. Gilt nämlich $0 \notin f([x])$, so gilt auch $0 \notin W_f([x])$. Abbildung 5 demonstriert, daß auch eine Vielzahl von Gleitkommaauswertungen keine solche Garantie liefern kann, denn zwischen den Auswertestellen könnte immer noch ein „Ausreißer“ ins Negative vorliegen. Erst die Intervallauswertung, als Obermenge des Wertebereichs, kann dies ausschließen.

Ganz analog zum reellen können die notwendigen Operationen natürlich auch auf Intervallvektoren und -matrizen ausgedehnt werden, wobei Begriffe wie Mittelpunkt, Durchmesser, Vereinigung oder Schnitt jeweils komponentenweise interpretiert werden. Damit ist es möglich Einschließungsalgorithmen zur Lösung von mehrdimensionalen Problemen zu entwickeln wie z. B. für lineare und nichtlineare Gleichungssysteme oder globale Optimierungsprobleme (vgl. auch [6]).

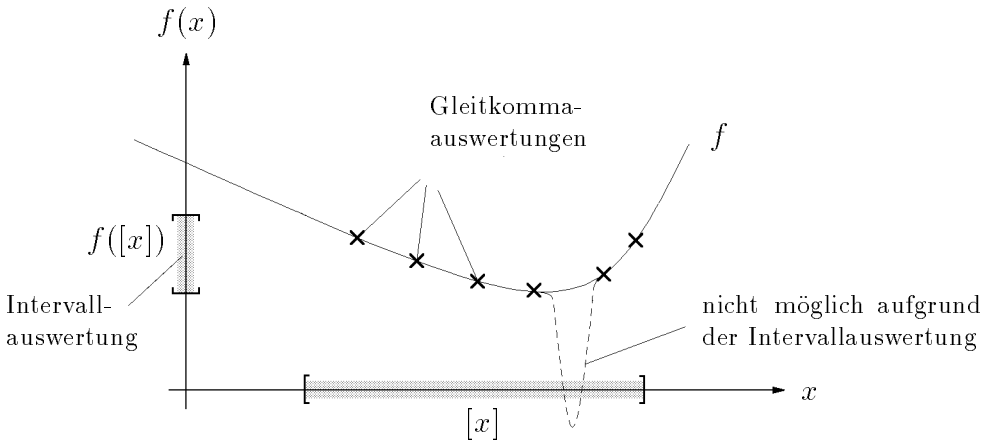


Abbildung 5: Intervallauswertung garantiert positive Werte von f in $[x]$

3 Globale Optimierung mit Ergebnisverifikation

Die Zielsetzung von Intervallverfahren zur Lösung des globalen Optimierungsproblems ist die Berechnung von *verifizierten Einschließungen* für das globale Minimum f^* *und* für die globalen Minimalstellen $x^* \in X^*$ innerhalb des Optimierungsbereichs $[x] \in I\!I\!R^n$.

Der zugrundeliegende Algorithmus basiert auf dem sogenannten *Intervall-Branch-and-Bound*-Prinzip:

> Zerlege den Startbereich $[x]$ in Teilbereiche $[y] \subset [x]$ (*branches*), bestimme garantierte Schranken (*bounds*) für f auf den Teilbereichen $[y]$ und eliminiere diejenigen Teilbereiche, die aufgrund dieser Schranken kein globales Minimum enthalten können.

Neben der natürlicherweise eingesetzten Intervallarithmetik muß das Verfahren mit einer möglichst optimalen Aufteilungsstrategie und Listenverwaltung für die Teilbereiche arbeiten. Weitere Hilfsmittel zur Beschleunigung des Algorithmus sind

- Lokale (approximative) Optimierungsmethoden,
- Cut-Off-Tests,
- Monotonietest (falls f eine C^1-Funktion),
- Konkavitätstest (falls f eine C^2-Funktion) und
- Intervall-Newton-Schritt (falls f eine C^2-Funktion).

Dabei ist es möglich, die benötigten Ableitungswerte mittels der sogenannten *automatischen Differentiation* zu berechnen. Diese ermöglicht es, bei der Berechnung eines Funktionswertes automatisch die Ableitungswerte mitzuberechnen. Dazu muß lediglich die Funktionsvorschrift bekannt sein, während die Ableitungsformeln nicht explizit angegeben werden müssen. Wir wollen auf diese Technik jedoch nicht weiter im Detail eingehen und verweisen für eine ausführlichere Darstellung auf [6].

Nach einer Beschreibung des Grund-Algorithmus, werden wir im folgenden auf die einzelnen Hilfsmittel eingehen, die in einem effizienten Verfahren zum Einsatz kommen. Auch dabei werden wir nicht bis ins kleinste Detail vorstoßen, um den Rahmen dieses Beitrages nicht zu sprengen.

3.1 Der Grund-Algorithmus

Wir wollen zunächst einmal schematisch die Arbeitsweise unseres Algorithmus beschreiben, der mit einer Liste L arbeitet, in der jeweils die noch zur Bearbeitung anstehenden Teilbereiche $[y]$ des Optimierungsbereichs $[x]$ abgespeichert werden. Außer dem jeweiligen Teilbereich $[y]$ ist zusätzlich die auf diesem Teilbereich berechnete garantierte Unterschranke

$$\underline{f_y} = \inf f([y])$$

für die Funktionswerte in $[y]$ mit abgespeichert. Zusätzlich verwendet der Algorithmus einen Wert $\widetilde{f}$, der eine *garantierte Oberschranke für den Wert des globalen Minimums* darstellt, d. h. $\widetilde{f} \geq f^*$. Der Algorithmus arbeitet nun mit L und $\widetilde{f}$ folgendermaßen:

Startphase:

- Die Arbeitsliste L wird mit dem Startbereich $[x]$ initialisiert:

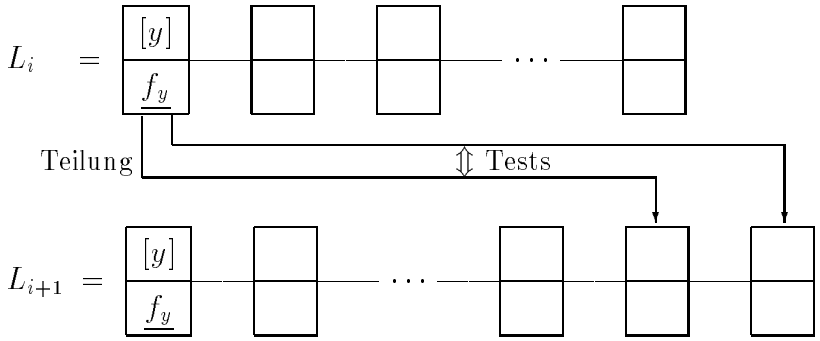
  $$L = L_0 := \boxed{\begin{array}{|c|}\hline [x] \\ \hline \underline{f_x} \\ \hline \end{array}}, \text{ wobei } [\underline{f_x}, \overline{f_x}] = [f_x] = f([x])$$

- Die garantierte Oberschranke $\widetilde{f} \geq f^*$ wird durch eine Intervallauswertung mit Punktargument initialisiert:

  Wähle $c \in [x]$ (z. B. den Mittelpunkt), berechne $[f_c] := f_{[]}(c)$ intervallmäßig und setze $\widetilde{f} := \overline{f_c}$.

Iteration:

- In der $(i+1)$-ten Iteration wird der erste Bereich $[y]$ in der Liste L_i (aus dem i-ten Iterationsschritt) aus der Liste entfernt und in zwei Teile aufgeteilt. Falls durch entsprechende Tests die Existenz einer globalen Minimalstelle in diesen Teilbereichen nicht ausgeschlossen werden kann, so werden die Teilbereiche an die Liste angehängt.



- Im nun neuen ersten Element der Liste L_{i+1} wird ein neues $c := m([y])$ gewählt und $[f_c] := f_{[]}(c)$ intervallmäßig berechnet. Dann wird ein Update des Wertes $\widetilde{f}$ durch die Bestimmung von $\widetilde{f} := \min\{\widetilde{f},\ \overline{f_c}\ \}$ durchgeführt.
- Mit dem neuen Wert $\widetilde{f}$ kann nun möglicherweise ein sogenannter *Cut-Off-Test* durchgeführt werden (s. u.).

Terminierung:

- Durch eine Genauigkeitsforderung (z. B. in Form einer Bedingung für die Durchmesser der Intervalle $[y]$ oder $f([y])$) kann der Algorithmus abgebrochen werden.

