

Lösen von dynamischen Wirtschaftsmodellen mit dünnen Gittern

Steffen Ruttscheidt

Geboren am 6. Mai 1994 in Bonn

19. Januar 2016

Bachelorarbeit Mathematik

Betreuer: Prof. Dr. Garcke

Zweitgutachter: Prof. Dr. Griebel

INSTITUT FÜR NUMERISCHE SIMULATION

MATHEMATISCH-NATURWISSENSCHAFTLICHE FAKULTÄT DER
RHEINISCHEN FRIEDRICH-WILHELMS-UNIVERSITÄT BONN

Danksagung

Hiermit möchte ich mich bei allen Personen bedanken, die mich während des Studiums und insbesondere bei dieser Arbeit unterstützt haben.

Zunächst möchte ich mich bei Prof. Dr. Garcke dafür bedanken, dass er mir ein interessantes Thema für die Bachelorarbeit vorgeschlagen hat. Des Weiteren wurden meine Fragen ausführlich beantwortet und die Diskussionen bei den aufgetretenen Problemen haben mir sehr weitergeholfen.

Bei Prof. Dr. Griebel möchte ich mich für das Zweitgutachten bedanken.

Mein besonderer Dank geht an Vasil Velikov, der mir während der gesamten Bachelorarbeit als Ansprechpartner zur *HCFFT*-Bibliothek zur Seite stand. Zudem hat er die *HCFFT*-Bibliothek um Funktionen erweitert, die mir die Nutzung von dieser ermöglichten.

Zuletzt danke ich Gabriel Aguirre Chan und Daniel Koenen für das Korrekturlesen.

Inhaltsverzeichnis

1	Einleitung	1
2	Lösen von dynamischen Wirtschaftsmodellen	4
2.1	Repräsentatives Agenten-Wachstums-Modell	6
2.1.1	Eulergleichung des repräsentativen Agenten-Wachstums-Modell	7
2.2	Multi-Country-Modell	8
2.3	Lösen der Eulergleichung	9
3	Dünne Gitter	11
3.1	Konstruktion eines dünnen Gitters	12
3.2	Interpolationspolynom	14
3.2.1	Basisfunktionen des Interpolationspolynoms	15
3.2.2	Koeffizienten des Interpolationspolynoms, Lagrangeansatz	16
3.3	Anisotrope dünne Gitter	16
3.3.1	Konstruktion von anisotropen dünnen Gittern	16
3.4	HCFFT	17
3.5	Vorbereitung	17
4	Numerische Utensilien	19
4.1	Ergodische Menge	19
4.2	Numerische Integrationsmethoden	20
4.2.1	Monte-Carlo-Integration	20
4.2.2	Ein-dimensionale Gauß-Hermite-Quadratur	21
4.2.3	Mehr-dimensionale deterministische Quadratur	21
5	Der Algorithmus	24
5.1	GSSA	24
5.2	Dünn-Gitter-Algorithmus	28
5.3	Änderungen der Algorithmen für das Multi-Country-Modell . . .	31
6	Ergebnisse der Implementierung	33
6.1	Ergebnisse beim Agenten-Wachstums-Modell	33
6.2	Ergebnisse beim Multi-Country-Modell	35
7	Zusammenfassung und Ausblick	38

8 Notationsverzeichnis	39
Literaturverzeichnis	42

Kapitel 1

Einleitung

Ökonomische Modelle helfen beispielsweise Managern, Politikern und Ökonomen den wirtschaftlichen Entscheidungsprozess zu analysieren. Jedes Modell basiert auf einer Reihe von Annahmen, da die wirtschaftlichen Hintergründe sehr komplex sind. Diese vereinfachen einerseits das Modell, andererseits verringert sich die Anzahl der benötigten Daten und das Modell ist somit besser anwendbar. Wichtig ist hierbei die richtige Wahl der Komponenten zu treffen, die in das Modell mit aufgenommen werden.

Sowohl Individuen als auch Firmen haben begrenztes Kapital oder andere begrenzte Ressourcen wie Rohstoffe zur Verfügung. Budgetbedingungen beeinflussen nahezu alle wirtschaftlichen Entscheidungen, da die Summe der Ausgaben nicht das verfügbare Kapital überschreiten sollte, wobei in manchen Fällen Kredite genommen werden können.

Viele Modelle haben die Maximierung des Nutzens unter Minimierung der Kosten zum Ziel, was eine gute Aufteilung der Ressourcen erfordert. Fast alle Modelle nehmen an, dass alle Entscheidungen rational sind. Das vereinfacht die Analyse, ist jedoch nicht unbedingt in der Realität gegeben, da beispielsweise Emotionen die Entscheidungen beeinflussen können.

In der Wirtschaftsliteratur werden immer komplexere dynamische Probleme mit einer größer werdenden Zahl von betrachteten Variablen analysiert. So gibt es Modelle, bei denen viele heterogene Agenten oder heterogene Firmen untersucht werden sollen. Weiter gibt es Modelle mit einer großen Anzahl von Zustandsvariablen und Lebenszyklus-Modelle, bei denen mindestens so viele Variablen vorliegen, wie es Perioden gibt.

Zudem werden bestehende Modelle um neue Parameter erweitert, die es ermöglichen genauere Analysen durchzuführen und in einigen Anwendungen müssen dynamische Wirtschaftsmodelle mehrfach unter verschiedenen Parametern gelöst werden.

Dynamische Modelle haben meistens keine geschlossene Lösung und müssen somit mithilfe von numerischen Methoden gelöst werden.

Aufgrund der großen Anzahl von Agenten oder Zustandsvariablen sind die Kosten für die Analyse extrem hoch. Deshalb ist es wichtig, effiziente Ansätze zur Lösungsfindung zu nutzen.

Problemstellung und Lösungsansätze in verwandter Literatur

In dieser Arbeit wird ein Modell mit bis zu zehn heterogenen Agenten und bis zu 20 Zustandsvariablen betrachtet. Hierbei soll der erwartete Gesamtnutzen über unendlich viele Perioden maximiert werden, wobei der Nutzen in der Zukunft durch einen Diskontfaktor abgewertet wird.

In der Literatur werden Perturbation, Projektion und stochastische Simulation zur Lösungsfindung genutzt. Diese Methoden haben verschiedene Nachteile. So verwenden Projektionsalgorithmen Produktansätze, welche in hoher Dimension aufgrund des Fluchs der Dimensionen nicht tragbar sind. Stochastische Simulation in Verbindung mit Monte-Carlo-Integration und kleinste-Quadrate Schätzern leidet unter der geringen Genauigkeit der Monte-Carlo-Integration und der numerischen Instabilität von kleinste-Quadrate Ansätzen.

Perturbations-Methoden lösen Modelle großer Dimension im Gleichgewicht mithilfe von Taylorreihen der Modellgleichungen. Allerdings ist ihre Genauigkeit unklar, besonders im nichtlinearen Fall.

Lösungsansätze in dieser Arbeit

In dieser Arbeit wird eine Kombination der oben genannten Ansätze verwendet, welche die Vorteile der jeweiligen Methoden ausnutzt.

Zunächst werden dünne Gitter verwendet, um die Kosten des Projektionsansatzes mit Tensorproduktgittern zu reduzieren. Weiter werden deterministische Quadraturregeln benutzt, um die Ungenauigkeit der Monte-Carlo-Quadratur zu umgehen. Insbesondere wird eine Monomregel vorgestellt, bei der die Kosten nur linear mit der Dimension steigen, was sie sehr attraktiv bei hohen Dimensionen macht. Darüber hinaus wird die Lösung nur auf dem relevanten Zustandsraum betrachtet, um dort eine höhere Genauigkeit zu erzielen. Zuletzt wird Vorberechnung genutzt, um die Kosten im iterativen Teil zu verringern. Die Einleitung orientiert sich an der von [SJ13] und [JMMV14].

Eigener Anteil

- Implementierung des GSSA aus [JMM09] in C
- Implementierung des Dünn-Gitter-Algorithmus aus [JMMV14] in C
- Alternativer Ansatz zum Matlab-Code von [JMMV14] für die Verwertung der Lösung des GSSA, welcher weder die Erstellung der Matrix der ausgewerteten Basisfunktionen (nicht in der *HCFIT* enthalten), noch das Lösen eines überbestimmten Gleichungssystems benötigt.
- Benutzung der *HCFIT*-Bibliothek zur Konstruktion und Interpolation auf dünnen Gittern
- Anregungen zur Erweiterung der *HCFIT*-Bibliothek um:
 - Chebyshev-Gauß-Lobatto-Gitterpunkte

- Chebyshev-Polynome werden jetzt durch die rekursive Darstellung berechnet und nicht durch $T_n(x) = \cos(n \arccos(x))$. Dies ermöglicht die Auswertung außerhalb von $[-1,1]$.
- Verwertung mehrerer Koeffizienten für die Auswertung bei gleichen Interpolationspunkten (schnellere Laufzeit)

Aufbau der Arbeit

In Kapitel 2 werden die betrachteten dynamischen Modelle – das repräsentative Agenten-Wachstums-Modell und das Multi-Country-Modell – erläutert und wichtige Fachbegriffe aus der Wirtschaftstheorie eingeführt. Des Weiteren wird erklärt, wie die Lösung eines solchen Problems gefunden wird.

Im nachfolgenden Kapitel werden dünne Gitter eingeführt. Es wird beschrieben, wie diese konstruiert werden und wie darauf interpoliert wird. Zuletzt wird das Prinzip der Vorberechnung dargestellt.

Das vierte Kapitel beschreibt numerische Methoden, welche im, in Kapitel 5 beschriebenen, Algorithmus verwendet werden. Zunächst wird hier erläutert, wie das Problem nur auf dem relevanten Zustandsraum gelöst wird. Danach werden verschiedene Quadraturmethoden beschrieben und es wird erklärt, wie mit korrelierten Zufallsvariablen umgegangen wird.

In Kapitel 5 wird der Algorithmus erklärt, welcher für diese Arbeit implementiert wurde. Er besteht aus zwei Teilalgorithmen, wobei der GSSA nur bessere Startwerte und den relevanten Zustandsraum für den Dünn-Gitter-Algorithmus liefert. Zuletzt werden Änderungen der Algorithmen beschrieben, durch die der, für das repräsentative Agenten-Wachstums-Modell beschriebene, Algorithmus eine Lösung für das Multi-Country-Modell liefert.

Das sechste Kapitel zeigt die Ergebnisse der Implementierung bei verschiedenen Modellparametern und unterschiedlicher Anzahl von Agenten.

Im siebten Kapitel werden die Resultate und Ansätze zusammengefasst und es wird ein Ausblick gegeben.

