Linearly-solvable Markov Decision Processes - Projektbericht

Marcel Uhlich
220685

1. Oktober 2008

Technische Universität Berlin

Inhaltsverzeichnis

1 Einleitung **2**

2 Die Einbettung in Gridworlds **2**
2.1 Versuchsaufbau 2
2.2 Auswertung 3
2.2.1 Zeit 3
2.2.2 Korrelationen der Value Functions 3
2.2.3 Policyvergleich 4

3 Ein Hydroelektrisches Kontrollproblem **5**
3.1 Das Problem 5
3.2 Das Model 5
3.2.1 Zustände 5
3.2.2 Aktionen 5
3.2.3 Fluss und Model 6
3.2.4 Übergangswahrscheinlichkeiten 6
3.2.5 Kosten 7
3.2.6 Testparameter 7
3.3 Auswertung 7
3.3.1 Grundlagen 7
3.3.2 Lösung durch Optimierung unter Nebenbedingung 8
3.3.3 Lösung durch Dimensionsreduktion 8
3.3.4 Resultat 8

4 Gegenbeispiel zur Existenz der Einbettung **8**
4.1 Herleitung des Minimierungsproblems 8
4.2 konkretes Beispiel 9

5 Schlusswort **10**

1 Einleitung

Die exakte Berechnung einer Policy für einen diskreten Markov Decision Process kann für große Zustandsräume erhebliche Zeit in Anspruch nehmen. In diesem Bericht wird die Güte und die Geschwindigkeit einer kontinuierlichen Einbettung eines diskreten Markov Decision Process empirisch untersucht. Zunächst wird der einfache Fall einer GridWorld betrachtet und anschließend der Versuch unternommen, die Einbettung auf ein größeres und „realistischeres“ Model anzuwenden. Die Einbettung basiert auf dem Artikel Linearly-solvable Markov decision problems von Emanuel Todorov.

2 Die Einbettung in Gridworlds

2.1 Versuchsaufbau

Wir betrachten Markov Decision Processes mit einem diskreten und endlichen Zustandsraum $Z = \{z_1, z_2, ..., z_N\}$ und endlichen Aktionen $A = \{a_1, a_2, ..., a_n\}$. In allen Versuchen wird davon ausgegangen, dass die Zustandsüberführungfunktion

$$T : Z \times Z \times A \to [0, 1] \quad (z_i, z_j, a) \mapsto P(z_j|z_i, a) \tag{1}$$

bekannt ist. Des Weiteren ist die Bewertungsfunktion

$$\tilde{l} : Z \times A \to \mathbb{R}_+ \quad (z_i, a) \mapsto \tilde{l}(z_i, a) \tag{2}$$

bekannt. Die Bewertungen werden hier als Kosten interpretiert. Die Zustände sind mit Feldern auf einem karierten Spielfeld identifizierbar. Zustände können Hindernisse, normale Felder oder absorbierende Zustände sein. Absorbierende Zustände zeichnen sich dadurch aus, dass Zustandskosten 0 und alle Austrittswahrscheinlichkeiten 0 sind. Der Agent hat die Auswahl aus 4 möglichen Aktionen die den Himmelsrichtungen Norden,Süden,Osten und Westen entsprechen. Der Agent kann sich immer nur ein Feld in eine dieser Richtungen fortbewegen. Sollte in der Zielrichtung ein Hindernis oder das Spielfeldende sein, wird keine Aktion ausgeführt. Die gewünschte Aktion hat eine Erfolgswahrscheinlichkeit p , jede andere Richtung bekommt die Wahrscheinlichkeit $\frac{1-p}{3}$, die Zustandsüberführungsfunktion genügt diesen Anforderungen. Das Ziel ist es, eine Policy π zu finden, so dass die erwarteten Kosten minimiert werden.
Es werden mehrere, zufällig generierte GridWorlds berechnet. Anschließend wird die Güte und die Geschwindigkeit der approximierten Value Function einer klassisch berechneten Value Function gegenüber gestellt. Die aus der Value Function generierten Policies werden ebenfalls verglichen. Als Erfolgswahrscheinlichkeit einer Aktion wird $p = 0.7$ gesetzt, entsprechend bekommen die Misserfolgsrichtungen die Wahrscheinlichkeit 0.1. Die MDPs haben $9, 16, 25, ..., 1600$ Zustände.