  Ergebnis: Für das globale Minimum und die globalen Minimalstellen ergibt sich nach dem Ende des Algorithmus, daß

$$f^* \in [\min\{\underline{f_y} \in L\},\ \widetilde{f}], \quad \text{und} \quad X^* \subseteq \bigcup_{[y]\in L} [y].$$

Zu jedem Zeitpunkt während der Durchführung des Algorithmus stellt der kleinste $\underline{f_y}$-Wert in der Liste stets die beste bekannte *garantierte Unterschranke* und $\widetilde{f}$ die beste bekannte *garantierte Oberschranke* für das globale Minimum f^* dar. Somit kann auch die Distanz beider Werte als mögliches Abbruchkriterium für den Algorithmus verwendet werden.

3.2 Der Cut-Off-Test

Mit der garantierten Oberschranke $\widetilde{f}$ für das Minimum f^* ist es möglich, alle Teilboxen $[y]$ aus der Liste zu entfernen, die die Beziehung

$$f^* \leq \widetilde{f} < \underline{f_y}$$

erfüllen, da $\underline{f_y}$ ja jeweils eine Unterschranke für die tatsächlichen Funktionswerte von f auf der Teilbox $[y]$ darstellt. Abbildung 6 verdeutlicht dies an einem Beispiel. Die schraffierten Rechtecke stellen dabei jeweils die Intervallauswertungen von f über dem entsprechenden

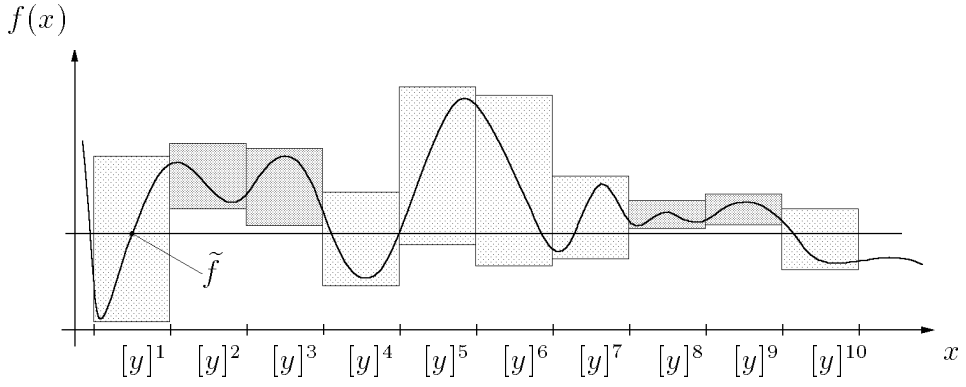


Abbildung 6: Der Cut-Off-Test

Teil der x-Achse dar. Mit der Höhenlinie auf dem Niveau von $\tilde{f}$ sieht man sofort, daß die dunkel schraffierten Gebiete kein globales Minimum enthalten können.

An diesem Beispiel wird auch deutlich, daß dieser Cut-Off-Test um so besser (schneller) funktioniert, je besser der Wert $\tilde{f}$ ist. Somit können gerade hier lokale (approximative) Verfahren (Abstiegsverfahren) zur Verbesserung des Punktes c und damit der garantierten Oberschranke $\tilde{f}$ zum Einsatz kommen. Würde nämlich im Beispiel (Abbildung 6) der Wert von $\tilde{f}$ in der Nähe des linken Randes von $[y]^1$ bestimmt werden, so könnte der Cut-Off-Test sogar alle anderen Teilintervalle $[y]^k$ mit $k = 2, \ldots, 10$ aus der Liste entfernen.

Diese Tatsache verdeutlicht die Vorgehensweise der Verifikationsverfahren zur globalen Optimierung, nämlich die *Ausnutzung der jeweiligen Vorteile beim wechselseitigen Einsatz von Gleitkomma- und Intervallrechnung*. Der Wert c kann approximativ „verbessert" werden, während der garantierte Wert $\tilde{f}$ durch eine Intervallauswertung bestimmt werden muß.

3.3 Algorithmische Darstellung

Wir wollen nun noch eine vereinfachte algorithmische Beschreibung unseres grundlegenden Verfahrens geben, um die prinzipielle Vorgehensweise zu verdeutlichen. Bei gegebener Arbeitsliste L, Listenelementen $([y], \underline{f_y})$, und zu optimierender Funktion $f : \mathbb{R}^n \to \mathbb{R}$, verwenden wir dabei die folgenden Notationen:

Notation	Bedeutung
$L := \{\,\}$	Initialisierung von L als leere Liste
$L := L + ([y], \underline{f_y})$	Einhängen von $([y], \underline{f_y})$ in L
$([y], \underline{f_y}) :=$ PopHead (L)	Aushängen von $([y], \underline{f_y})$ mit kleinstem $\underline{f_y}$ aus L
$\underline{f_y}$	Unterschranke der Intervallauswertung $[f_y] := f([y])$
$f_{[]}(c)$	Intervallauswertung von f im Punkt c

Algorithmus GlobalOptimize $(f, [x], \varepsilon, L_{\text{res}}, [f^*])$

1. $[f_c] := f_{[]}(m([x]))$; $\quad \tilde{f} := \overline{f_c}$;
2. $[y] := [x]$; $\quad L := \{\,\}$; $\quad L_{\text{res}} := \{\,\}$;
3. **repeat**
   (a) $k :=$ OptimalComponent $([y])$; $\quad$ Bisect $([y], k, [u]^1, [u]^2)$;
   (b) **for** $i := 1$ **to** 2 **do**
      i. $[f_u] := f([u]^i)$;
      ii. **if** $\tilde{f} < \underline{f_u}$ **then next**$_i$;
      iii. Apply additional tests and methods;
      iv. $L := L + ([u]^i, \underline{f_u})$; $\quad \{$Store $[u]^i\}$
   (c) *Bisect* := *false*;
   (d) **while** $(L \neq \{\,\})$ **and** (**not** *Bisect*) **do**
      i. $([y], \underline{f_y}) :=$ PopHead (L); $\quad c := m([y])$;

ii. Apply approximate method to improve c;
iii. $[f_c] := f_{[]}(c)$; $\widetilde{f} := \min\{\widetilde{f}, \overline{f_c}\}$; CutOffTest $(L, \widetilde{f})$;
iv. **if** $(d(f([y])) < \varepsilon)$ **or** $(d([y]) < \varepsilon)$ **then**
$L_{\text{res}} := L_{\text{res}} + ([y], \underline{f_y})$;
else $Bisect := true$;

until (**not** $Bisect$);

4. $[f^*] := [\min\{\underline{f_y} \in L\},\ \widetilde{f}]$;
5. **return** L_{res}, $[f^*]$;

Als Eingabedaten erhält GlobalOptimize die Funktion f, die Start-Box $[x]$ und einen (oder auch mehrere) Genauigkeitsparameter ε. In Schritt 1 wird zunächst durch eine intervallmäßige Mittelpunktsauswertung eine erste garantierte Oberschranke $\widetilde{f}$ für f^* bestimmt. Danach wird die aktuelle Box $[y]$ mit der Startbox $[x]$ und die Arbeitsliste L sowie die Ergebnisliste L_{res} jeweils mit einer leeren Liste initialisiert.

Schritt 3 stellt die eigentliche Iteration dar, in der in Schritt (a) jeweils eine Bisektion der aktuellen Box $[y]$ bezüglich einer möglichst optimal gewählten Koordinate k durchgeführt wird (Details zu dieser Richtungswahl finden sich z. B. in [20]). Danach werden die beiden Hälften $[u]^1$ und $[u]^2$ daraufhin untersucht, ob sie eine globale Minimalstelle enthalten können oder nicht. In Schritt (b)iii können dabei die in den nachfolgenden Abschnitten beschriebenen Methoden zum Einsatz kommen. Kann die Teilbox nicht eliminiert werden, so wird sie in Schritt (b)iv an die Liste L angehängt.

In Schritt 3(d) wird jeweils eine neue aktuelle Box $[y]$ aus der Liste ausgehängt und ausgehend von deren Mittelpunkt eine lokale approximative Methode zur Verbesserung des Wertes c und damit der Oberschranke $\widetilde{f}$ angewendet. Mit diesem Wert wird dann der Cut-Off-Test durchgeführt. Wenn die aktuelle Box bereits einer bestimmten Abbruchbedingung bezüglich des Durchmessers genügt, dann wird diese in die Ergebnisliste L_{res} eingehängt, andernfalls wird sie weiter halbiert.