Es folgt das Notationsverzeichnis und das Literaturverzeichnis.

Der Code befindet sich auf der beigelegten CD.

Kapitel 2

Lösen von dynamischen Wirtschaftsmodellen

Es gibt eine Vielzahl von dynamischen Wirtschaftsmodellen, welche zur Analyse, Optimierung oder Prognose zukünftiger Entwicklungen verwendet werden. In dieser Arbeit werden Modelle betrachtet, welche die Maximierung des lebenslangen Erwartungsnutzens von einem oder mehreren Agenten unter bestimmten Nebenbedingungen zum Ziel haben.

Der lebenslange Erwartungsnutzen entspricht der diskontierten Summe von Nutzenfunktionen über alle Perioden ab der Anfangsperiode. Dabei wird der Diskontfaktor für den Nutzen nachfolgender Perioden verwendet, um die Zeitpräferenz des Konsums eines Gutes darzustellen.

Durch eine Ressourcennebenbedingung wird das Verhältnis zwischen Kapital, Konsum und Produktion der jeweiligen Periode und dem Kapital der darauffolgenden Periode dargestellt. Durch weitere Nebenbedingungen können beispielsweise stochastische Komponenten modelliert werden.

Das Lösen eines dynamischen Optimierungsproblems lässt sich nach [Cao08] generell in folgende Schritte aufteilen, welche im Verlauf des Kapitels näher erklärt werden. Zunächst wird das Problem rekursiv formuliert, Zustands- und Kontrollvariablen bestimmt und die Bellmangleichung formuliert. Nutzen des Lagrangeansatzes ([For08]) oder des Umhüllungssatzes ([MCWG95], [SB94] und [Pam09]) liefert die Bedingungen erster Ordnung. Zuletzt wird auf die Policyfunktion aufgelöst, was beispielsweise durch iterative Substitution oder numerische Simulation realisiert werden kann.

Dynamische Programmierung ([Ski08]) ist eine wichtige und in dieser Arbeit genutzte Methode zum Lösen von dynamischen Optimierungsproblemen. Zur Erklärung des Aufbaus und des Lösungsansatzes sei nun folgendes simple Modell betrachtet:

$$\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t u(c_t) \quad (2.1)$$

$$\text{s. d. } k_{t+1} = f(c_t, k_t) \quad (2.2)$$

Dabei wird mit β der Diskontfaktor mit u die Nutzenfunktion und mit f die Produktionsfunktion bezeichnet. Im ökonomischen Kontext betrachtet, maximiert der repräsentative Agent seinen lebenslangen Nutzen $\sum_{t=0}^{\infty} \beta^t u(c_t)$ durch Wählen seines Konsums c_t gegeben dem Kapital k_t in Periode t . Hierbei werden Variablen, deren Werte vom Konsumenten entschieden werden, *Kontrollvariablen* genannt und solche, die nicht direkt gewählt werden, *Zustandsvariablen*. Als *Policyfunktion* wird eine Funktion $c_t = h(k_t)$ bezeichnet, die für jeden Zustand k_t die optimale Konsumententscheidung c_t liefert. Durch das Wählen eines Konsums c_t wird der Zustand der nächsten Periode durch die, von der Produktionsfunktion abhängige, Nebenbedingung berechnet. Da sich der Vorgang in allen folgenden Perioden wiederholt wird er rekursiv genannt.

Die Grundidee der dynamischen Programmierung ([Bel57]) ist es, ein multi-periodisches Problem in eine Sequenz von zwei-Perioden-Problemen umzuschreiben. Dafür wird die rekursive Struktur des Problems ausgenutzt.

$$\begin{aligned} V(k_t) &= \max_{c_t, k_{t+1}} \sum_{i=0}^{\infty} \beta^i u(c_{t+i}) = \max_{c_t, k_{t+1}} \left\{ u(c_t) + \beta \sum_{i=0}^{\infty} \beta^i u(c_{t+i+1}) \right\} \\ &= \max_{c_t, k_{t+1}} \left\{ u(c_t) + \beta V(k_{t+1}) \right\} \end{aligned} \quad (2.3)$$

$$\text{s.d. } k_{t+1} = f(c_t, k_t)$$

Hierbei ist die Wertfunktion $V(\cdot)$ nur von k_t abhängig, da der optimale Wert von c_t eine Funktion von k_t ist.

Die Gleichung $V(k_t) = \max_{c_t, k_{t+1}} u(c_t) + \beta V(k_{t+1})$ wird *Bellmangleichung* genannt. Zuletzt wird die *Eulergleichung*

$$u'(c_t) = \beta u'(c_{t+1}) \frac{\partial f(c_t, k_t)}{c_t} \quad (2.4)$$

bestimmt, die für jede Lösung erfüllt ist und somit eine notwendige Bedingung liefert. Diese Darstellung erlaubt es, bei bestimmten Problemen die Lösung leichter zu finden. Die Eulergleichung kann beispielsweise mit dem Lagrangeansatz oder dem Umhüllungssatz erhalten werden.

Wie schon erklärt, gibt die Policyfunktion $h(\cdot)$ die optimale Lösung c_t für jede Periode t für gegeben Zustand k_t an. Analog gibt es auch eine Policyfunktion $k_{t+1} = \bar{h}(k_t)$. Zum aktuellen Zeitpunkt ist k_{t+1} noch keine Zustandsvariable, sondern lässt sich durch Wahl des Konsums indirekt als Kontrollvariable ansehen, falls dies durch die Produktionsfunktion in der Nebenbedingung wohldefiniert ist.

Nach [Kru12] existiert die Lösung zur Bellmangleichung und ist eindeutig. Weiter konvergiert die Fixpunktiteration unabhängig vom Startwert gegen die korrekte Lösung. Beides folgt aus dem Banachschen Fixpunktsatz beziehungsweise Blackwells Theorem.

Das Bellmansche Optimalitätsprinzip sagt aus, dass die Lösung der Bellmangleichung eine Lösung des ursprünglichen Problems ist.

Die Transversalitätsbedingung ist eine hinreichende Bedingung dafür, dass die Lösung der Eulergleichung eine Lösung des ursprünglichen Modells ist. Für die Notwendigkeit der Transversalitätsbedingung gibt es keine generelle Aussage.

Wenn für die Eulergleichung keine geschlossene Lösung existiert, muss das Problem numerisch gelöst werden. Hierfür werden in der Literatur verschiedene Ansätze, unter anderem Zeititeration und Fixpunktiteration, verfolgt, welche im Abschnitt 2.3 genauer beschrieben werden.

Das Ziel dieser Arbeit ist es, die Policyfunktion bei den nachfolgend beschriebenen Modellen zu approximieren, da diese die optimale Konsumwahl für jeden Zustand liefert.

2.1 Repräsentatives Agenten-Wachstums-Modell

Das repräsentative Agenten-Wachstums-Modell unterscheidet sich vom obigem Modell vor allem darin, dass es eine stochastische Komponente beinhaltet.

$$\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t u(c_t) \right] \quad (2.5)$$

$$\text{s. d. } c_t + k_{t+1} = (1 - \delta)k_t + a_t f(k_t) \quad (2.6)$$

$$\ln(a_t) = \rho \ln(a_{t-1}) + \sigma \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2) \quad (2.7)$$

Der Anfangszustand (k_0, a_0) ist dabei vorgegeben, wobei $k_t \geq 0$ das Kapital und $a_t \geq 0$ die Produktivität der Periode t darstellt. Weiter ist $c_t \geq 0$ der Konsum, $\beta \in (0,1)$ der Diskontfaktor, $\delta \in (0,1]$ die Abschreibungsrate und $u(c_t) \in \mathcal{C}^1$ die Nutzenfunktion, welche als steigend und konkav angenommen wird. \mathbb{E}_t bezeichnet den Operator der bedingten Erwartung bezüglich des Zustandes (k_t, a_t) und mit $f(k_t) \in \mathcal{C}^1$ wird die Produktionsfunktion bezeichnet. Das Produktivitätslevel a_t folgt einem autoregressiven Prozess erster Ordnung mit Autokorrelationskoeffizienten $\rho \in (-1,1)$ und Standardabweichung $\sigma > 0$. Die Schocks werden zu Beginn jeder Periode realisiert. Dementsprechend ist die Produktivität bekannt, wenn über den Konsum entschieden und das Kapital der nächsten Periode berechnet wird.

Im wirtschaftlichen Kontext betrachtet wird der lebenslange Erwartungsnutzen (2.5) unter der Ressourcen Nebenbedingung (2.6) maximiert. Durch diese wird, abhängig vom Konsum c_t , Kapital k_t und Produktivität a_t , das Kapital k_{t+1} der nachfolgenden Periode berechnet. Die Modellierung ist so gewählt, dass das Kapital k_{t+1} der Summe aus

- dem Kapital k_t (abgewertet durch die Abschreibungsrate δ , welche die Wertminderung des Kapitals darstellt),
- der vom Kapital k_t abhängigen Produktionsfunktion f (gewichtet mit dem Produktivitätsfaktor a_t),
- abzüglich des Konsums c_t

entspricht.

Die Nebenbedingung (2.7) repräsentiert die Schwankungen in der Wirtschaft. So können beispielsweise negative Schwankungen durch den Ausfall einer Maschine und positive Schwankungen durch die Entwicklung einer neuen Technologie entstehen. In diesem Modell wird die Produktivität der aktuellen Periode als abhängig von der Produktivität der vorherigen Periode angenommen. Durch diese Korrelation gibt es, außer bei sehr kleinem Autokorrelationskoeffizienten ρ , keine schlagartigen Wechsel der Produktivität. Die Normalverteilung mit Erwartungswert null lässt Ausschläge in beide Richtungen mit gleicher Wahrscheinlichkeit zu und verhindert, bei kleiner Standardabweichung σ , häufiges Auftreten von großen Schocks.