Neben der Einbettung wird ein weiterer Algorithmus benutzt um die optimale Value Function zu berechnen. Hierbei handelt es sich um eine klassische [1]Value Iteration. Die Optimierung verläuft mit Discountfaktor 1 (undiscounted).

[1] Sutton, Richard S. : Reinforcement Learning - An Introduction. Chapter 4.4

2.2 Auswertung

2.2.1 Zeit

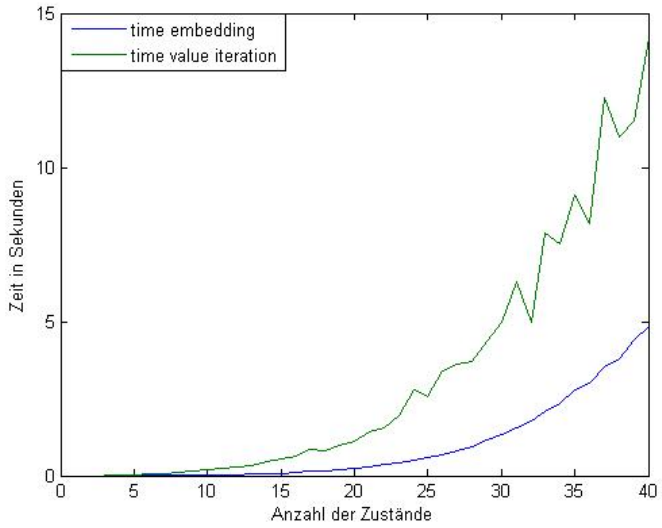

Abbildung 1: Zeitauswertung

Abbildung 1 zeigt die Zeit die beide Verfahren benötigt haben um zu terminieren. Auf der X-Achse sind die Wurzeln der Anzahl der Zustände aufgetragen und die Y-Achse beschreibt die Zeit in Sekunden. Für die blaue Kurve wurden die Zeit für die Berechnung der Einbettung und die Zeit für die Eigenwertiteration (zur Berechnung der Value-function) addiert. Man sieht deutlich, dass die Zeit die man für die Approximation durch die Einbettung benötigt um einiges geringer ist als mit einer herkömmlichen value Iteration. Interessant ist auch zu beobachten, dass der Anstieg der Zeit, die benötigt wird um die Einbettung zu berechnen, wesentlich glatter verläuft. Eine sehr ähnliche Kurve wurde für ca. 20 weitere Versuche beobachtet.

2.2.2 Korrelationen der Value Functions

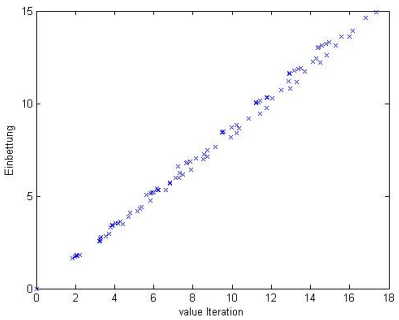

Abbildung 2: 100 Zustände

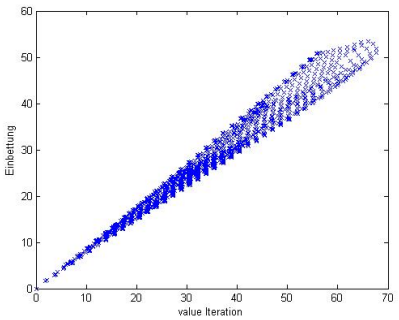

Abbildung 3: 1600 Zustände

In den Abbildungen 2 und 3 wurden die approximierten Values gegen die optimalen Values geplottet. Im idealen Fall sollten alle Punkte auf der Winkelhalbierenden des ersten Quadranten liegen. In diesem Fall wären die approximierten Werte gleich den Optimalen. Auf den Abbildungen sieht man, dass sich die Values entlang einer Geraden verteilen. Diese ist in Richtung der x-Achse geneigt. Dies entspricht auch dem Kommentar Todorovs, dass die approximierten Values numerisch kleiner sind als die optimalen Values. Anhand der skalierung der Achsen sieht man aber, dass die Abweichung von der optimalen Geraden relativ klein sind.