In Schritt 4 wird noch die bestmögliche Einschließung $[f^*]$ für den Wert des globalen Minimums bestimmt, und zuletzt werden L_{res} und $[f^*]$ als Ergebnisse des Algorithmus zurückgegeben.

In den folgenden Abschnitten wollen wir nun noch auf die weiteren Hilfsmittel eingehen, die in Schritt 3(b)iii zum Einsatz kommen können, sofern die zu behandelnde Funktion f die jeweils erforderlichen Differenzierbarkeitsvoraussetzungen erfüllt.

3.4 Der Monotonietest

Ist f streng monoton in einer Box $[y] \subset [x]$, so kann diese aus der Liste gelöscht werden, denn $[y]$ kann in diesem Fall keinen stationären Punkt und somit auch kein lokales oder gar globales Minimum enthalten. Die Ausnahme bilden natürlich Randminima auf dem Rand des ursprünglichen Optimierungsgebietes $[x]$, die keine stationäre Punkte sind.

Erfüllt also die intervallmäßige Auswertung des Gradienten $[g] = \boldsymbol{\nabla} f([y])$ die Bedingung

$$0 \notin [g]_i \quad \text{für ein } i = 1, \ldots, n,$$

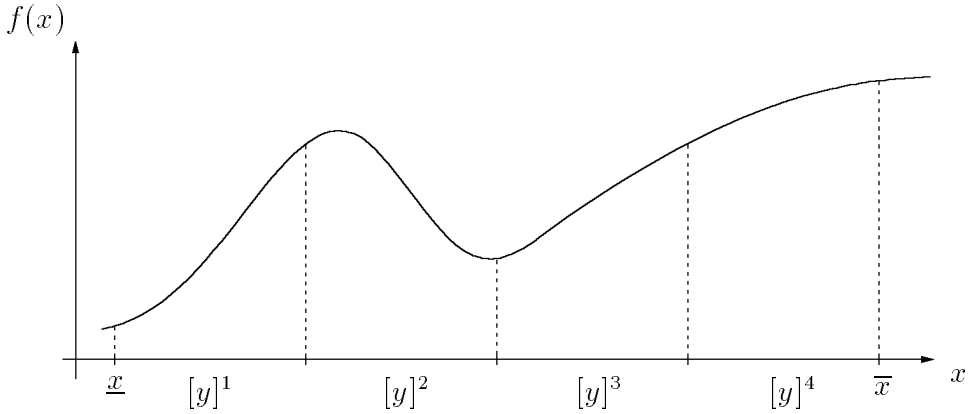


Abbildung 7: Zum Monotonietest

dann besitzt der Gradient keine Nullstelle in $[y]$, und f ist streng monoton in $[y]$ bezüglich der i-ten Koordinate. Die Box $[y]$ kann also gelöscht werden. Enthält $[y]$ allerdings Teile des Randes von $[x]$, so dürfen diese (zumindest vom Monotonietest) je nach Monotonieeigenschaft *nicht* gelöscht werden.

In dem in Abbildung 7 skizzierten Fall mit $n = 1$ und für vier Teilintervalle kann der Monotonietest $[y]^1$ auf den Randpunkt $\underline{x}$ reduzieren, die Box $[y]^2$ bleibt unverändert, und $[y]^3$ kann gelöscht werden. Da f in $[y]^4$ monoton *steigend* ist, kann das ganze Intervall gelöscht werden, da ja ein Minimum gesucht wird.

In diesem Beispiel reduziert der Monotonietest die ursprüngliche Liste von vier Intervallen $[y]^1$, $[y]^2$, $[y]^3$ und $[y]^4$ auf zwei Intervalle $[\underline{x}]$ und $[y]^2$. Hier wäre sogar der Fall gegeben, daß ein anschließender Cut-Off-Test mit einer Funktionsauswertung in $[\underline{x}]$ außerdem noch $[y]^2$ eliminieren würde und damit die eindeutige Lösung $x^* = \underline{x}$ zum Preis von nur vier Intervallauswertungen des Gradienten und zwei Intervallauswertungen von f bestimmt wäre.

4 Der Konkavitätstest

Dieser Test prüft eigentlich die „Nicht-Konvexität“, und er erhielt seinen Namen in der Literatur wohl zur Vereinfachung der Sprechweise. Er wird verwendet, um festzustellen ob die zu minimierende Funktion f auf einer Teilbox $[y] \subset [x]$ *nicht konvex* ist. Ist dies der Fall, so braucht $[y]$ nicht mehr weiter auf eine Minimalstelle untersucht zu werden, außer wenn $[y]$ Teile des Randes von $[x]$ enthält. In diesem Fall kann $[y]$, wie auch im Monotonietest, zumindest auf den Rand, der möglicherweise eine globale Minimalstelle enthält, reduziert werden.

Eine Funktion f ist konvex in $[y]$, wenn ihre Hessematrix überall in $[y]$ positiv semidefinit ist (vgl. Satz 1.6.3 in [24]). Eine notwendige Bedingung für die positive Semidefinitheit ist, daß alle Diagonalelemente der Hessematrix nichtnegativ sind (vgl. Satz 1.1.2 in [20]). Gilt nun für alle Hessematrizen in $[y]$, daß *ein* Diagonalelement kleiner als 0 ist, so kann die Hessematrix *nicht* positiv semidefinit sein.

Erfüllt also die intervallmäßige Auswertung der Hessematrix $[H] = \nabla^2 f([y])$ die Be-

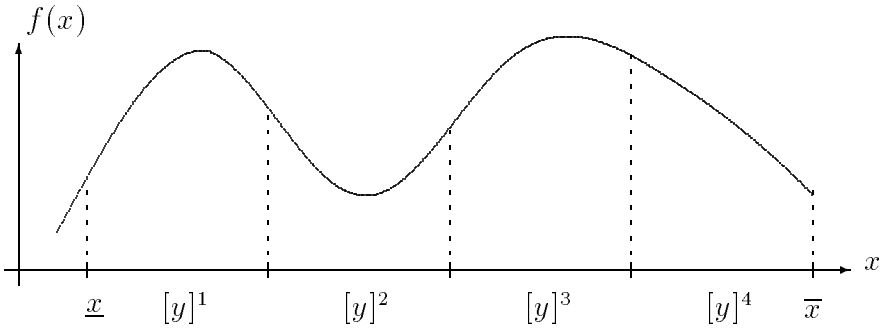


Abbildung 8: Zum Konkavitätstest

dingung

$$\overline{H}_{ii} < 0 \quad \text{für ein } i = 1, \ldots, n,$$

dann ist $H_{ii} < 0$ für $H = \nabla^2 f(y)$ und für alle $y \in [y]$. Somit kann $[y]$ gelöscht werden.

Abbildung 8 skizziert den eindimensionalen Fall für vier Teilintervalle, in dem der Konkavitätstest $[y]^1$ auf den Randpunkt $\underline{x}$ reduzieren, $[y]^3$ löschen und $[y]^4$ auf den Randpunkt $\overline{x}$ reduzieren kann. Die Box $[y]^2$ bleibt erhalten.

4.1 Intervall-Newton-(Gauß-Seidel)-Schritt

Ein weiteres Hilfsmittel zur Elimination bzw. zur Verkleinerung einer Teil-Box $[y]$ ist die Anwendung eines einzelnen Schrittes aus der Intervall-Newton- bzw. Gauß-Seidel-Iteration zur Lösung des nichtlinearen Gleichungssystems

$$\nabla f(y) = 0, \quad y \in [y].$$

Dabei wird eine a-priori Einschließung $[y]$ der Nullstelle des Gradienten durch Auflösen des Intervallgleichungssystems

$$[A](c - x) = b$$

mit $[A] \in I\!I\!R^{n \times n}$ und $b \in I\!I\!R^n$ verbessert. $[A]$ und b sind dabei gegeben durch

$$[A] := R \cdot \nabla^2 f([y]) \quad \text{und} \quad b := R \cdot \nabla f(c),$$

wobei $R \approx (m(\nabla^2 f([y])))^{-1}$ und $c := m([y])$.