2.1.1 Eulergleichung des repräsentativen Agenten-Wachstumsmodell

Die folgenden Umformungen, welche benutzt werden um die Eulergleichung zu erhalten, können beispielsweise in [Smi99], [Ger06] und [Ell08] gefunden werden. Auch hier kann die Bellmangleichung, ähnlich wie zuvor, aufgestellt werden.

$$V(k_t, a_t) = \max_{k_{t+1}} u(\underbrace{(1 - \delta)k_t - k_{t+1} + a_t f(k_t)}_{=:H}) + \beta \mathbf{E}_t[V(k_{t+1}, a_{t+1})]$$

Die Bedingung erster Ordnung bezüglich k_{t+1} liefert nach dem Satz über monotone Konvergenz

$$\frac{\partial H}{\partial k_{t+1}} = -u'(c_t) + \beta \mathbf{E}_t \left[\frac{\partial V(k_{t+1}, a_{t+1})}{\partial k_{t+1}} \right] \stackrel{!}{=} 0.$$

Es resultiert

$$u'(c_t) = \beta \mathbf{E}_t \left[\frac{\partial V(k_{t+1}, a_{t+1})}{\partial k_{t+1}} \right].$$

Nach dem Umhüllungssatz folgt

$$\frac{\partial V(k_t, a_t)}{\partial k_t} = u'(c_t)(1 - \delta + a_t f'(k_t)).$$

Da die Gleichungen für alle t gelten, insbesondere für $t+1$, ergibt sich durch einen Periodenshift eine Periode vorwärts

$$\frac{\partial V(k_{t+1}, a_{t+1})}{\partial k_{t+1}} = u'(c_{t+1})(1 - \delta + a_{t+1} f'(k_{t+1})).$$

Zusammen ergibt sich die Eulergleichung

$$u'(c_t) = \beta \mathbf{E}_t \left[u'(c_{t+1})(1 - \delta + a_{t+1} f'(k_{t+1})) \right]. \quad (2.8)$$

Das Lösen der Eulergleichung folgt in 2.3.

2.2 Multi-Country-Modell

Zum Testen der Performance bei höherdimensionalen Problemen wird das Multi-Country-Modell nach [JMMV14] beschrieben. Es kann als eine Erweiterung des repräsentativen Agenten-Wachstums-Modell auf mehrere Agenten aufgefasst werden. Hier werden $N \in \mathbb{N}$ Agenten betrachtet, welche als Länder $h \in \{1, \dots, N\}$ interpretiert werden.

Ziel ist es, den erwarteten lebenslangen Weltnutzen zu maximieren. Das heißt, die Länder handeln optimal für einen maximal großen Gesamtnutzen und nicht für die Maximierung ihres eigenen Nutzens. Formal beschrieben folgt für einen sozialen Planer das Optimierungsproblem

$$\max_{\{c_t^h, k_{t+1}^h\}_{t=0, \dots, \infty}^{h=1, \dots, N}} \mathbb{E}_0 \left[\sum_{h=1}^N (\lambda^h \sum_{t=0}^{\infty} \beta^t u^h(c_t^h)) \right] \quad (2.9)$$

unter den Bedingungen

$$\sum_{h=1}^N c_t^h + \sum_{h=1}^N k_{t+1}^h = \sum_{h=1}^N k_t^h (1 - \delta) + \sum_{h=1}^N a_t^h A f^h(k_t^h) \quad (2.10)$$

$$\ln(a_{t+1}^h) = \rho \ln(a_t^h) + \epsilon_{t+1}^h, \quad h = 1, \dots, N. \quad (2.11)$$

Hierbei sind k_t^h , a_t^h , c_t^h , u^h , E_t , β und δ definiert wie im repräsentativen Agenten-Wachstums-Modell, wobei h das jeweilige Land angibt. A ist eine Normalisierungskonstante der Produktionsfunktion und λ^h bezeichnet das Wohlfahrtsgewicht, welches die Wichtigkeit des Nutzens des jeweiligen Landes für den Gesamtnutzen beschreibt. Der Anfangszustand $\{k_0^h, a_0^h\}_{h=1, \dots, N}$ ist gegeben. Die Produktivitätsschocks sind multivariat normal verteilt

$$(\epsilon_{t+1}^1, \dots, \epsilon_{t+1}^N)^T \sim \mathcal{N}(0_N, \Sigma),$$

wobei $0_N \in \mathbb{R}^N$ Erwartungswert null für alle Länder angibt und $\Sigma \in \mathbb{R}^{N \times N}$ die Kovarianzmatrix ist. Es wird angenommen, dass $\epsilon_{t+1}^h = \zeta_t^h + \zeta_t$, $h = 1, \dots, N$ mit $\zeta_t^h \sim \mathcal{N}(0, \sigma^2)$ eine landesspezifische und $\zeta_t \sim \mathcal{N}(0, \sigma^2)$ eine weltweite Komponente ist. Daraus resultiert die Kovarianzmatrix

$$\begin{pmatrix} 2\sigma^2 & \dots & \sigma^2 \\ \dots & \dots & \dots \\ \sigma^2 & \dots & 2\sigma^2 \end{pmatrix}. \quad (2.12)$$

Im wirtschaftlichen Kontext betrachtet, wird in (2.9) der lebenslange erwartete Gesamtnutzen aller Länder dargestellt. Anstelle der Ressourcen Nebenbedingung (2.6) aus dem repräsentativen Agenten-Wachstums-Modell für jedes Land, muss nur die Summe von (2.6) über alle Länder gelten. Dementsprechend wird durch (2.10) eine Bedingung beschrieben, welche es ermöglicht, dass Länder die Ressourcen Nebenbedingung (2.6) verletzen. Das liegt daran, dass hier die weltweiten Zustände und Ressourcen beschrieben werden, welche den Zusammenhang auf der ganzen Welt (über die betrachteten Länder) darstellen. Durch das Auflösen der Gleichung (2.10) auf $\sum_{h=1}^N k_{t+1}^h$ kann diese so interpretiert

werden, dass das Gesamtkapital der nächsten Periode durch (2.10) bestimmt ist, die Aufteilung auf die Länder jedoch nicht direkt.

Durch (2.11) werden die Produktivitäten analog zum repräsentativen Agenten-Wachstums-Modell berechnet, wobei hier ein weltweiter und ein landesspezifischer Schock vorliegt.

In dieser Arbeit wird das Modell mit identischen Präferenzen ($u^h = u$) und identischem technologischem Fortschritt ($f^h = f$) jeweils für $h = 1, \dots, N$ betrachtet. Unter diesen Annahmen wird ein sozialer Planer allen Ländern das gleiche Wohlfahrtsgewicht $\lambda^h = 1$ zuweisen, was wiederum im gleichem Konsum aller Länder $c_t^h = c_t$, $h = 1, \dots, N$ resultiert. Da die Effizienz bezüglich höherdimensionalen Problemen betrachtet werden soll, wird die Symmetrie der Problemstellung nicht ausgenutzt.

2.3 Lösen der Eulergleichung

In verwandter Literatur werden hauptsächlich Zeititeration und Fixpunktiteration zur Lösungsfindung der Eulergleichung verwendet.

Nach [JMMV14] wird bei der Zeititeration zunächst die Kapitalentscheidungsfunktion (Policyfunktion) mithilfe eines Interpolationspolynoms $k_{t+1} = K(k_t, a_t; b)$ parametrisiert, wobei b der Koeffizientenvektor ist. Danach wird auf jedem Punkt des Gitters ein System aus den drei Gleichungen (2.6), (2.7) und (2.8) nach k_{t+1} gelöst. Es wird über b iteriert, indem nach dem aktuellen Kapital k_t , gegeben dem Interpolanten für das zukünftige Kapital $K(k_{t+1}, a_{t+1}; b)$, aufgelöst wird. Für die Interpolation muss für jede Iteration ein numerischer Löser verwendet werden, was hohe Kosten bei höherdimensionalen Problemen, wie dem Multi-Country-Modell bei einer großen Anzahl von Ländern, zur Folge hat.

Es ist sinnvoller bei hoher Dimension Fixpunktiteration zu verwenden, da hier nur simple Berechnungen genutzt werden.

Auch bei der Fixpunktiteration wird die Kapital-Policyfunktion mit einem Interpolationspolynom $k_{t+1} = K(k_t, a_t; b)$ parametrisiert. Zur Implementierung muss die Eulergleichung zur Fixpunkt-Eulergleichung

$$k_{t+1} = \beta E_t \left[\frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1} f'(k_{t+1})) k_{t+1} \right] \quad (2.13)$$

umgeschrieben werden, was möglich ist, da $u'(c_t) \neq 0$ und k_{t+1} t -messbar ist. Bei der Fixpunktiteration werden immer neue $k_{t+1} = K(\cdot; b)$ berechnet und in die rechte Seite eingesetzt. Daraus resultieren neue Werte \hat{k}_{t+1} für k_{t+1} auf der linken Seite. Bei der richtigen Lösung ist $\hat{k}_{t+1} = k_{t+1}$ und kann gekürzt werden. Dadurch entspricht die Lösung der von der ursprünglichen Eulergleichung. Die Iteration wird demzufolge so lange durchgeführt bis beide Seiten übereinstimmen. Das resultierende System sieht wie folgt aus:

$$k_{t+1} = K(k_t, a_t; b) \quad \text{und} \quad k_{t+2} = K(k_{t+1}, a_{t+1}; b) \quad (2.14)$$

$$c_t = (1 - \delta)k_t + a_t f(k_t) - k_{t+1} \quad (2.15)$$

$$c_{t+1} = (1 - \delta)k_{t+1} + a_{t+1}f(k_{t+1}) - k_{t+2} \quad (2.16)$$

Die berechneten Werte werden, wie oben beschrieben, in folgende Gleichung eingesetzt.

$$\hat{k}_{t+1} = \beta E_t \left[\frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1}f'(k_{t+1}))k_{t+1} \right] \quad (2.17)$$

Hierbei ist zu beachten, dass für c_{t+1} und dadurch auch für k_{t+2} je nach Quadraturmethode mehrere Realisierungen berechnet werden müssen, da diese vom autoregressiven Prozess (2.7) abhängen. In einem rekursiven Gleichgewicht sind die Entscheidungen der Periode t Funktionen des Zustands (k_t, a_t) . Das Ziel ist es, die Policyfunktionen für das Kapital $k_{t+1} = K(k_t, a_t)$ und damit aufgrund der Nebenbedingung (2.6) indirekt für den Konsum $c_t = C(k_t, a_t)$ zu finden, welche die Gleichungen (2.6), (2.7) und (2.8) erfüllen.

Wie im repräsentativen Agenten-Wachstums-Modell werden beim Multi-Country-Modell die Kapital-Policyfunktionen $K^h(\{k_t^h, a_t^h\}^{h=1, \dots, N})$, $h = 1, \dots, N$ gesucht. Sie werden mit landesspezifischen Approximationen $\hat{K}^h(\{k_t^h, a_t^h\}^{h=1, \dots, N}; b)$, $h = 1, \dots, N$ angenähert. Dafür muss aus dem Modell die Fixpunkt-Eulergleichung hergeleitet werden, welche laut [JMMV14]

$$k_{t+1}^h = E_t \left[\beta \frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1}^h A f'(k_{t+1}^h)) k_{t+1}^h \right], \quad h = 1, \dots, N \quad (2.18)$$

ist.

Kapitel 3

Dünne Gitter

In diesem Kapitel wird die Konstruktion von dünnen Gittern beschrieben. Außerdem wird gezeigt, wie das Interpolationspolynom generiert und die Koeffizienten berechnet werden. Die Beschreibungen folgen [JMMV14].