2.2.3 Policyvergleich

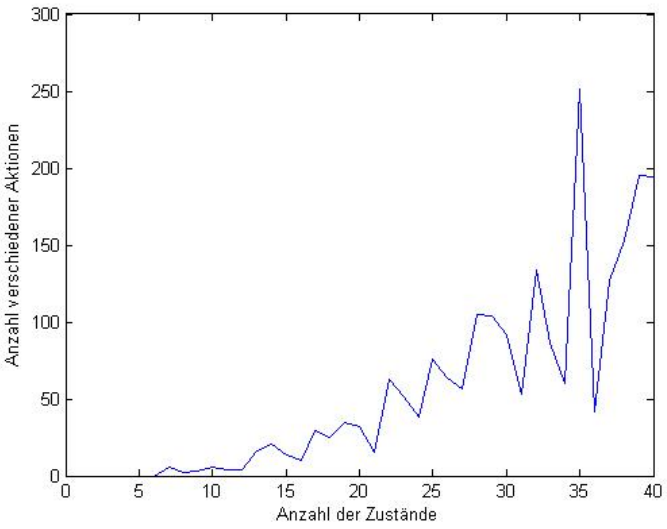

Abbildung 4: Policydifferenz

Abbildung 4 zeigt die Unterschiede der optimalen Policy und der approximierten Policy. Es wurden die Valuefunctions berechnet, und dann aufgrund dieser die Policy bestimmt. Der Algorithmus dafür wird kurz skizziert.

```
Compute value function v;
FOR each state i DO
  Begin
    max := large value;
    out_action := North; //if every action is equally rewarded, go north
    FOR a in { North, South, West, East } DO
      Begin
        IF sum_j P(i,j|a) * (l(i,a) + discount * v(j)) < max THEN
          Begin
            max = sum_j P(i,j|a) * (l(i,a) + discount * v(j));
            out_action = a;
          End;
      End;
    set action in state i to out_action;
  End;
```

Es wird also immer die Aktion gewählt , die die erwarteten Kosten gegeben der Value Function minimiert. Dazu wurden die Policies über die Zustände berechnet und die Aktionen gezählt in denen sich die approximierte Policy und die optimale Policy unterscheiden. Auf der X-Achse wurden die Wurzeln der Anzahl der Zustände aufgetragen und auf der Y-Achse die Anzahl der unterschiedlichen Aktionen. Man sieht deutlich , dass die approximierte Policy Fehler macht. Die Fehlerquote (unterschiedliche Aktionen) liegt bei diesem Plot bei ungefähr 0 bis 20 Prozent (abhängig von der Anzahl der Zustände). Wiederholungen des Experiments haben ähnliche Bilder erzeugt.

3 Ein Hydroelektrisches Kontrollproblem

3.1 Das Problem

Ein Wasserkraftwerk erzeugt eine gewisse Menge an Energie um einen bestimmten Bedarf zu decken. Das Kraftwerk hat die Möglichkeit zu entscheiden wieviel Wasser verwendet werden kann. Wird der Bedarf allerdings nicht gedeckt muss Energie von anderen Erzeugern gekauft werden, wie etwa Kohlekraftwerken. Dieses verursacht Kosten. Wieviel Wasser man verwenden kann hängt vom Inhalt des Reservoirs ab und dieser Inhalt hängt vom Wasserstand des Flusses ab, der den Stausee füllt.
Der Energiebedarf kann relativ gut abgeschätzt werden, während der Wasserstand des Flusses schwer vorherzusagen ist. Daher stellt sich das Problem, gegeben den aktuellen Reservoirinhalten, der aktuellen Zeit im Jahr und der beobachteten Wasserstände der letzten Jahre, soviel Wasser zur Energieerzeugung zu benutzen, so dass die erwarteten Kosten minimiert werden. Das Ziel ist es, zu jedem Zeitpunkt (zum Beispiel Tag,Woche,Monat) die, in Bezug auf die Kosten, optimale Aktion zu wählen. Aktionen sind hier das Benutzen von Wasser zur Energiegewinnung.

3.2 Das Model

Das Problem wird durch einen diskreten Markov Decision Process beschrieben. Das Model orientiert sich am Grand Coulee Staudamm in Colorado (USA). Als Daten liegen die täglichen Flussdichten vom 01.01.1953 bis 01.01.2004 vor.