Die neue (verbesserte) Einschließung $N'_{\mathrm{GS}}([y])$ berechnet sich gemäß

$$\left.\begin{aligned} [z] &:= [y] \\ [z]_i &:= \Big(c_i - \Big(b_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} [A]_{ij} \cdot ([z]_j - c_j) \Big) \Big/ [A]_{ii} \Big) \cap [z]_i \\ & \qquad i = 1, \ldots, n \\ N'_{\mathrm{GS}}([y]) &:= [z] \end{aligned}\right\}.$$

Bei der Division durch $[A]_{ii}$ tritt hier das zunächst formale Problem auf, daß der Nenner die Null enthalten könnte. Aus diesem Grund bedient man sich im Intervall-Gauß-Seidel-Schritt der sogenannten *erweiterten Intervallarithmetik* (vgl. [6], [9]), die eine Division

durch Intervalle, die die Null enthalten, erlaubt. Wir wollen dies hier nicht weiter vertiefen und verweisen für eine detaillierte Beschreibung des Intervall-Gauß-Seidel-Schrittes auf [1], [6, Kapitel 13,14] und [20, Abschnitt 2.5].

Im Rahmen unseres globalen Optimierungsverfahrens wäre eine vollständige Intervall-Newton-Iteration zwar möglich, aus „Kostengründen" wird dies aber nicht praktiziert. Der einzelne Newton-Schritt ist aufgrund der notwendigen Hessematrix-Auswertung relativ teuer, während die anderen Tests, die möglicherweise eine Box löschen bevor ein weiterer Newton-Schritt notwendig wird, wesentlich billiger sind. Außerdem behandelt der Newton-Schritt das Nullstellenproblem für den Gradienten und zielt damit auf die Berechnung eines stationären Punktes ab, der nicht notwendigerweise ein Minimalpunkt sein muß.

Die Anwendung des Intervall-Gauß-Seidel-Schrittes innerhalb unseres Verfahrens kann nun folgende Ergebnisse liefern:

- Die Ergebnisbox $N'_{\mathrm{GS}}([y])$ ist leer, d. h. $[y]$ enthält keinen stationären Punkt.
- Die Ergebnisbox $N'_{\mathrm{GS}}([y])$ ist signifikant verkleinert worden.
- Ergebnisbox $N'_{\mathrm{GS}}([y])$ ist insgesamt verkleinert und in Teilboxen aufgespalten worden.

Wir wollen dies im eindimensionalen Fall kurz graphisch erläutern.

Ähnlich wie das klassische Newton-Verfahren kann auch das Intervall-Newton-Verfahren geometrisch dadurch interpretiert werden, daß in jedem Iterationsschritt an der Stelle $c = m([y])$ zwei Geraden an den Graphen der Funktion f angelegt werden. Es handelt sich dabei um die Geraden mit der Steigung $\underline{f'([y])}$ bzw. $\overline{f'([y])}$, also mit der kleinsten bzw. größten Steigung von f im Intervall $[y]$. Ihre Schnittpunkte mit der x-Achse entsprechen den Intervallgrenzen der neuen Iterierten, die nachfolgend noch mit der alten Iterierten $[y]$ geschnitten wird. Anhand von zwei Skizzen wollen wir diese geometrische Interpretation für den Fall $0 \notin f'([y])$ und für den Fall $0 \in f'([y])$ veranschaulichen.

In Abbildung 9 ist für $[y] = [\underline{y}, \overline{y}]$ der Fall $0 \notin f'([y])$ skizziert. Die Gerade mit der kleinsten Steigung schneidet die x-Achse in λ, die Gerade mit der größten Steigung schneidet sie in ρ. In einem Newton-Schritt wird also zunächst das Intervall

$$[w] := c - \frac{f(c)}{f'([y])} = [\lambda, \rho]$$

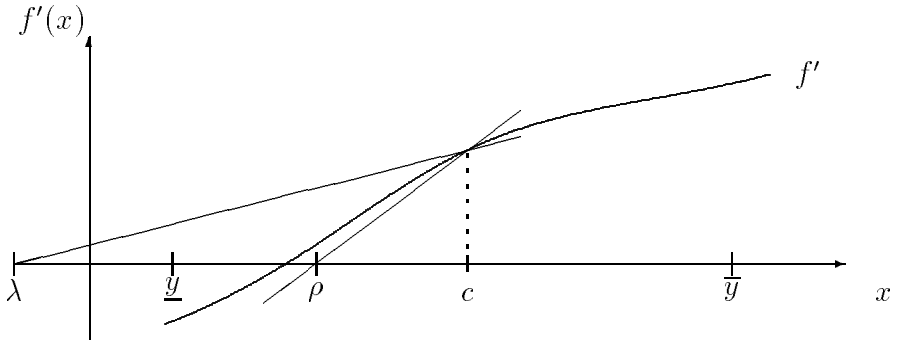


Abbildung 9: Intervall-Newton-Schritt mit $0 \notin f'([y])$

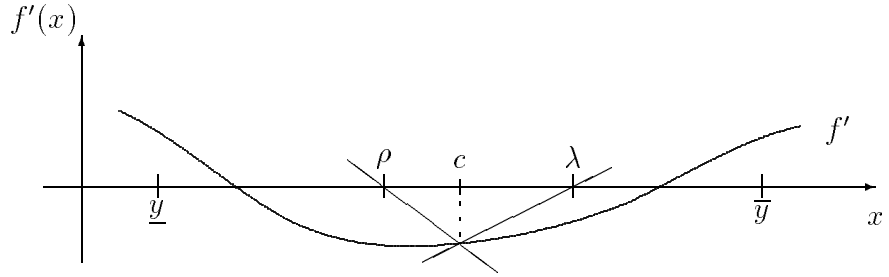


Abbildung 10: Intervall-Newton-Schritt mit $0 \in f'([y])$

berechnet, das in der Skizze links vom Mittelpunkt liegt. Die im Anschluß daran erfolgende Schnittbildung liefert somit

$$N'([y]) := [w] \cap [y] = [\lambda, \rho] \cap [\underline{y}, \overline{y}] = [\underline{y}, \rho],$$

als neue Iterierte und somit verbesserte Einschließung der Nullstelle von f'.

In Abbildung 10 ist für $[y] = [\underline{y}, \overline{y}]$ der Fall $0 \in f'([y])$ skizziert. Die Gerade mit der kleinsten Steigung schneidet die x-Achse in ρ, die Gerade mit der größten Steigung schneidet sie in λ. In einem (erweiterten) Newton-Schritt wird also zunächst das erweiterte Intervall

$$[w] := c - \frac{f(c)}{f'([y])} = (-\infty, \rho] \cup [\lambda, \infty)$$

berechnet und die anschließende Schnittbildung liefert

$$N'([y]) := [w] \cap [y] = ((-\infty, \rho] \cup [\lambda, \infty)) \cap [\underline{y}, \overline{y}] = [\underline{y}, \rho] \cup [\lambda, \overline{y}],$$

also die Vereinigung zweier Intervalle, die nach wie vor die Nullstellen von f' in $[y]$ enthält.

4.2 Eindeutigkeitsaussagen

Unter Anwendung von speziellen Fixpunktsätzen, deren Voraussetzungen mittels Intervallrechnung *auf dem Rechner überprüft werden können*, ist es möglich, daß das Verfahren selbst nachweist, daß eine Teilbox $[y]$ in der Ergebnisliste L_{res} eine (lokal in $[y]$) eindeutige Minimalstelle enthält. Dies ist möglich durch den Nachweis eines eindeutigen stationären Punktes in $[y]$, d. h. den Nachweis von Existenz und Eindeutigkeit einer Nullstelle von $\boldsymbol{\nabla} f$ in $[y]$ zusammen mit dem Nachweis, daß f in $[y]$ streng konvex ist, d. h. durch den Nachweis der positiven Definitheit aller $\boldsymbol{\nabla}^2 f$ in $[y]$.

Aufgrund der Eigenschaften des Intervall-Gauß-Seidel-Schrittes läßt sich der erste Punkt recht leicht überprüfen. Wir fassen diese im nachfolgenden Satz zusammen.

Satz 2 *Sei $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ eine zweimal stetig differenzierbare Funktion, und sei $[y] \in I\mathbb{R}^n$ mit $[y] \subseteq D$. Dann besitzt $N'_{\text{GS}}([y])$ die folgenden Eigenschaften:*

1. *Jede Nullstelle x^* in $[y]$ von $\boldsymbol{\nabla} f$ liegt auch in $N'_{\text{GS}}([y])$.*
2. *Ist $N'_{\text{GS}}([y]) = \emptyset$, dann existiert keine Nullstelle von $\boldsymbol{\nabla} f$ in $[y]$.*
3. *Gilt $N'_{\text{GS}}([y]) \overset{\circ}{\subset} [y]$, dann existiert eine eindeutige Nullstelle von $\boldsymbol{\nabla} f$ in $[y]$ und damit auch in $N'_{\text{GS}}([y])$.*

Beweis: Siehe z. B. [9].