Die Anwendung von dünnen Gittern bei dynamischen Wirtschaftsmodellen ist sinnvoll, da oftmals viele Zustandsvariablen Einfluss auf die Entscheidungen haben. Bei d Variablen und je s Punkten pro Variable werden beim Tensorproduktgitter s^d Gitterpunkte benutzt. Das kann offensichtlich bei großem d sehr teuer werden.

Um den hohen Kosten bei vielen Dimensionen entgegenzuwirken, wurde von Smolyak 1963 ([Smo63]) eine Methode entwickelt, welche die gleiche Genauigkeit bei weniger Gitterpunkten erreicht. Die Idee der Methode besteht darin, dass bestimmte Punkte potentiell wichtiger für die Interpolation sind als andere. Diese Wichtigkeit wird mit sogenannten (Approximations-)Level μ beschrieben. Bei ansteigendem Level werden mehr Punkte hinzugefügt und die Genauigkeit der Approximation steigt.

In der Literatur wird das exponentielle Ansteigen der Anzahl der Gitterpunkte mit der Dimension als Fluch der Dimensionen ([Bel61]) bezeichnet. Diesem Fluch sind dünne Gitter nicht ausgesetzt, denn hier steigt die Anzahl der Gitterpunkte polynomiell mit der Dimension. Nach [BNR00] ist die Genauigkeit der Approximation mit dünnen Gittern abhängig von der Glattheit der zu interpolierenden Funktion f . Genauer gesagt gilt für den Fehler im Raum

$$F_d^k = \{f : [-1,1] \rightarrow \mathbb{R} \mid D^\alpha f \text{ stetig für } \alpha_i \leq k \text{ für alle } i\},$$

dass

$$\|f - \tilde{f}\| \leq c_{d,k} \cdot m^{-k} \cdot \log(m)^{(k+2)(d-1)+1}, \quad (3.1)$$

wobei \tilde{f} die Approximation der Funktion f , $c_{d,k}$ eine von d und k abhängige Konstante und m die Anzahl der Gitterpunkte ist. Die Glattheit der zu interpolierenden Funktion in den betrachteten Modellen hängt von der Wahl der modellierenden Funktionen ab.

Zur Interpolation auf dünnen Gittern werden in dieser Arbeit globale Polynomfunktionen genutzt. Sie haben den Vorteil, dass der Interpolant glatt und

differenzierbar ist und die Interpolation nicht dem Fluch der Dimensionen unterliegt.

3.1 Konstruktion eines dünnen Gitters

Zur Konstruktion eines dünnen Gitters werden Mengen S_i , $i = 1, 2, \dots$ von eindimensionalen Gitterpunkten generiert. Aus diesen Mengen werden Tensorprodukte und damit mehrdimensionale Gitterpunkte aufgebaut. Nach der später folgenden Smolyakregel (3.6) wird danach eine Teilmenge aus diesen Punkten ausgewählt.

Zur Generierung der eindimensionalen Gitterpunkte werden in dieser Arbeit Extrema von Chebyshev-Polynomen genutzt. Nach [BNR00] ist der Exaktheitsgrad dieser Punkte, welche auch als Chebyshev-Gauß-Lobatto oder Clenshaw-Curtis Punkte bekannt sind, fast optimal. Sie werden durch das Auswählen bestimmter Extrema der Chebyshev-Polynome verschiedenen Grades bestimmt. Diese Auswahl folgt zwei Regeln:

1. Eine Menge S_i , $i = 1, 2, \dots$ hat $m(i) = 2^{i-1} + 1$ Punkte für $i \geq 2$ und $m(1) \equiv 1$.
2. Jede nachfolgende Menge S_i enthält die Punkte der vorherigen Menge S_{i-1} . Die Mengen werden als geschachtelt bezeichnet.

Chebyshev-Polynome sind im Intervall $[-1, 1]$ definiert und unterliegen der Rekursion

$$\begin{aligned} T_0(x) &= 1, \\ T_1(x) &= x, \\ T_n(x) &= 2xT_{n-1}(x) - T_{n-2}(x) \text{ für } n \geq 2. \end{aligned} \tag{3.2}$$

Bei einem Polynomgrad von $n-1$ liegen n Extrema q_j , $j = 1, \dots, n$ vor, welche mit

$$q_j = -\cos\left(\frac{\pi(j-1)}{n-1}\right), \quad j = 1, \dots, n \tag{3.3}$$

für $n \geq 2$ berechenbar sind, wobei das Extremum von $T_0(x) = 1$ als 0 angenommen wird. Aufgrund der beiden Regeln können nicht einfach Extrema aufeinander folgender Chebyshev-Polynome gewählt werden. So wird mit 0 gestartet und der Polynomgrad iterativ erhöht, wobei die Mengen von Extrema ausgewählt werden, die die zuvor ausgewählten Extrema beinhalten.

Da, wie schon erwähnt, das Tensorprodukt der eindimensionalen Mengen der unterschiedlichen Dimensionen genommen wird, ist es nicht nötig, dass pro Dimension eindimensionale Punkte mehrmals vorliegen. Dies ist allerdings nach obiger Konstruktion der Fall, da jede Menge S_i die Punkte der Menge S_{i-1} beinhaltet.

Es ist möglich, die gleichen mehrdimensionalen Gitterpunkte durch Mengen zu konstruieren, die in der jeweiligen Dimension disjunkt untereinander sind. Die

Idee ist, die Punkte, welche in der vorherigen Menge von eindimensionalen Gitterpunkten enthalten sind, nicht mit in die neue Menge aufzunehmen. Formal beschrieben bedeutet dies

$$A_1 = S_1 \text{ und } A_i = S_i \setminus S_{i-1} \text{ für } i \geq 2. \quad (3.4)$$

Offensichtlich sind die Mengen A_i disjunkt nach Konstruktion. Die Anzahl der Elemente in der Menge S_i entspricht der Anzahl der Elemente in S_i abzüglich der Anzahl der Elemente in S_{i-1} , das heißt

$$2^{i-1} - 2^{i-2} = 2^{i-2}(2 - 1) = 2^{i-2} \text{ für } i \geq 3 \quad (3.5)$$

Elemente, wobei in A_1 ein und in A_2 zwei Elemente sind.

Aus den, mit dem Tensorprodukt der disjunkten eindimensionalen Mengen, konstruierten mehrdimensionalen Gitterpunkten werden diejenigen ausgewählt, die der Smolyakregel

$$\sum_{j=1}^d i_j \leq d + \mu \quad (3.6)$$

genügen. Offensichtlich sind die mehrdimensionalen Gitter $H^{d,\mu}$ für feste Dimension d und fortlaufendes Level μ , trotz der disjunkten eindimensionalen Mengen, geschachtelt. Das heißt

$$H^{d,0} \subset H^{d,1} \subset H^{d,2} \subset .. \quad (3.7)$$

$A_{i_1} \setminus A_{i_2}$		$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
		0	-1,1	$\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}$
$i_1 = 1$	0	(0,0)	(0, -1), (0,1)	$(0, \frac{-1}{\sqrt{2}}), (0, \frac{1}{\sqrt{2}})$
$i_1 = 2$	-1	(-1,0)	(-1, -1), (-1,1)	$(-1, \frac{-1}{\sqrt{2}}), (-1, \frac{1}{\sqrt{2}})$
	1	(1,0)	(1, -1), (1,1)	$(1, \frac{-1}{\sqrt{2}}), (1, \frac{1}{\sqrt{2}})$
$i_1 = 3$	$\frac{-1}{\sqrt{2}}$	$(\frac{-1}{\sqrt{2}}, 0)$	$(\frac{-1}{\sqrt{2}}, -1), (\frac{-1}{\sqrt{2}}, 1)$	$(\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}), (\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$
	$\frac{1}{\sqrt{2}}$	$(\frac{1}{\sqrt{2}}, 0)$	$(\frac{1}{\sqrt{2}}, -1), (\frac{1}{\sqrt{2}}, 1)$	$(\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}), (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$

Tabelle 3.1: Tensorprodukt der eindimensionalen Mengen (d=2)

Im Tabelle 3.1 werden die Chebyshev-Gitterpunkte im zweidimensionalen Fall dargestellt. Dabei entspricht das hellste Grau dem Gitterpunkt des Levels null. Für jedes weitere Level werden die Punkte des nächst-dunkleren Grautons

dazugenommen, wobei die Konstruktion für $\mu \geq 3$ fortgesetzt werden muss.

Nachfolgend werden die Gitterpunkte im zweidimensionalen Fall für die Dünn-Gitter-Level $\mu = 1, 2, 3$ anschaulich dargestellt.

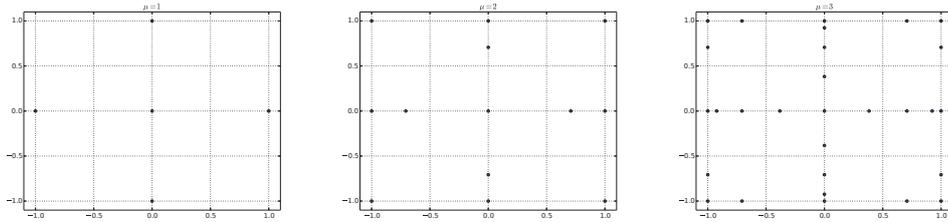


Abbildung 3.1: Dünne Gitter im Zweidimensionalen bei verschiedenen Level

Es gibt noch andere Regeln zur Konstruktion von geschachtelten eindimensionalen Mengen, welche sich in der Wahl und der Anzahl der Gitterpunkte unterscheiden. Des Weiteren existieren auch dünne Gitter, welche keine geschachtelte Struktur nutzen.

3.2 Interpolationspolynom

Betrachtet sei folgendes Interpolationsproblem bei dem $f : [-1,1]^d \rightarrow \mathbb{R}$ eine stetige Funktion sei, welche auf dem d -dimensionalen Hyperwürfel definiert ist. $\Psi_n : [-1,1]^d \rightarrow \mathbb{R}$, $n=1, \dots, M$ seien d -dimensionale Basisfunktionen und $b = (b_1, \dots, b_M)$ der Koeffizientenvektor. Dann kann das Interpolationspolynom $\hat{f}(\cdot; b)$ definiert werden als

$$\hat{f}(x; b) = \sum_{n=1}^M b_n \Psi_n(x), \quad (3.8)$$

wobei $x \in [-1,1]^d$.

Zunächst werden M Gitterpunkte $x_1, \dots, x_M \in [-1,1]^d$ konstruiert. Für diese Arbeit werden die im vorherigen Abschnitt beschriebenen Chebyshev-Gitterpunkte gewählt.