3.2.1 Zustände

Ein Jahr wird unterteilt in N gleichgroße Zeitintervalle $\{t_1, t_2, ..., t_N\}$. Weiterhin wird das Volumen des Stausees in M Volumeneinheiten $\{V_1, V_2, ..., V_M\}$ zerlegt mit der Eigenschaft $|V_i - V_j| = |V_k - V_l| \; \forall i, j, k, l \; |i - j| = 1 \wedge |k - l| = 1$. Ein Zustand z ist dann definiert als

$$z \in \{V_1, V_2, ..., V_M\} \times \{t_1, t_2, ..., t_N\} \tag{3}$$

Wir setzen $Z := \{V_1, V_2, ..., V_M\} \times \{t_1, t_2, ..., t_N\}$ Der absorbierende Zustand, der Voraussetzung ist, wird im Mai/Juni angesiedelt, da dort aufgrund von Schmelzwasser mehr Wasser zufließt als benutzt und gespeichert werden kann.

3.2.2 Aktionen

Aktionen beschreiben wieviel Wasser in einem Zustand benutz werden kann. Diese Menge wird in Volumen ausgedrückt. Es gibt ein maximales Volumen was man in

einer Zeiteinheit benutzen kann. Eine Aktion ist definiert durch

$$a \in \{a_1, a_2, ..., a_{\max}\} \tag{4}$$

Wir setzen $A := \{a_1, a_2, ..., a_{\max}\}$. Sei $Z = (V, t)$ ein Zustand, eine Aktion heisst gültig wenn $V - a \in \{V_1, V_2, ..., V_M\}$ gilt. Dieses soll sicher stellen, dass nicht mehr Wasser benutzt werden kann als möglich und dass es keine negativen Aktionen gibt. Wie (4) schon suggeriert, gibt es ein Maximum an Wasser was in einem Zeitinterval t benutzt werden kann.

3.2.3 Fluss und Model

Der Zufluss im Zeitinterval i wird durch eine Zufallsvariable X_i $i \in \{1, 2, ..., N\}$ ausgedrückt. Die Werte sind Volumeneinheiten. Es wird definiert :

$$X_i \in \{x_0 = 0, x_1, ..., x_K\} \tag{5}$$

mit der Eigenschaft $V_n + x_i \in \{V_1, V_2, ..., V_M\}$. Dieses ist eine Konsistenzbedingung um innerhalb des Zustandsraumes zu bleiben. Mit den obigen Definition ergibt sich folgendes stochastisches Model :

$$(V_n - a + x_i, t_i) = (V_m, t_{i+1}) \quad n, m \in \{1, 2, ..., M\}\ i \in \{1, 2, ..., N-1\} \tag{6}$$

Nehmen wir an wir wechseln im Interval i von dem Volumenzustand V_n in den Volumenzustand V_m unter der Aktion a, der Zufluss der benötigt wird ist dann :

$$x_i = V_m + a - V_n \tag{7}$$

3.2.4 Übergangswahrscheinlichkeiten

Die Wahrscheinlichkeit von einem Zustand (V_n, t_i) in den Zustand (V_m, t_{i+1}) unter der Aktion a zu wechseln ist abhängig davon, wieviel Wasser im Interval t aus dem Fluss in das Reservoir fließt. Es werden für jedes Interval $i \in \{1, 2, ..., N\}$ Mittelwert μ und Varianz σ^2 aus den Daten geschätzt und eine Normalverteilung über den Fluss angenommen. Die Wahrscheinlichkeit für einen Zustandswechsel ist dann gleich der Wahrscheinlichkeit für den bestimmten Fluss der für diesen Wechsel unter Aktion a nötig ist. Die Wahrscheinlichkeit für einen Fluss ist :

$$P(x_n) = \int_{x_{n-1}}^{x_n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left[\frac{(t-\mu)^2}{\sigma^2}\right]\right) dt \tag{8}$$

Damit ergibt sich für die Zustandsüberführung nach (7) :

$$P((V_m, t_{i+1})|a, (V_n, t_i)) = P(V_m + a - V_n) = P(x_i) \tag{9}$$

Weiterhin ist für $|i - j| > 1, i \geq j$ oder falls $x_i < 0$ $P((V_m, t_j)|a, (V_n, t_i)) = 0$. Wir können also nicht in der Zeit vor bzw. zurück springen und erlauben auch keine negativen Zuflüsse.