Somit muß das Verfahren lediglich überprüfen, ob $N'_{\mathrm{GS}}([y])$ echt im Inneren von $[y]$ liegt. Mit einer ähnlichen Inklusionsbedingung läßt sich auch die positive Definitheit aller Hessematrizen in $[y]$ nachweisen. Wir wollen jedoch auch dies, um den Rahmen dieses Beitrags nicht zu sprengen, hier nicht weiter vertiefen und verweisen auf [6, Kapitel 14] und [20, Abschnitt 2.7].

5 Historie der Methoden und Varianten

Um dem interessierten Leser bzw. der interessierten Leserin die Möglichkeit zu geben, sich tiefer in die vorgestellte Methodik einzuarbeiten, geben wir nun einen kleinen historischen Überblick. Darin führen wir die wichtigsten Stationen der Entstehung verschiedener Varianten von Intervallverfahren zur globalen Optimierung einschließlich entsprechender Literaturreferenzen auf. Wir erheben jedoch keinen Anspruch auf Vollständigkeit.

1966: Durch das Buch von Moore [17] wird der Grundstock zur Verwendung der Intervallarithmetik für die globale Optimierung gelegt.

1974: Skelboe kombiniert in [22] eine Branch-and-Bound-Strategie mit Moores Intervallverfahren und entwickelt ein erstes ableitungsfreies Verfahren, das darauf abzielt, Einschließungen für f^* zu berechnen.

1979: Ichida und Fujii [10] entwickeln eine Modifikation des Skelboe-Verfahrens unter Einbeziehung des Mittelpunktstests und des Intervall-Newton-Verfahrens. Auch diese zielt darauf ab, nur Einschließungen für f^* zu berechnen.

1979/80: Eines der wichtigsten Intervall-Verfahren, veröffentlicht Hansen 1979 für den eindimensionalen Fall [7] und 1980 für den mehrdimensionalen Fall [8]. Es integriert die Mittelpunkts-, Monotonie- und Konkavitätstests sowie ein spezialisiertes Intervall-Newton-Verfahren mit dem Ziel, Einschließungen für f^* *und* für X^* zu berechnen.

1988: Ratschek und Rokne fassen in [18] die obigen Verfahren zusammen und führen umfangreiche Konvergenzuntersuchungen und Vergleiche durch.

1988/89/90: In [2], [3] und [4] beschreibt Csendes ein Verfahren mit Cut-Off-Test und Monotonietest speziell für Parameterschätzungsaufgaben im Bereich von Fertigungstoleranzen. Auch hier gilt das Interesse vornehmlich f^*.

1991/92: Jansson und Knüppel stellen in [11] und [12] ein Intervall-Branch-and-Bound-Verfahren vor, das ohne Ableitungen auskommt und unter intensiver Nutzung lokaler approximativer Verfahren sehr schnell gute Näherungen und garantierte Schranken für f^* liefert.

1991/92/93/94: In [19], [20], [6] und [21] beschreiben wir effiziente Modifikationen des Verfahrens von Hansen unter Einsatz von optimierter Bisektion, Cut-Off-, Monotonie- und Konkavitätstests, Intervall-Gauß-Seidel-Schritt mit modifiziertem

Box-Splitting, spezieller Randbehandlung, Eindeutigkeitsnachweisen und automatischer Differentiation. Zielsetzung ist dabei, Einschließungen von hoher Genauigkeit für f^* *und* X^* zu berechnen.

1992: In seinem Buch [9] faßt Hansen Modifikationen früherer Varianten seines Verfahrens, algorithmische Beschreibungen, theoretische Konvergenzuntersuchungen und zahlreiche Testbeispiele zusammen.

1993: In der „Numerical Toolbox for Verified Computing I" [6] wird (unter anderem) erstmals Public-Domain-Software für die globale Optimierung mit automatischer Ergebnisverifikation zur Verfügung gestellt.

1994: Jansson beschreibt in [13] ein sehr effizientes Verfahren unter Verwendung von Ableitungen, mit Eindeutigkeitstests für stationäre Punkte und mit einem sogenannten Expansionsprinzip. Dieses Verfahren ermöglicht es, hochgenaue Einschließungen sowohl für f^* als auch für X^* zu berechnen.

6 Numerische Resultate und Laufzeitvergleiche

Mit den nachfolgend aufgeführten Ergebnissen für Testbeispiele aus der Literatur wollen wir die Effizienz der Intervallverfahren für die globale Optimierung unterstreichen. Wir beschränken uns dabei auf die Resultate in [9], [14], [20] und [21].

6.1 Standard-Testbeispiele

Wir wollen uns zunächst einmal den Funktionen widmen, die in [5] zu einem Standardsatz zum Test von globalen Optimierungsverfahren erklärt worden sind. Neben den Funktionsvorschriften geben wir auch die mit einer geforderten relativen Genauigkeit 10^{-12} berechneten Einschließungen für die Minimalwerte und die globalen Minimalstellen an.

S5, S7, S10: Die Shekel-Funktionen ($x \in \mathbb{R}^4$):

$$f_{\mathrm{S}m}(x) = -\sum_{i=1}^{m} \frac{1}{(x-A_i)(x-A_i)^T + c_i},$$

für $m = 5$, $m = 7$ und $m = 10$. Dabei ist

$$A = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix} \quad \text{und} \quad c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{pmatrix}.$$

Startbereich: $0 \le x_i \le 10$, $i = 1, \ldots, 4$.

Ergebnisse für **S5**:

Minimalstelle:
```
[ 4.000037152818980E+000, 4.000037152821031E+000]
[ 4.000133276591354E+000, 4.000133276591888E+000]
[ 4.000037152819639E+000, 4.000037152819693E+000]
[ 4.000133276591559E+000, 4.000133276591561E+000]
```

Minimum:
```
[-1.015319967905829E+001,-1.015319967905816E+001]
```

Ergebnisse für **S7**:

Minimalstelle:
```
[ 4.000572916185218E+000, 4.000572916186515E+000]
[ 4.000689366185151E+000, 4.000689366185395E+000]
[ 3.999489708859148E+000, 3.999489708859153E+000]
[ 3.999606158858631E+000, 3.999606158858633E+000]
```

Minimum:
```
[-1.040294056681887E+001,-1.040294056681845E+001]
```

Ergebnisse für **S10**:

Minimalstelle:
```
[ 4.000746531591439E+000, 4.000746531592502E+000]
[ 4.000592934138421E+000, 4.000592934138670E+000]
[ 3.999663398040321E+000, 3.999663398040324E+000]
[ 3.999509800586807E+000, 3.999509800586809E+000]
```

Minimum:
```
[-1.053640981669226E+001,-1.053640981669182E+001]
```

H3: Die Hartman-Funktion der Dimension 3 ($x \in I\!R^3$):

$$f_{\mathrm{H3}}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} A_{ij}(x_j - P_{ij})^2\right).$$

$$A = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix} \quad \text{und} \quad P = \begin{pmatrix} 0.3689 & 0.1170 & 0.2673 \\ 0.4699 & 0.4378 & 0.7470 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}.$$

Startbereich: $0 \le x_i \le 1$, $i = 1, \ldots, 3$.

Ergebnisse:

Minimalstelle:
```
[ 1.145248868047914E-001, 1.145248868047942E-001]
[ 5.555230190395223E-001, 5.555230190395230E-001]
[ 8.525997844999939E-001, 8.525997844999945E-001]
```

Minimum:
```
[-3.861305797100195E+000,-3.861305797100181E+000]
```

H6: Die Hartman-Funktion der Dimension 6 ($x \in \mathbb{R}^6$):

$$f_{\mathrm{H6}}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right).$$

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \quad c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix} \quad \text{und}$$

$$P = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}.$$

Startbereich: $0 \le x_i \le 1$, $i = 1, \ldots, 6$.