Danach müssen M Basisfunktionen gewählt werden. Dafür gibt es eine Vielzahl von möglichen Optionen. Im Rahmen dieser Arbeit werden Chebyshev Basisfunktionen gewählt. Sie haben den Vorteil, dass beim Lagrangeansatz seltener numerische Instabilitäten auftreten. Mehr Probleme gäbe es zum Beispiel bei den ordinären Polynomen $O_m(x) = x^m$, $m = 0, 1, \dots$, da die Basisfunktionen stark korreliert sind. Die Chebyshev-Basisfunktionen sind hingegen orthogonal bezüglich des gewichteten Skalarprodukts $\int_{-1}^1 f(x)g(x) \frac{1}{\sqrt{1-x^2}} dx$ und somit weniger stark korreliert. Für weitere Informationen siehe [JMM09]. Die Konstruktion der mehrdimensionalen Basisfunktionen folgt im nächsten Abschnitt. Ziel ist es einen Koeffizientenvektor $b = (b_1, \dots, b_M)$ zu finden, sodass die Funktionsauswertungen der Approximation $\hat{f}(\cdot; b)$ in allen Gitterpunkten mit der ei-

gentlichen Funktion f übereinstimmt.

Mit der Darstellung der Funktion durch das Interpolationspolynom können die Funktionswerte in jedem Punkt $x \in [-1,1]^d$ ausgewertet werden.

3.2.1 Basisfunktionen des Interpolationspolynoms

Die Konstruktion der Basisfunktionen ist sehr ähnlich zu der der Gitterpunkte. Es werden Mengen B_i von eindimensionalen Chebyshev Basisfunktionen $T_i(x)$, $i \geq 0$ generiert. Die Größe der eindimensionalen Mengen entspricht der Anzahl der eindimensionalen Gitterpunkte in den jeweiligen disjunkten Mengen A_i . Aus den eindimensionalen Basisfunktionen wird das Produkt genommen und mit der Smolaykregel (3.6) werden die passenden mehrdimensionalen Basisfunktionen ausgewählt. Offensichtlich ist die Anzahl der Gitterpunkte gleich der Anzahl der Basisfunktionen und somit auch gleich der Anzahl der Koeffizienten im Interpolationspolynom.

Für andere Basisfunktionen geht die Konstruktion analog.

$B_{i_1} \setminus B_{i_2}$		$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
		$T_0(y)$	$T_1(y), T_2(y)$	$T_3(y), T_4(y)$
$i_1 = 1$	$T_0(x)$	$T_0(x)$	$T_1(y), T_2(y)$	$T_3(y), T_4(y)$
$i_1 = 2$	$T_1(x)$	$T_1(x)$	$T_1(x)T_1(y), T_1(x)T_2(y)$	$T_1(x)T_3(y), T_1(x)T_4(y)$
	$T_2(x)$	$T_2(x)$	$T_2(x)T_1(y), T_2(x)T_2(y)$	$T_2(x)T_3(y), T_2(x)T_4(y)$
$i_1 = 3$	$T_3(x)$	$T_3(x)$	$T_3(x)T_1(y), T_3(x)T_2(y)$	$T_3(x)T_3(y), T_3(x)T_4(y)$
	$T_4(x)$	$T_4(x)$	$T_4(x)T_1(y), T_4(x)T_2(y)$	$T_4(x)T_3(y), T_4(x)T_4(y)$

Tabelle 3.2: Mehrdimensionale Chebyshev-Basisfunktionen (d=2)

Im Beispieletabelle 3.2 wird der zweidimensionalen Fall dargestellt. Das hellste Grau entspricht der Chebyshev-Basisfunktion des Levels null. Für jedes weitere Level werden die Basisfunktionen des nächst-dunkleren Grautons dazugenommen, wobei die Konstruktion für $\mu \geq 3$ fortgesetzt werden muss. Diese sind jeweils das Produkt der eindimensionalen Basisfunktionen aus den Mengen B_i . Zu beachten ist noch, dass der Übersichtlichkeit halber bei Produkten mit $T_0(x) = T_0(y) = 1$ direkt das Ergebnis eingetragen wurde.

3.2.2 Koeffizienten des Interpolationspolynoms, Lagrangeansatz

Das Interpolationsproblem kann auch in Matrixschreibweise dargestellt werden.

$$\begin{pmatrix} f(x_1) \\ \dots \\ f(x_m) \end{pmatrix} = \begin{pmatrix} \hat{f}(x_1; b) \\ \dots \\ \hat{f}(x_M; b) \end{pmatrix} = \begin{pmatrix} \Psi_1(x_1) & \dots & \Psi_M(x_1) \\ \dots & \dots & \dots \\ \Psi_1(x_M) & \dots & \Psi_M(x_M) \end{pmatrix} \begin{pmatrix} b_1 \\ \dots \\ b_M \end{pmatrix} \quad (3.9)$$

Wenn die Matrix der ausgewerteten Basisfunktionen vollen Rang hat, so sind die Koeffizienten eindeutig und können mithilfe der Inversen der ausgewerteten Basismatrix direkt durch ein Matrixvektorprodukt berechnet werden.

3.3 Anisotrope dünne Gitter

In vielen Anwendungsbereichen, insbesondere in ökonomischen Modellen, sind einige Variablen wichtiger als andere. So kann beispielsweise für ein Land das eigene Kapital wichtiger sein, als das eines anderen Landes. Funktionen, die dies beschreiben, werden asymmetrisch genannt. Es ist dementsprechend sinnvoll, die Variablen die mehr Anteil an der Lösung haben, genauer zu betrachten als solche mit wenig Ausschlag. Das heißt, es ist von Vorteil für asymmetrische Funktionen auch asymmetrische Gitter zu verwenden. Die bisher betrachtete Version von dünnen Gittern erzeugt allerdings gleich viele Punkte in jeder Dimension. Gitter bei denen in mindestens einer Dimension ein anderes Level verwendet wird, als in einer anderen, werden anisotrop genannt. Ziel ist es, anisotrope Gitter so zu konstruieren, dass das Level bei den Variablen, von denen die Lösung mehr abhängt, höher gewählt wird, als bei den Anderen. Bei bekannten Funktionen können die Level so angepasst werden, dass die wichtigsten Variablen genauer betrachtet werden. Bei unbekanntem Funktionen können einzelne Level erhöht werden und nachfolgend kann getestet werden ob die Erhöhung der Genauigkeit den Anstieg der Kosten wert ist. Diese Art von dünnen Gittern ist nützlich, da es oft zu teuer ist für eine größere Genauigkeit das Level in allen Dimensionen zu erhöhen.

3.3.1 Konstruktion von anisotropen dünnen Gittern

In diesem Abschnitt wird die Konstruktion eines anisotropen dünnen Gitters nach [JMMV14] beschrieben, welche sich auf diejenige der isotropen dünnen Gitter stützt.

Es werden analog eindimensionale disjunkte Mengen von Gitterpunkten konstruiert und darüber das Tensorprodukt genommen. Der Unterschied liegt in der Auswahl der mehrdimensionalen Gitterpunkte.

Im Abschnitt über isotrope dünne Gitter ist zu erkennen, dass für jede Dimension $j = 1, \dots, d$

$$i_j \leq \mu + 1 = i_j^{\max} \quad (3.10)$$

gilt.

Sei für den anisotropen Fall mit $\mu_j = i_j^{\max} - 1$ das Level in der jeweiligen Dimension j bezeichnet. Das Gitter heißt anisotrop, wenn ein μ_k existiert mit

$\mu_k \neq \mu_l$ für ein $l \neq k$, wobei $k, l = 1, \dots, d$.

Definiere $\mu_{\max} = \max\{\mu_1, \dots, \mu_d\}$ zur Darstellung der modifizierten Smolyakregeln:

$$\sum_{j=1}^d i_j \leq \mu_{\max} + d \quad (3.11)$$

$$i_j \leq \mu_j + 1, \quad j = 1, \dots, d \quad (3.12)$$

Die Konstruktion und Berechnung der Koeffizienten im anisotropen Fall ist analog zum isotropen Fall, nur dass hier die modifizierten Smolyakregeln befolgt werden.

Nachfolgend werden die Gitterpunkte im zweidimensionalen Fall für die Level $\mu = (2,1)$ und $\mu = (2,3)$ anschaulich dargestellt.

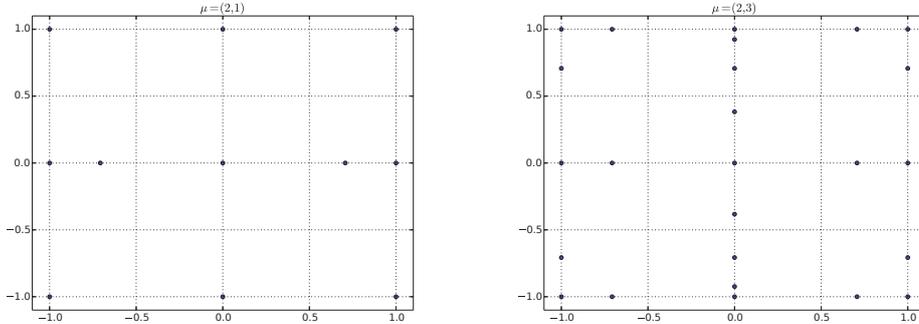


Abbildung 3.2: Anisotrope dünne Gitter im Zweidimensionalen

3.4 HCFFT

Die HCFFT (Hyperbolic Cross Fast Fourier Transformation)-Bibliothek wird in dieser Arbeit genutzt um dünne Gitter zu konstruieren. Um die in [JMMV14] benutzten Chebyshev-Gitterpunkte zu verwenden wurde eine leichte Modifikation an den in der Bibliothek vorhandenen Chebyshev-Gitterpunkten vorgenommen. Durch das Setzen der modifizierten Smolyakregel in den Zusatzparametern ist es möglich anisotrope Gitter zu konstruieren.

Zudem kann die Bibliothek die Funktionswerte des Interpolationspolynoms mithilfe der Koeffizienten an beliebigen Punkten auswerten und ermöglicht das Berechnen der Koeffizienten bei vorhandenen Funktionswerten an bestimmten Punkten.

Die Bibliothek basiert unter anderem auf [GH14].

3.5 Vorberechnung

Um die Laufzeit des in Kapitel 5 beschriebenen Algorithmus zu verbessern, kann nach [JMMV14] das Prinzip der Vorberechnung genutzt werden.

Sei für die folgende Beschreibung mit X die Matrix der in den Gitterpunkten

ausgewerteten Basisfunktionen bezeichnet, welche in (3.9) beschrieben wurde. Im Algorithmus wird in jeder Iteration das Interpolationspolynom auf den Gitterpunkten ausgewertet. Dies kann somit als Matrixvektormultiplikation mit der immer gleichen Matrix X aufgefasst werden.