3.2.5 Kosten

Der erwartete Bedarf wird durch die Funktion $L : \{t_1, t_2, ..., t_N\} \to \mathbb{R}+$ beschrieben. Die Kostenfunktion ist dann definiert durch :

$$\tilde{l} : Z \times A \quad (V_n, t_i, a) \mapsto L(i) - p \cdot a \left(V_0 + \tfrac{1}{2}(V_n + V_{n'})\right) \tag{10}$$

$V_{n'}$ beschreibt das Volumen im Folgezustand. Die Kraftwerkskonstante p beschreibt die Umwandlung der Energie, V_0 beschreibt zusätzliche Energie (in Volumen ausgedrückt) die unabhängig vom benutzen Wasser ist (z. Bsp. Gesammtdruck des Sees auf die Turbinen). Die Kostenfunktion kann negativ werden, Todorov fordert aber nicht negative Kosten. Daher wird nach Modelberechnung das betragsmäßige Maximum aller Kosten berechnet und auf alle Kosten addiert.

3.2.6 Testparameter

Der Stausee des Kraftwerkes in Colorado hat eine Kapazität 9,7 Mio AF (Acre-Foot). Das Volumen wurde in 195 Volumeneinheiten aufgeteilt und die Intervalle in Monate. Es gibt also $195 \cdot 12 = 2340$ Zustände. Die möglichen Aktionen wurden in 120 Aktionen aufgeteilt. Die Flusswahrscheinlichkeiten wurden für jeden Monat einmal berechnet. p ist $0.00228 \frac{\text{MW-Tage}}{V^2}$ und $L(i) = 24.900$ MW-Tage $\forall i$.

3.3 Auswertung

3.3.1 Grundlagen

Seien $i, l \in Z, k \in A$, wir betrachten die Matrix $B(i)_{kl} = P(l|i, k)$. Des Weiteren genügt die Matrix B allen Anforderungen die für die Einbettung notwendig sind. Wesentlicher Bestandteil der Berechnung der Einbettung ist die Lösung von

$$q\mathbf{1} - Bx = y \tag{11}$$

Dazu wird

$$-Bx = y \tag{12}$$

unter der Bedingung

$$\sum_{i=1}^{n} \exp(x_i) \leq 1 \tag{13}$$

gelößt. Die Dimensionen sind entsprechend anzupassen. x beinhaltet die log-Wahrscheinlichkeiten des kontinuierlichen MDPs, q beschreibt die Kosten im Zustand i und y ist die Differenz von Kosten und Entropie. Die Bedingung (13) wird gestellt, da die Kosten q größergleich Null sein müssen. Ist x eine Lösung von (12) und gilt (13) so ist

$$q = -\log\left(\sum_{i=1}^{n} \exp(x_i)\right) \tag{14}$$

Das Problem ist die Lösung von (12) unter der Bedingung(13). Todorov schlägt zwei Methoden vor um eine Lösung zu erhalten. Multiplikation der Kostenvektoren mit einer Konstanten größer Null und Addition von Vektoren aus dem Nullraum von $(-B)^{\dagger}$. Beide Methoden führen nur in einigen Zuständen zum Ziel, daher werden erweiterte Lösungsmethoden versucht.

3.3.2 Lösung durch Optimierung unter Nebenbedingung

Sei $x \in \mathbb{R}^n$ und $\varepsilon \in \mathbb{R}$. Ist $x_i \leq -\log(n)$ so ist $\sum_{i=1}^{n} \exp(x_i) \leq n \cdot \frac{1}{n} = 1$. Es wird dann

$$\max_{x \in \mathbb{R}} \mathbf{1}^T x \tag{15}$$

gelößt unter den Bedingungen

$$-Bx = y \tag{16}$$

$$x_i \leq -\log(n) + \varepsilon \tag{17}$$

Die Lösung hängt von ε ab. Ist $\varepsilon \leq 0$ so ist jede Lösung konsistent mit den Anforderungen in (13). Erlauben wir größere ε können Lösungen (13) auch verletzen. In beiden Fällen ist aber nicht für alle Zustände eine Lösung zu generieren. Vielmehr ist das Optimierungsproblem für einige Zustände für $\varepsilon \leq 0$ nicht lösbar. Lässt man größere ε zu, so ist in diesen Zuständen (13) verletzt. Probleme gab es vor allem wenn die Anzahl der Zeilen der Matrix B groß wurden. Während es noch einfach war eine Lösung für $B \in \mathbb{R}^{1,195}$ zu finden, war eine Lösung für $B \in \mathbb{R}^{20,195}$ nicht mehr möglich. Dieses Verhalten war in jedem Zustand zu beobachten. Eine Vermutung wäre, dass je geringer die Differenz von Anzahl der Spalten und Zeilen, um so wahrscheinlicher ist die Nichtlösbarkeit. Dies würde aber dem Abschnitt 2 widersprechen, da dort $B \in \mathbb{R}^{4,4} \vee B \in \mathbb{R}^{3,4} \vee B \in \mathbb{R}^{2,4}$ gilt, aber es trotzdem für alle Zustände eine Lösung gab.