Ergebnisse:

Minimalstelle:
```
[ 2.016895110066914E-001, 2.016895110067208E-001]
[ 1.500106918234515E-001, 1.500106918234638E-001]
[ 4.768739742218904E-001, 4.768739742219030E-001]
[ 2.753324304940550E-001, 2.753324304940572E-001]
[ 3.116516166001127E-001, 3.116516166001138E-001]
[ 6.573005340656198E-001, 6.573005340656208E-001]
```

Minimum:
```
[-3.322368011415553E+000,-3.322368011415477E+000]
```

BR: Die Branin-Funktion ($x \in \mathbb{R}^2$):

$$f_{\mathrm{BR}}(x) = \left(\frac{5}{\pi}x_1 - \frac{5.1}{4\pi^2}x_1^2 + x_2 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10.$$

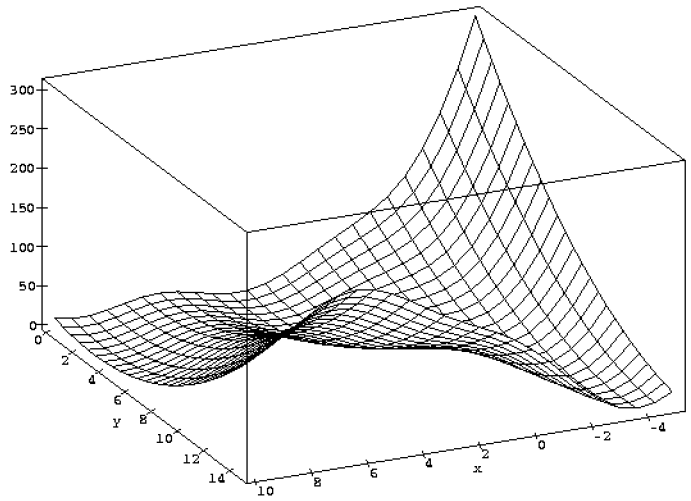


Abbildung 11: Die Branin-Funktion

Startbereich: $-5 \le x_1 \le 10,\ 0 \le x_2 \le 15$.

Ergebnisse:

```
Minimalstelle(n):  1. [ 3.141592653589792E+000, 3.141592653589795E+000]
                      [ 2.274999999999996E+000, 2.275000000000005E+000]
                   2. [ 9.424777960769374E+000, 9.424777960769387E+000]
                      [ 2.474999999999978E+000, 2.475000000000021E+000]
                   3. [-3.141592653589798E+000,-3.141592653589789E+000]
                      [ 1.227499999999998E+001, 1.227500000000002E+001]

Minimum:              [ 3.978873577297381E-001, 3.978873577297435E-001]
```

SHCB: Die *Six-Hump-Camel-Back*-Funktion ($x \in \mathbb{R}^2$):

$$f_{\mathrm{SHCB}}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4.$$

Startbereich: $-2.5 \le x_i \le 2.5$.

Ergebnisse:

```
Minimalstelle(n):  1. [-8.984201310032836E-002,-8.984201310031070E-002]
                      [ 7.126564030207390E-001, 7.126564030207405E-001]
                   2. [ 8.984201310031189E-002, 8.984201310032657E-002]
                      [-7.126564030207403E-001,-7.126564030207390E-001]

Minimum:              [-1.031628453489896E+000,-1.031628453489858E+000]
```

Wir wollen nun die Laufzeiten verschiedener Verfahren vergleichen. Wir verwenden dazu sogenannte Standardzeiteinheiten (auch STUs genannt). Bei der STU handelt es

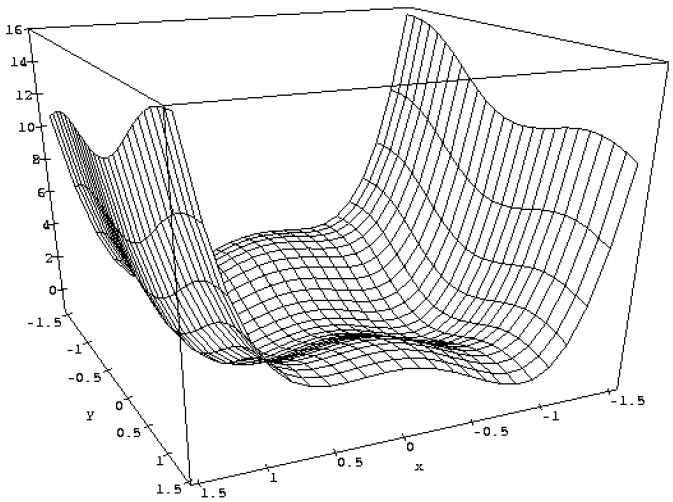


Abbildung 12: Six-Hump-Camel-Back-Funktion

sich um die sogenannte *Standard Time Unit*, die der Berechnungszeit für 1000 reelle Auswertungen der Shekel-5-Funktion (**S5**) entspricht. Sie wird als nahezu von Compiler und Rechner unabhängige Zeitmeßkonstante üblicherweise in der Literatur verwendet.

In der nachfolgenden Tabelle vergleichen wir die Verifikationsverfahren von Hansen [9], Jansson [13] und Ratz [20] mit klassischen Näherungsverfahren [23]. Bei letzteren ist der angegebene STU-Wert ein Mittel aus den Zeiten, die verschiedene dieser approximativen Verfahren benötigen (vgl. [23]).

Vergleich der Standardzeiteinheiten						
	H3	**H6**	**S5**	**S7**	**S10**	**BR**
Näherungsverfahren (im Schnitt)	11.5	29.1	15.4	19.4	22.2	6.2
Verifikationsverfahren (Hansen)	4.4	3641	2.0	6.4	10.2	0.7
Verifikationsverfahren (Ratz)	2.3	57.3	1.2	1.5	2.3	0.7
Verifikationsverfahren (Jansson)	5.6	40.1	1.5	1.8	2.3	2.2

Die Zahlen machen deutlich, daß die neuen Verfahren in ihrer Effizienz durchaus mit den Näherungsverfahren vergleichbar und diesen in den meisten Fällen sogar deutlich überlegen sind. Das ableitungsfreie Verfahren von Jansson [12], das Näherungen für den Minimalwert und die Minimalstellen sowie garantierte Schranken für den Minimalwert liefert, kann sogar mit noch kürzeren Laufzeiten aufwarten.

6.2 Weitere Testbeispiele

Wir wollen nun noch Resultate für einige weitere Testfunktionen aus [9] und [23] angeben. Darunter befinden sich auch extreme Beispiele mit mehreren 1000 oder sogar 10^{10} lokalen Minimalstellen im Optimierungsbereich.

In den jeweils nach den numerischen Ergebnissen folgenden Tabellen zu den Testfunktionen vergleichen wir für die Verfahren von Hansen [9], Ratz [20] und Jansson [14] die Anzahl der notwendigen Funktions-, Gradienten- und Hessematrixauswertungen sowie die Laufzeiten in Standardzeiteinheiten. Der STU-Wert für das Hansen-Verfahren ist ein aus den in [9] angegebenen Laufzeiten grob ermittelter Wert, da keine STU-Angaben vorliegen. Beim Verfahren von Jansson ist zu bemerken, daß aufgrund des Einsatzes approximativer Verfahren neben den Intervallauswertungen noch reelle Auswertungen notwendig sind, die jedoch in den Tabellen nicht aufgeführt sind.

L5: Die Levy-Funktion Nr. 5 ($x \in \mathbb{R}^2$):

$$f_{\mathrm{L5}}(x) \sum_{i=1}^{5} i \cos((i-1)x_1 + i) \sum_{j=1}^{5} j \cos((j+1)x_2 + j) + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2.$$

Startbereich: $-10 \leq x_i \leq 10$, $i = 1, 2$ (darin: 760 lokale Minima!).

Ergebnisse:

Minimalstelle: `[-1.306853009753580E+000,-1.306853009753564E+000]`
`[-1.424845041560682E+000,-1.424845041560681E+000]`

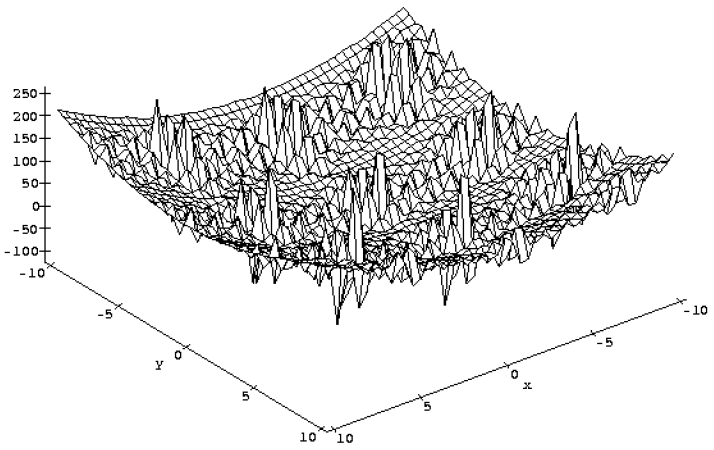


Abbildung 13: Die Levy-Funktion Nr. 5

Minimum: `[-1.761375780016305E+002,-1.761375780016283E+002]`

Die Abbildungen 13 und 14 zeigen die Levy-Funktion Nr. 5 im originalen Optimierungsgebiet und in einem etwas kleineren Ausschnitt davon.