Es kann Laufzeit gespart werden, wenn diese Matrix vor dem iterativen Teil berechnet wird und immer zur Berechnung verwendet wird. Des Weiteren kann direkt die Inverse von X berechnet werden, da dies die direkte Berechnung der Koeffizienten durch ein Matrixvektorprodukt erlaubt.

Kapitel 4

Numerische Utensilien

4.1 Ergodische Menge

Mit der ergodischen Menge wird der Support der ergodischen Verteilung beschrieben. Diese kann bei einem dynamischen Wirtschaftsmodell jede Art von Form im d -dimensionalen Raum haben und sogar unbeschränkt sein.

Der in Kapitel 5.1 vorgestellte Algorithmus verwendet Simulation. Mit dieser kann herausgefunden werden, welche Zustände mit hoher Wahrscheinlichkeit auftreten und somit relevant sind.

Um die Daten auf dem relevanten Zustandsraum zu betrachten, muss dieser zunächst gefunden werden. Hierfür existieren mehrere Ansätze, welche Simulation benutzen.

Der triviale Ansatz für eine Abbildung in diesen Raum ist es, einen Hyperwürfel zu konstruieren, der in jeder Dimension vom kleinsten bis zum größten Element der Simulation geht ([JMMV14]). Das Finden eines solchen Hyperwürfels ermöglicht es eine Abbildung zu definieren, die die Zustände $(k_t, a_t) \in \mathbb{R}_+^d$ in $[-1,1]^d$ abbildet. Das ist nützlich, da die Chebyshev-Polynome, welche zur Interpolation auf dem dünnen Gitter verwendet werden, nur im dem Intervall $[-1,1]^d$ definiert sind.

Seien $\{(k_t, a_t)_{t=1, \dots, T}\}$ die simulierten Zustände, wobei $k_t, a_t \in \mathbb{R}_+^N$ jeweils die Zustände des Agenten beziehungsweise aller Länder der Periode t sind. Bezeichne mit s_i das kleinste und mit b_i das größte Element der jeweiligen Dimension $i \in \{1, 2, \dots, 2N\}$. Durch die lineare Transformation

$$\tilde{x}_i = 2 \frac{x_i - s_i}{b_i - s_i} - 1 \tag{4.1}$$

kann eine Zustandsvariable $x_i \in [s_i, b_i]$ in $\tilde{x}_i \in [-1,1]$ abgebildet werden. Durch das Verwenden dieser Transformation für alle Dimensionen i ergibt sich eine Abbildung die $(k, a) \in (s_1, b_1) \times (s_2, b_2) \times \dots \times (s_d, b_d) = (\underline{k}_1, \bar{k}_1) \times (\underline{k}_2, \bar{k}_2) \times \dots \times (\underline{k}_N, \bar{k}_N) \times (\underline{a}_1, \bar{a}_1) \times \dots \times (\underline{a}_N, \bar{a}_N)$ in $[-1,1]^d$ abbildet, wobei \underline{k}_h und \underline{a}_h jeweils das kleinste Kapital beziehungsweise die kleinste Produktivität und \bar{k}_h und \bar{a}_h jeweils das größte Kapital beziehungsweise die größte Produktivität des Landes $h \in \{1, \dots, N\}$ darstellt.

Die Inverse Abbildung folgt direkt durch Umformen von (4.1) in

$$x_i = \frac{(\tilde{x}_i + 1)(b_i - s_i)}{2} + s_i \quad (4.2)$$

und analoges Vorgehen.

Diese Methode hat den Nachteil, dass zum Beispiel bei einer oval-förmigen Punktverteilung sehr viel Rechenaufwand in irrelevante Zustände gesteckt wird. Alternativ kann der Principal-Components-Ansatz ([JMM10]) verfolgt werden. Hier werden alle Werte so transformiert, dass der Erwartungswert bei null liegt und die Standardabweichung gleich eins ist. Dies hat den Vorteil, dass die betrachteten Punkte besser im konstruierten Hyperwürfel verteilt sind. Im zweidimensionalen heißt das kreisförmig. Dementsprechend wird im Verhältnis zum obigen Ansatz weniger Aufwand in irrelevante Zustände gesteckt.

4.2 Numerische Integrationsmethoden

Da das betrachtete Modell auf stochastischen Ausschlägen basiert, wird der Erwartungswert betrachtet. Um diesen zu berechnen, werden in dieser Arbeit numerische Integrationsmethoden verwendet. Hierbei wird zwischen simulationsbasierten und deterministischen Quadraturmethoden unterschieden.

Zur Fehler- und Konvergenzratenanalyse sei der absolute Integrationsfehler $\varepsilon_t^I = |y_t - E_t[\cdot]|$ betrachtet, wobei E_t den genauen Wert und y_t die Approximation der bedingten Erwartung darstellt.

Die folgenden Abschnitte orientieren sich an [JMM09].

4.2.1 Monte-Carlo-Integration

Bei der Monte-Carlo-Quadratur wird das Integral mithilfe von Realisierungen von Zufallszahlen berechnet. Das heißt, im Gegensatz zu den in 4.2.2 bis 4.2.3 beschriebenen deterministischen Quadraturmethoden wird der Integrand nicht auf bestimmten Knoten ausgewertet, sondern auf zufälligen Punkten im Integrationsbereich.

Die Monte-Carlo-Quadratur mit einem Knoten wurde zum Beispiel in [Mar88] verwendet. Hierbei wird das Integral mit der Realisierung der nächsten Periode approximiert. Im betrachteten Modell heißt das, dass nur ein Schock bezüglich der Normalverteilung generiert wird. Mit $\epsilon_{t+1,1} \equiv \epsilon_{t+1}$ und $\omega_{t,1} = 1$ ergibt sich die Approximation des bedingten Erwartungswerts (2.13) zu

$$y_t = \beta \frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1} f'(k_{t+1})) k_{t+1} \quad (4.3)$$

Bei der Monte-Carlo-Quadratur mit J Knoten werden J Realisierungen $\{\epsilon_{t+1,j}\}_{j=1,\dots,J}$ genommen, welche unabhängig von den bisherigen Schocks sind. Den verschiedenen Auswertungen wird jeweils das gleiche Gewicht $\omega_{t,j} = \frac{1}{J}$ für alle t und j zugewiesen. Daraus ergibt sich die Approximation des bedingten Erwartungswerts (2.13) zu

$$y_t = \sum_{j=1}^J \frac{1}{J} \beta \frac{u'(c_{t+1,j})}{u'(c_t)} (1 - \delta + a_{t+1,j} f'(k_{t+1})) k_{t+1} \quad (4.4)$$

Die mehrknotige Monte-Carlo-Quadraturmethode wird im Algorithmus nicht benutzt und ist hier nur der Vollständigkeit wegen erklärt, da sie alternativ eingebunden werden kann.

Ein Vorteil liegt in der leichten Implementierung, besonders im Vergleich zu deterministischen Quadraturmethoden bei höherdimensionalen Integralen. Ein weiterer großer Vorteil der Monte-Carlo-Quadratur liegt vor allem darin, dass die Kosten unabhängig von der Anzahl der Dimensionen ist. Die Konvergenzrate hängt für beschränkte Varianz von der Anzahl der Simulationen T ab. Es gilt

$$\varepsilon_t^I = \mathcal{O}(T^{-\frac{1}{2}}) = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right), \quad (4.5)$$

was direkt aus dem Gesetz der großen Zahlen folgt.

Ein großer Nachteil des Verfahrens ist, dass für eine hohe Genauigkeit extrem viele Perioden simuliert werden müssen, was in hohen Kosten resultiert.

4.2.2 Ein-dimensionale Gauß-Hermite-Quadratur

Deterministische Quadraturmethoden benutzen keine Simulation. Im repräsentativen Agentenwachstumsmodell sind die Zufallsvariablen normalverteilt. Hier ist es möglich die eindimensionalen Integrale mithilfe der Gauß-Hermite-Quadratur zu berechnen. Bei dieser Integrationsmethode werden zur Berechnung von (2.13) deterministisch bestimmte Knoten und Gewichte verwendet. Die ein-knotige Gauß-Hermite-Quadratur ist dabei ein Spezialfall, da $\epsilon_{t+1,1} = 0$ und $\omega_{t,1} = 1$ ist. Das heißt, dass der Schock null ist und die Produktivität der nächsten Periode nur von der aktuellen Produktivität abhängt. Formal bedeutet das

$$a_{t+1} = a_t^p \exp(0) = a_t^p. \quad (4.6)$$

Die genaue Berechnung folgt als Spezialfall der mehrdimensionale Quadratur im nachfolgenden Abschnitt.

4.2.3 Mehr-dimensionale deterministische Quadratur

In diesem Abschnitt wird diskutiert, wie bei mehreren exogen gegebenen Zufallsvariablen mithilfe von deterministischen Quadraturmethoden integriert wird. Das ist zum Beispiel beim Multi-Country-Modell notwendig, bei dem nicht nur ein Agent Produktivitätsschocks unterliegt, sondern mehrere Länder.

Ziel ist es, mehrdimensionale Integrale der Form

$$\int_{\mathbb{R}^N} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon \quad (4.7)$$

zu berechnen, wobei $\epsilon \equiv (\epsilon^1, \dots, \epsilon^N)^T \in \mathbb{R}^N$. Hierbei ist $\epsilon \sim \mathcal{N}(\mu, \Sigma)$ normalverteilt, wobei $\mu = (\mu^1, \dots, \mu^N)^T \in \mathbb{R}^N$ der Vektor der Erwartungswerte und $\Sigma \in \mathbb{R}^{N \times N}$ die Kovarianzmatrix ist. Weiter ist

$$\mathcal{W}(\epsilon) = (2\pi)^{-N/2} \det(\Sigma)^{-1/2} \exp\left(-\frac{1}{2}(\epsilon - \mu)^T \Sigma^{-1}(\epsilon - \mu)\right). \quad (4.8)$$

die Dichtefunktion der multivariaten Normalverteilung.

Die deterministischen Quadraturformeln werden unter der Annahme konstruiert, dass die Zufallsvariablen unkorreliert sind und Erwartungswert null, sowie Varianz eins haben. Aus diesem Grund muss im Fall von korrelierten Zufallsvariablen zunächst das Integral umgeschrieben werden. Hierfür wird die Cholesky-Zerlegung genutzt.

Für symmetrisches und positiv definites Σ existiert die Cholesky-Zerlegung $\Sigma = LL^T$, wobei L eine untere Dreiecksmatrix mit strikt positiven Diagonaleinträgen ist.