3.3.3 Lösung durch Dimensionsreduktion

Ein weiterer Versuch um eine Lösung zu erzeugen war die Reduktion der Dimension. Die stärkste Reduktion waren 5 Volumenzustände pro Monat und 4 mögliche Aktionen. Allerdings hat es auch hier keine vollständige Lösung des Problems gegeben. In Kombination mit 3.3.2 wurde auch kein Ergebnis erzielt.

3.3.4 Resultat

Für keine der vorgestellten Lösungsmethoden konnte eine vollständige Einbettung berechnet werden. Teilweise Lösungen können nicht betrachtet werden, da die Eigenvektoriteration eine vollständige Berechnung der $q(i)$ voraussetzt. Vielmehr hat die Lösung durch Dimensionsreduktion (3.3.3) eine Matrix B erzeugt, die ein Gegenbeispiel zur Existenz einer Einbettung darstellt.

4 Gegenbeispiel zur Existenz der Einbettung

4.1 Herleitung des Minimierungsproblems

Sei $B \in \mathbb{R}^{n,n}$ eine invertierbare Matrix die den Eigenschaften von Todorov (positive Matrixeinträge,normierte Zeilensummen, voller Zeilenrang) genügt. Seien weiterhin

$\tilde{l}, h \in \mathbb{R}^n$ mit $\tilde{l}_i \geq 0$ und $h_i = \sum_{j=1}^{n} B_{ij} \log(B_{ij})$. Betrachte die Funktion

$$f(\varepsilon) = \sum_{i=1}^{n} \exp\left(\left[(-B)^{-1}(\varepsilon\tilde{l} - h)\right]_i\right) \tag{18}$$

Diese Funktion beschreibt die linke Seite in Bedingung (13). Wir zeigen durch Minimierung von $f(\varepsilon)$ anhand eines konkreten Beispiels, dass eine Lösung von (12) unter Bedingung (13) nicht notwendigerweise existiert. ε entspricht dabei dem Skalierungsfaktor den Todorov vorschlägt um eine Lösung zu erhalten. Da wir invertierbare Matrizen B betrachten ist der Kern von $(-B)^{-1}$ natürlich trivial, weshalb der zweite Lösungsvorschlag Todorovs nicht betrachtet werden muss. Betrachte die Ableitungen

$$f'(\varepsilon) = \sum_{i=1}^{n} \left[(-B)^{-1}\tilde{l}\right]_i \exp\left(\left[(-B)^{-1}(\varepsilon\tilde{l} - h)\right]_i\right) \tag{19}$$

$$f''(\varepsilon) = \sum_{i=1}^{n} \left[(-B)^{-1}\tilde{l}\right]_i^2 \exp\left(\left[(-B)^{-1}(\varepsilon\tilde{l} - h)\right]_i\right) \tag{20}$$

Wegen $\exp(x) > 0\ \forall x \in \mathbb{R}$ folgt dass $f''(\varepsilon) > 0$, also ist jedes ε mit $f'(\varepsilon) = 0$ ein Minimum. Daraus folgt bereits die Eindeutigkeit des Minimums, denn wegen $f''(\varepsilon) > 0$ ist $f(\varepsilon)$ konvex und damit das Minimum eindeutig. Allerdings ist die Gleichung $f'(\varepsilon) = 0$ nicht analytisch lösbar, daher wird die Lösung der Gleichung durch das Newtonverfahren

$$m_{n+1} = m_n - \frac{f'(m_n)}{f''(m_n)} \tag{21}$$

approximiert.