Levy-Funktion Nr. 5			
$\varepsilon = 10^{-12}$	Hansen	Ratz	Jansson
f-Auswertungen	2166	59	732
∇f-Auswertungen	2021	319	2
$\nabla^2 f$-Auswertungen	725	69	11
Laufzeit in STUs	> 30	13.4	7.9

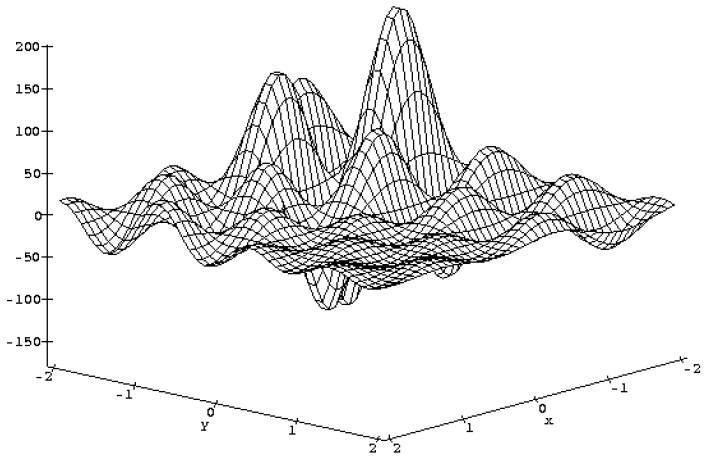


Abbildung 14: Die Levy-Funktion Nr. 5 (Detail)

L12: Die Levy-Funktion Nr. 12 ($x \in \mathbb{R}^{10}$):

$$f_{\mathrm{L12}}(x) = \sum_{i=1}^{9} y_i^2(1+10\sin^2(\pi(1+y_{i+1}))) + \sin^2(\pi(1+y_1)) + y_{10}^2,$$

$$\text{mit } y_i = (x_i - 1)/4,\ i = 1,\ldots,10$$

Startbereich: $-10 \le x_i \le 10$, $i = 1,\ldots,10$ (darin 10^{10} lokale Minima!).

Ergebnisse:

Minimalstelle:
```
[ 9.999999999999998E-001, 1.000000000000001E+000]
[ 9.999999999999998E-001, 1.000000000000001E+000]
...
[ 9.999999999999998E-001, 1.000000000000001E+000]
[ 9.999999999999998E-001, 1.000000000000001E+000]
```

Minimum:
```
[ 0.000000000000000E+000, 6.522368011415477E-030]
```

Abbildung 15 zeigt die Levy-Funktion Nr. 12 in nur zwei Dimensionen, so daß sich der Leser einen Eindruck über die Gestalt der Funktion machen kann.

Levy-Funktion Nr. 12			
$\varepsilon = 10^{-12}$	Hansen	Ratz	Jansson
f-Auswertungen	559	89	141
∇f-Auswertungen	497	177	1
$\nabla^2 f$-Auswertungen	184	41	5
Laufzeit in STUs	> 34	20.1	4.7

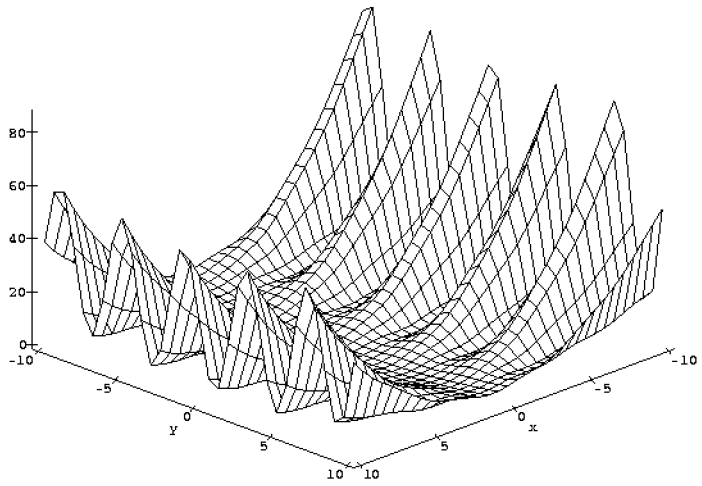


Abbildung 15: Die Levy-Funktion Nr. 12 mit $x \in \mathbb{R}^2$

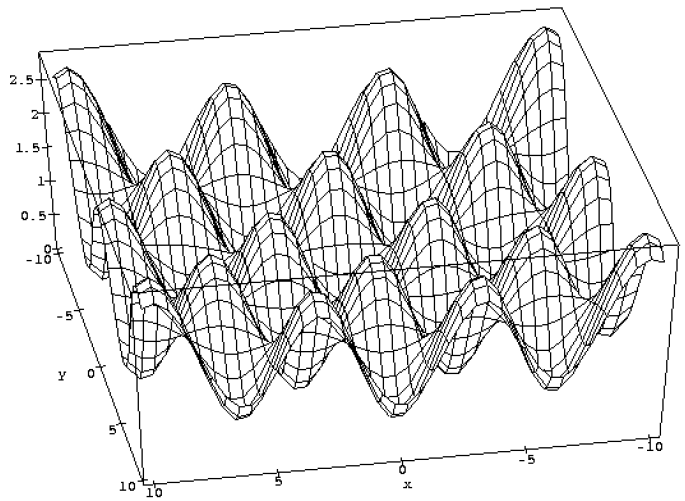


Abbildung 16: Die Griewank-Funktion für $n = 2$

Gn: Die Griewank-Funktionen ($x \in I\!R^n$):

$$f_{Gn}(x) = \sum_{i=1}^{n} \frac{x_i^2}{a_n} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1.$$

Startbereich: $-500 \le x_i \le 600$, $i = 1, \ldots, n$ (darin für $n = 2$ etwa 500 und für $n = 10$ mehrere 1000 lokale Minima!).

Ergebnisse:

Minimalstelle:
```
[ 0.000000000000000E+000, 0.000000000000000E+000]
[ 0.000000000000000E+000, 0.000000000000000E+000]
...
[ 0.000000000000000E+000, 0.000000000000000E+000]
```

Minimum:
```
[ 0.000000000000000E+000, 0.000000000000000E+000]
```

Auch die Griewank-Funktion läßt sich von Intervallverfahren effizienter bearbeiten als von klassischen Verfahren. Vergleicht man die in [23] aufgeführten Ergebnisse für die Methoden von Griewank und Snyman mit den Resultaten für die Verfahren von Jansson [14] und Ratz [20], so ist zunächst festzustellen, daß die verallgemeinerte Abstiegsmethode von Griewank nur ein suboptimales lokales Minimum findet. Für $n = 2$ benötigt Snymans Multi-Start-Algorithmus etwa 1.4 STUs und für $n = 10$ bereits 90 STUs. Das Verfahren von Jansson benötigt für $n = 2$ etwa eine STU und für $n = 10$ rund neun STUs. Unser Verfahren aus [20] benötigt für $n = 2$ etwa zwei STUs und für $n = 10$ rund 16 STUs.

RB: Die Rosenbrock-Funktion ($x \in I\!R^2$):

$$f(x) = 100(x_2 - x_1^2)^2 + (x_1 - 1)^2$$

Startbereiche: $-1.2 \le x_i \le 1.2$, $i = 1, 2$ bzw. $-10^6 \le x_i \le 10^6$, $i = 1, 2$.