Mit der Cholesky-Zerlegung von Σ ist es durch den linearen Variablenwechsel

$$\nu = \frac{L^{-1}(\epsilon - \mu)}{\sqrt{2}} \quad (4.9)$$

möglich, die korrelierten Zufallszahlen ϵ in unkorrelierte Variablen ν zu transformieren. Es folgt $d\epsilon = 2^{N/2} \det(L) d\nu$ und offensichtlich ist $\Sigma^{-1} = (L^{-1})^T L^{-1}$ und $\det(\Sigma) = \det(L)^2$. Daraus ergibt sich

$$\int_{\mathbb{R}^N} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon = \pi^{-N/2} \int_{\mathbb{R}^N} f(\sqrt{2}L\nu + \mu) \exp(-\nu^T \nu) d\nu. \quad (4.10)$$

Integrale werden bei deterministischen Integrationsmethoden durch eine gewichtete Summe von endlich vielen Auswertungen des Integranden an den Integrationsknoten $\{\nu_j\}_{j=1,\dots,J}$ bestimmt. Mit Integrationsgewichten $\{\omega_j\}_{j=1,\dots,J}$ folgt

$$\int_{\mathbb{R}^N} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon \approx \pi^{-N/2} \sum_{j=1}^J \omega_j f(\sqrt{2}L\nu_j + \mu). \quad (4.11)$$

Im Folgenden sei $\mu = 0_N$.

Mehrdimensionale Gauß-Hermite-Quadratur

Für den eindimensionalen Fall $N = 1$ folgt mit (4.11) und der Wahl der Gewichte $\{\omega_j\}_{j=1,\dots,J}$ und Knoten $\{\nu_j\}_{j=1,\dots,J}$ nach der Gauß-Hermite-Quadraturregel, welche in den zugehörigen Tabellen, beispielsweise in [Jud98], nachgesehen werden können.

$$\int_{\mathbb{R}} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon \approx \pi^{-1/2} \sum_{j=1}^J \omega_j f(\sqrt{2}L\nu_j) \quad (4.12)$$

Bei einer kleinen Anzahl von normalverteilten exogenen Zufallsvariablen können mehrdimensionale Integrale mit der Gauß-Hermite-Produktregel berechnet werden. Dann gilt

$$\int_{\mathbb{R}^N} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon \approx \pi^{-N/2} \sum_{j_1=1}^{J_1} \cdots \sum_{j_N=1}^{J_N} \omega_{j_1}^1 \cdots \omega_{j_N}^N f(\sqrt{2}L(\nu_{j_1}^1, \dots, \nu_{j_N}^N)^\top), \quad (4.13)$$

wobei $\{\omega_{j_h}\}_{j=1,\dots,J}$ und $\{\mu_{j_h}\}_{j=1,\dots,J}$ jeweils die Gewichte und Knoten der Dimension h sind. Bei dieser Methode werden mehrdimensionale Knoten als Tensorprodukte von eindimensionalen Knoten konstruiert. Dadurch wächst die Anzahl der Knoten bei J Knoten pro Dimension exponentiell mit der Anzahl N

der exogenen Zufallsvariablen. Das heißt, es werden J^N Knoten konstruiert. Dementsprechend ist diese, sowie jede andere Quadraturmethode, welche den Produktansatz verfolgt, nicht sinnvoll bei hoher Dimension.

Einen Spezialfall stellt die Gauß-Hermite-Quadratur mit nur einem Knoten dar, da hierbei, unabhängig von der Anzahl der Zufallsvariablen, nur ein Knoten gebraucht wird. Diese Quadraturmethode ist nicht so genau, wie die meisten Ansätze mit mehreren Knoten, allerdings ist sie sehr preiswert und somit aufgrund ihrer Dimensionsunabhängigkeit nützlich bei extrem großer Dimension. Eine Alternative liefern die im nächsten Abschnitt gezeigten Monomregeln, welche kein Tensorprodukt nutzen.

Monomregeln

Monomregeln haben den Vorteil, dass typischerweise die Zahl der Knoten polynomiell mit der Zahl der exogenen Zufallsvariablen wächst. In dieser Arbeit wird eine Monomregel verwendet welche $2N$ Knoten verwendet. Hierbei werden aufeinanderfolgende Abweichungen der Zufallsvariablen von ihrem Erwartungswert betrachtet, wobei jeweils die anderen Zufallsvariablen auf ihren Erwartungswert fixiert werden. Da die Kosten nur linear mit N wachsen, kann diese Methode auch in hoher Dimension eingesetzt werden.

Ziel ist es das Integral (4.10) mithilfe von (4.9) zu approximieren, sodass korrelierte Variablen auftreten dürfen. Nach [JMM09] lässt sich das Integral durch

$$\int_{\mathbb{R}^N} f(\epsilon) \mathcal{W}(\epsilon) d\epsilon \approx \frac{1}{2^N} \sum_{h=1}^N f(\pm R z^h) \quad (4.14)$$

approximieren, wobei $R \equiv \sqrt{N}L$ und z^h der Vektor ist, dessen h -ter Eintrag eins ist und dessen andere Elemente null sind.

In [JS11] wird Vergleich der Genauigkeit und Kosten von Produktregeln, Monomregeln und Dünn-Gitter-Integration durchgeführt. Bei der Dünn-Gitter-Integration wird Symmetrie ausgenutzt, sodass nur ein Teil der durch die Produktregel erzeugten Knoten genutzt wird, wobei die Gewichte angemessen angepasst werden. Daraus resultieren weitaus geringere Kosten und eine annähernd gleich gute Genauigkeit. Zudem sind die Knoten und Gewichte für eine höhere Genauigkeit leichter zu berechnen, als bei Monomregeln. Die niedrigsten Kosten hingegen werden mit Monomregeln und insbesondere mit der Beschriebenen erreicht, sodass diese für die Anwendung in hohen Dimension sehr sinnvoll ist.

Kapitel 5

Der Algorithmus

Der Algorithmus, welcher im Laufe dieser Arbeit implementiert wurde, setzt sich aus zwei Teilalgorithmen zusammen. Der erste Teil ist der GSSA (Generalized Stochastic Simulation Algorithm) aus [JMM09] und der zweite Teil, welcher dünne Gitter benutzt, wird in [JMMV14] beschrieben. Beide Algorithmen benutzen den oben beschriebenen Ansatz der Fixpunktiteration um die Kapital-Policyfunktion

$$k_{t+1} = K(k_t, a_t) \tag{5.1}$$

zu approximieren.

5.1 GSSA

Der GSSA wird benutzt, um eine Anfangsschätzung für die Koeffizienten des Interpolanten zu erhalten, sodass der Algorithmus aus [JMMV14] konvergiert. Des Weiteren liefert er durch Simulation die ergodische Menge, die es ermöglicht, Berechnungen nur im relevanten Zustandsraum durchzuführen. Dies steigert nach [JMM09] die Effizienz des Algorithmus enorm.

Um das beschriebene Modell zu lösen, wird eine flexible funktionale Form $\psi(k_t, a_t; b)$ benutzt und es wird ein Koeffizientenvektor b gesucht, sodass

$$K(k_t, a_t) \approx \psi(k_t, a_t; b) \tag{5.2}$$

für eine Menge von Punkten (k_t, a_t) im Zustandsraum gilt. In dieser Arbeit wird für ψ ein Polynom ersten Grades verwendet. Wie schon in Kapitel 1 erklärt, kann die Eulergleichung zu

$$k_{t+1} = \text{E}_t \left[\beta \frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1} f'(k_{t+1})) k_{t+1} \right] \tag{5.3}$$

umgeschrieben werden.

Dadurch wird k_{t+1} einmal als Wahl durch die Policyfunktion durch (5.1) zusammen mit (5.2) und außerdem als bedingte Erwartung zum Zeitpunkt $t+1$ durch (5.3) ausgedrückt. Durch diese Konstruktion ist es möglich $k_{t+1} = K(k_t, a_t)$ in die rechte Seite von (5.3) einzusetzen und somit die Policyfunktion als Fixpunkt auszudrücken. Durch die Berechnung der rechten Seite von (5.3) ergibt

sich $k_{t+1} = K(k_t, a_t)$ für alle (k_t, a_t) im betrachteten Zustandsraum.

Der GSSA verwendet stochastische Simulation, um eine Lösung der Eulergleichung (5.3) durch Fixpunktiteration zu finden.

Zunächst wird die Kapital-Policyfunktion geschätzt, danach wird die Lösung für die betrachteten Perioden simuliert und die bedingte Erwartung in jedem simulierten Punkt berechnet. Mithilfe der Daten wird die Schätzung der Policyfunktion verbessert. Dieser Vorgang wird wiederholt, bis ein Fixpunkt gefunden wird.

Algorithmus 1 GSSA

- 1: **Initialisierung:**
 - 2: Wähle eine Startschätzung für $b^{(1)}$ und damit für die Policyfunktion
 - 3: Wähle den Anfangszustand (k_0, a_0) für die Simulation
 - 4: Wähle die Simulationslänge T
 - 5: Realisiere Produktivitätsschocks $\{\epsilon_t\}_{t=1,\dots,T}$
 - 6: Berechne Produktivitäten $\{a_t\}_{t=1,\dots,T}$ nach (2.7)
 - 7: Sei $i = 0$ die jeweilige Iteration der while-Schleife
 - 8: **while** $\frac{1}{T} \sum_{t=1}^T \left| \frac{k_{t+1}^{(i)} - k_t^{(i-1)}}{k_{t+1}^{(i)}} \right| > \bar{\omega}\xi$ **do**
 - 9: **for** $t = 1, \dots, T$ **do** \triangleright Berechne das Kapital der Perioden $1, \dots, T$
 - 10: $k_{t+1} \leftarrow \psi(k_t, a_t; b^{(i)})$ \triangleright Berechne Kapital der nächsten Periode mit (5.2)
 - 11: **end for**
 - 12: **for** $t = 1, \dots, T$ **do** \triangleright Berechne den Konsum der Perioden $1, \dots, T$
 - 13: $c_t \leftarrow (1 - \delta)k_t + a_t f(k_t) - k_{t+1}$ \triangleright Berechne den Konsum der aktuellen Periode nach (2.15)
 - 14: **end for**
 - 15: **for** $t = 1, \dots, T-1$ **do** \triangleright Berechne die bedingte Erwartung in der Fixpunkt-Eulergleichung der Perioden $1, \dots, T-1$
 - 16: **for** $j = 1, \dots, J$ **do** \triangleright Berechne $k_{t+2,j}$ und $c_{t+1,j}$ für die Berechnung der bedingten Erwartung
 - 17: $k_{t+2,j} \leftarrow \psi(k_{t+1}, a_{t+1,j}; b^{(i)})$ \triangleright Berechne Kapital zwei Perioden später, wobei einmalig die Produktivität $a_{t+1,j}$ für die jeweiligen Knoten berechnet werden muss
 - 18: $c_{t+1,j} \leftarrow (1 - \delta)k_{t+1} + a_{t+1,j} f(k_{t+1}) - k_{t+2,j}$ \triangleright Berechne Konsum der folgenden Periode nach (2.16)
 - 19: **end for**
 - 20: $y_t \leftarrow \sum_{j=1}^J \omega_{t,j} \beta \frac{u'(c_{t+1,j})}{u'(c_t)} (1 - \delta + a_{t+1,j} f'(k_{t+1})) k_{t+1}$ \triangleright Berechne bedingte Erwartung in der Fixpunkt-Eulergleichung nach (5.3)
 - 21: **end for**
 - 22: $\hat{b} \leftarrow$ Fehler-minimierendes b in $y_t = \psi(k_t, a_t; b) + \epsilon_t$ bzgl. $\|\cdot\|_2$ für $t = 1, \dots, T-1$, wobei ϵ_t der Fehler in Periode t ist
 - 23: $b^{(i+1)} \leftarrow (1 - \xi)b^{(i)} + \xi\hat{b}$ \triangleright Update die Koeffizienten
 - 24: $i \leftarrow i + 1$
 - 25: **end while**
-

Der GSSA ist in [JMM09] bewusst allgemein gehalten, da mehrere Varianten existieren, die für verschiedene Anwendungen sinnvoll sind.