4.2 konkretes Beispiel

Betrachte die Matrix $B = \begin{pmatrix} 0.0981 & 0.3744 & 0.4821 & 0.0454 \\ 0.0617 & 0.4181 & 0.3880 & 0.1322 \\ 0.1902 & 0.1633 & 0.3309 & 0.3156 \\ 0.3864 & 0.0356 & 0.3714 & 0.2066 \end{pmatrix}$ und den Kostenvektor $\tilde{l} = \begin{pmatrix} 4 \\ 3 \\ 2 \\ 1 \end{pmatrix}$. Die Determinante von B ist 0.0032 also ist B invertierbar. Wir betrachten nun f, f', f'' wie in (18),(19),(20). Wir starten die Iteration mit dem Wert $m_0 = 0.1$. Bereits nach 5 Iterationen ist $|m_{n+1} - m_n| < 1 \cdot 10^{-8}$. Das Minimum wird über 100000 Iterationen approximiert. Die Ergebnisse sind $\min = -0.0225, f(\min) = 1.2951, f'(\min) = -1.0270 \cdot 10^{(-14)}$. Die Abbildungen zeigen die Funktion $f(\varepsilon)$ einmal mit normaler Achsenskalierung und einmal mit Fokus auf den Bereich, in dem das Minimum angesiedelt ist. Mit $f(\min) = 1.2951 + O(10^{-14})$ gibt es also kein $\varepsilon > 0$ so dass $-Bx = \varepsilon \cdot \tilde{l} - h$ unter der Bedingung (13) lösbar ist, und da B invertierbar ist existiert die kontinuierliche Einbettung also nicht zwingend.

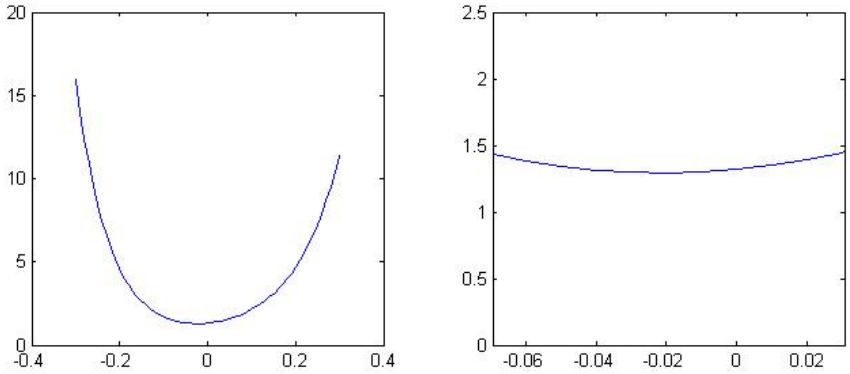

Abbildung 5: $f(\varepsilon)$

5 Schlusswort

Da die Existenz der Einbettung nicht immer unter den genannten Voraussetzungen gegeben ist, gestaltet sich die Anwendung auf ein beliebiges Problem schwierig. Wenn eine Einbettung existiert, so haben wir gesehen dass die Zeit zur Berechnung einer solchen und anschließender Approximation der Value-Function um einiges schneller ist als herkömmliche Verfahren zur Berechnung der optimalen Value-Function. Allerdings machen wir dann auch Fehler. Wenn man sicher ist, dass eine Einbettung existiert kann man sich sicherlich Fragen ob man sich diese Fehler leisten kann. Dies hängt dann aber vom konkreten Problem ab.
Es bleibt die Frage ob man gewisse Eigenschaften an die Matrix B fordern kann, um doch eine Einbettung zu erhalten. Betrachte zum Beispiel eine

Matrix $B' = \begin{pmatrix} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.7 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.7 & 0.1 \end{pmatrix}$ und den Rewardvektor $r \in \mathbb{R}^4$ mit $r_i = 1 \; \forall i$.

Also ein Beispiel aus dem GridWorld-Problem in Teil 2. Dann gibt es eine Lösung von (12) unter (13). Vergleicht man diese Matrix mit der Matrix B des Gegenbeispiels fällt zunächst auf, dass bei B' auch die Spaltensummen 1 sind und die Einträge von B näher an der Gleichverteilung sind als bei B'. Ob dies wichtige oder unwichtige Kriterien sind, kann an dieser Stelle nicht geklärt werden.

Literatur

[1] Linearly-solvable Markov decision problems Emanuel Todorov Advances in Neural Information Processing Systems 19: 1369-1376, Scholkopf et al (eds), MIT Press (2006)

[2] Reinforcement Learning: An Introduction Richard S. Sutton and Andrew G. Barto