Ergebnisse:

Minimalstelle: `[ 9.999999999999603E-001, 1.000000000000133E+000]`
`[ 9.999999999999705E-001, 1.000000000000266E+000]`

Minimum: `[ 0.000000000000000E+000, 2.969873293021112E-023]`

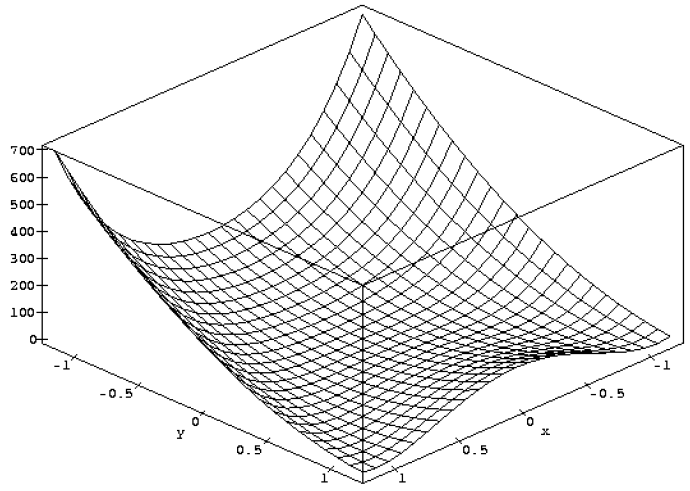


Abbildung 17: Die Rosenbrock-Funktion

Die Rosenbrock-Funktion wurde in der Literatur als eine *schwere* Testfunktion für neue Optimierungsverfahren akzeptiert. Eine ihrer Besonderheiten ist die Tatsache, daß ihre Hessematrix auf allen Punkten der Parabel $800x^2 - 800y + 4 = 0$ singulär ist.

Die nachfolgende Tabelle, in der wiederum die Verfahren von Hansen, Ratz und Jansson verglichen werden, demonstriert die (im Vergleich zur enormen Vergrößerung des Optimierungsbereiches) nur geringe Erhöhung der Auswertungszahlen und Laufzeiten, wenn der Startbereich um den Faktor 10^6 vergrößert wird.

Rosenbrock-Funktion						
$\varepsilon = 10^{-12}$	Hansen	Ratz	Jansson	Hansen	Ratz	Jansson
Startbereich X_i	$[-1.2, 1.2]$			$[-10^6, 10^6]$		
f-Auswertungen	640	111	150	12321	1213	12172
∇f-Auswertungen	583	187	1	12827	2399	1
$\nabla^2 f$-Auswertungen	238	50	13	4949	593	13
Laufzeit in STUs	> 2.5	0.2	1.5	> 50	4.1	35

7 Schlußbemerkung

Viele der Anwender und Entwickler numerischer Verfahren und entsprechender Software werden heutzutage immer noch vom Begriff Intervallrechnung geradezu abgeschreckt. Dies liegt wohl vor allem an der Tatsache, daß dieses relativ junge Gebiet der Mathematik,

aufgrund zahlreicher Fehlinterpretationen von einigen naiven Anwendern der Anfangszeit, mit einem schlechten Ruf zu kämpfen hat. So wird von vielen nach wie vor die Intervallrechnung als diejenige Sparte der Numerik abgetan, die „eigentlich den Wert 5 berechnen will, als Ergebnis aber eine Einschließung $[-1000, 1000]$ liefert". Der Grund liegt darin, daß es bei naiver Anwendung der Intervallarithmetik zu drastischen Überschätzungen der Wertebereiche kommen kann.

Wir hoffen, dieser Artikel weckt bei einigen Lesern, die sich vielleicht bisher eher zu den abgeschreckten zählten, zumindest ein gewisses Interesse an Intervallverfahren. Die vorangegangenen Abschnitte demonstrierten schließlich, daß es speziell auf dem Gebiet der globalen Optimierung hochgenaue und schnelle Intervallverfahren gibt, die sogar effizienter als klassische approximative Methoden arbeiten können. Darüber hinaus sind diese Verfahren in der Lage, mathematische Aussagen über den garantierten Einschluß der gesuchten Lösung(en) zu machen.

Die Grundlage für die Durchführung solcher Verfahren auf einem Rechner bildet eine mathematisch formulierte Rechnerarithmetik [16] und eine entsprechende Implementierung dieser Arithmetik, wie sie z. B. in PASCAL–XSC [15] realisiert wurde. Weitere Voraussetzung für eine erfolgreiche Entwicklung solcher Intervallverfahren ist natürlich stets, daß die eingangs angesprochenen Überschätzungs- und Aufblähungseffekte durch ein problemspezifisches und sorgfältiges Design der Algorithmen weitestgehend vermieden werden.

Literatur

[1] Alefeld, G., Herzberger, J.: *Introduction to Interval Computations*. Academic Press, New York, 1983.

[2] Csendes, T.: *Nonlinear Parameter Estimation by Global Optimization*. Acta Cybernetica, Tom 8 (Fasc. 4), 361–370, 1988.

[3] Csendes, T.: *An Interval Method for Bounding Level Sets of Parameter Estimation Problems*. Computing **41**, 75–86, 1989.

[4] Csendes, T.: *Interval Methods for Bounding Level Sets: Revisited and Tested with Global Optimization Problems*. BIT **30**, 650–657, 1990.

[5] Dixon, L. C., Szegö, G. (Hrsg.): *Towards Global Optimization 2*. North-Holland, Amsterdam, 1978.

[6] Hammer, R., Hocks, M., Kulisch, U., Ratz, D.: *Numerical Toolbox for Verified Computing I – Basic Numerical Problems*. Springer-Verlag, Heidelberg, New York, 1993.

[7] Hansen, E.: *Global Optimization Using Interval Analysis – The One-Dimensional Case*. Journal of Optimization Theory and Applications **29**, 331–344, 1979.

[8] Hansen, E.: *Global Optimization Using Interval Analysis – The Multi-Dimensional Case*. Numerische Mathematik **34**, 247–270, 1980.

[9] Hansen, E.: *Global Optimization Using Interval Analysis*. Marcel Dekker, New York, 1992.

[10] Ichida, K., Fujii, Y.: *An Interval Arithmetic Method for Global Optimization.* Computing **23**, 85–97, 1979.

[11] Jansson, C.: *A Global Minimization Method: The One-Dimensional Case.* Technical Report 91.2, Forschungsschwerpunkt Informations- und Kommunikationstechnik, TU Hamburg-Harburg, 1991.

[12] Jansson, C., Knüppel, O.: *A Global Minimization Method: The Multi-Dimensional Case.* Technical Report 92.1, Forschungsschwerpunkt Informations- und Kommunikationstechnik, TU Hamburg-Harburg, 1992.

[13] Jansson, C.: *On Self-Validating Methods for Optimization Problems.* In: Herzberger, J.: *Studies in Computational Mathematics*, North-Holland, Amsterdam, erscheint 1994.

[14] Jansson, C., Knüppel, O.: *Numerical Results for a Self-Validating Global Optimization Method.* Technical Report 94.1, Forschungsschwerpunkt Informations- und Kommunikationstechnik, TU Hamburg-Harburg, 1994.

[15] Klatte, R., Kulisch, U., Neaga, M., Ratz, D., Ullrich, Ch.: *PASCAL–XSC – Sprachbeschreibung mit Beispielen.* Springer-Verlag, Heidelberg, 1991.

[16] Kulisch, U., Miranker, W. L.: *Computer Arithmetic in Theory and Practice.* Academic Press, New York, 1981.

[17] Moore, R. E.: *Interval Analysis.* Prentice Hall, Engelwood Cliffs, New Jersey, 1966.

[18] Ratschek, H., Rokne, J.: *New Computer Methods for Global Optimization.* Ellis Horwood Limited, Chichester, 1988.

[19] Ratz, D.: *An Inclusion Algorithm for Global Optimization in a Portable PASCAL–XSC Implementation.* In: Atanassova, L. und Herzberger, J. (Hrsg.): *Computer Arithmetic and Enclosure Methods*, 329–338. North-Holland, Elsevier, Amsterdam, 1992.

[20] Ratz, D.: *Automatische Ergebnisverifikation bei globalen Optimierungsproblemen.* Dissertation, Universität Karlsruhe, 1992.

[21] Ratz, D.: *Box-Splitting Strategies for the Interval Gauss-Seidel Step in a Global Optimization Method.* Computing, Springer-Verlag, Wien, erscheint 1994.

[22] Skelboe, S.: *Computation of Rational Interval Functions.* BIT **4**, 87–95, 1974.

[23] Törn, A., Žilinskas, A.: *Global Optimization.* Lecture Notes in Computer Science, No. 350, Springer-Verlag, Berlin, 1989.

[24] Wolfe, M. A.: *Numerical Methods for Unconstrained Optimization – An Introduction.* Van Nostrand Reinhold, New York, 1978.