Initialisierung

Nachfolgend wird die Wahl der Parameter und Methoden beschrieben, die benutzt werden, um eine Anfangsschätzung der Koeffizienten des Dünn-Gitter-Interpolationspolynoms für den zweiten Teil des Algorithmus zu erhalten. Das repräsentative Agenten-Modell wird, wie in [JMM09], um einen Normalisierungsfaktor

$$A = \frac{1 - \beta + \beta\delta}{\alpha\beta} \quad (5.4)$$

in (2.6) zu

$$c_t + k_{t+1} = (1 - \delta)k_t + a_t Af(k_t) \quad (5.5)$$

erweitert. Mit dieser Normalisierung ist der Kapital-Gleichgewichtszustand \bar{k} gleich eins. Der Produktivitäts-Gleichgewichtszustand \bar{a} ist, wie direkt aus (2.7) ersichtlich, ebenfalls gleich eins. Des Weiteren wird für die Startschätzung $b^{(1)}$ der Ansatz aus [JMMV14] verwendet, dass $k_{t+1} = 0,9k_t + 0,1a_t$ angenommen wird.

Der Anfangszustand (k_0, a_0) wird auf $(\bar{k}, \bar{a}) = (1, 1)$ gesetzt. Es ist zu erkennen, dass jetzt $k_1 = 0,9 \cdot 1 + 0,1 \cdot 1 = 1 = \bar{k}$ gilt. Diese Schätzung der Koeffizienten entspricht dementsprechend dem Gleichgewichtszustand des Kapitals.

Als Simulationslänge wird $T = 10000$ verwendet. Diese Länge hat gegenüber einer kleineren Simulationslänge den Vorteil, dass bei der Bestimmung von \hat{b} mit der Methode der kleinsten Quadrate bei der Berechnung des Schätzers keine Probleme bei der Inversenberechnung auftreten.

Die in Zeile 5 bezüglich der Normalverteilung für alle betrachteten Perioden generierten Zufallsvariablen werden zur iterativen Berechnung der Produktivität in Zeile 6 nach (2.7) genutzt.

Iterativer Teil

In den Zeilen 9-11 wird iterativ mithilfe der simulierten Produktivitäten a_t , $t = 1, \dots, T$ und der Approximationsfunktion ψ das Kapital k_{t+1} der jeweils nächsten Periode berechnet.

Der Konsum c_t der T Perioden wird in den Zeilen 12-14 nach (2.15) bestimmt, wobei die Zustände a_t und k_t , sowie das zuvor berechnete Kapital k_{t+1} genutzt werden.

Für den bedingten Erwartungswert in (5.3) lässt sich mit einer Quadraturregel eine Approximation y_t berechnen. Es ist möglich deterministische, sowie simulationsbasierte Methoden zu verwenden. In dieser Arbeit wird die billige ein-Knoten Monte-Carlo-Quadratur verwendet, da der GSSA hier nur für den Anfangswert der Koeffizienten im den Dünn-Gitter-Algorithmus benutzt wird. Durch die Wahl dieser Quadraturmethode fallen die Berechnungen in den Zeilen 16-19 weg, da für den Konsum c_{t+1} der nachfolgenden Periode der in den Zeilen 12-14 berechnete Konsum verwendet wird. Die Schleife geht nur bis $T-1$, da kein Wert für den Konsum der Folgeperiode von Periode T existiert.

Um den Fehler in der Regressionsgleichung (Zeile 19) zu minimieren wird die

Methode der kleinsten Quadrate verwendet (siehe [JMM09] für alternative Ansätze).

Die neuen Koeffizienten sind eine mit $\xi \in (0,1]$ gewichtete Summe aus den alten und den neu berechneten Koeffizienten. ξ ist ein Dämpfungsparameter, welches zwar die Konvergenzgeschwindigkeit verlangsamt, ohne das der Algorithmus allerdings nicht konvergieren würde. Das Abbruchkriterium wird an ξ angepasst um eine Lösungsgenauigkeit der gleichen Ordnung, unabhängig von ξ zu erhalten.

Der Algorithmus bricht ab, wenn der Unterschied des in dieser Iteration berechneten Kapitals der nächsten Periode nahe genug an dem Ergebnis der letzten Iteration ist. Hierfür wird der einheitsfreie über alle Perioden gemittelte Unterschied betrachtet.

Verwendung der Lösung des GSSA

Als Resultat des Algorithmus sind nicht direkt die Koeffizienten und die Policyfunktion von Interesse, sondern eine Schätzung des Kapitals der nächsten Periode k_{t+1}^n auf den Gitterpunkten x_n , $n = 1, \dots, M$. Da der GSSA die Policyfunktion mit einem Polynom ersten Grades approximiert, ist es möglich die Approximation direkt auf den Gitterpunkten auszuwerten und somit die Funktionswerte der Policyfunktion auf den Gitterpunkten zu erhalten. Mit diesen können die Startkoeffizienten für den Interpolanten auf dem dünnen Gitter berechnet werden. Des Weiteren kann mit den simulierten Werten bestimmt werden, welche Werte für die Berechnung der Policyfunktion von Belang sind. Außerhalb der simulierten Werte für k und a sind die Zustände nicht relevant für den Gleichgewichtszustand. Dies ermöglicht es, für den zweiten Algorithmusteil, eine Abbildung zu definieren, die den relevanten Zustandsraum in den normierten Hyperwürfel abbildet.

5.2 Dünn-Gitter-Algorithmus

Definiere der Übersichtlichkeit halber für den zweiten Algorithmusteil $k := k_t$, $k' := k_{t+1}$ und $k'' = k_{t+2}$ und analog für a .

Algorithmus 2 Dünn-Gitter-Algorithmus

-
- 1: **Initialisierung:**
 - 2: Wähle ein Approximationslevel μ
 - 3: Konstruiere das dünne Gitter auf $[-1,1]^2$ und erhalte Gitterpunkte x_1, \dots, x_M
 - 4: Definiere eine Abbildung $\Phi : (k, a) \rightarrow (v, w)$ wobei $(k, a) \in \mathbb{R}_+^2$ und $(v, w) \in [-1,1]^2$
 - 5: Die Inverse Abbildung Φ^{-1} wird benutzt, um die Zustände (k_n, a_n) in den Modellgrößen aus den Gitterpunkten $x_n = (v_n, w_n) \in [-1,1]^2$, $n = 1, \dots, M$ zu erhalten
 - 6: Wähle Integrationsknoten ϵ_j und Gewichte w_j , $j = 1, \dots, J$
 - 7: Berechne zukünftige Produktivität nach (2.7)
 $a'_{n,j} \leftarrow a_n^\rho \exp(\epsilon_j)$ für $j = 1, \dots, J$ und $n = 1, \dots, M$
 - 8: Berechne eine Anfangsschätzung für $b^{(1)}$ aus der Lösung des GSSA
 - 9: **while** $\frac{1}{M} \sum_{n=1}^M \left| \frac{(k')^{(i)} - (k')^{(i-1)}}{(k')^{(i)}} \right| > \bar{\omega}$ **do**
 - 10: **for** $n = 1, \dots, M$ **do** \triangleright Berechne für alle Gitterpunkte
 - 11: $k'_n \leftarrow$ Auswertung des Polynoms auf den Gitterpunkten \triangleright Berechne Kapital der nächsten Periode bei Zustand $\Phi^{-1}(x_n)$ nach (2.14)
 - 12: $c_n \leftarrow (1 - \delta)k_n + a_n f(k_n) - k'_n$ \triangleright
Berechne Konsum der aktuellen Periode bei Zustand $\Phi^{-1}(x_n) = (k_n, a_n)$ und berechnetem Kapital der nächsten Periode k'_n nach (2.15)
 - 13: **for** $j = 1, \dots, J$ **do** \triangleright Berechne für alle Knoten
 - 14: $(v'_n, w'_{n,j}) \leftarrow \Phi(k'_n, a'_{n,j})$ \triangleright Transformiere Zustandsvariablen der nächsten Periode in $[-1 - \varepsilon, 1 + \varepsilon]^2$, $\varepsilon \geq 0$
 - 15: $k''_{n,j} \leftarrow$ Auswertung des Polynoms an den Punkten $(v'_n, w'_{n,j})$ \triangleright
Berechne das Kapital zwei Perioden später nach (2.14)
 - 16: $c'_{n,j} \leftarrow (1 - \delta)k'_n + a'_{n,j} f(k'_n) - k''_{n,j}$ \triangleright Berechne den Konsum der folgenden Periode bei Zustand (k'_n, a'_n) und berechnetem Kapital $k''_{n,j}$ nach (2.16)
 - 17: **end for**
 - 18: $\hat{k}'_n \leftarrow \beta \sum_{j=1}^J \omega_j \frac{u'(c'_{n,j})}{u'(c_n)} (1 - \delta + a'_{n,j} f'(k'_n)) k'_n$ \triangleright
Berechne bedingte Erwartung in der Fixpunkt-Eulergleichung (5.3) mithilfe gewählter Quadraturmethode
 - 19: **end for**
 - 20: $\hat{b} \leftarrow$ passenden Koeffizienten für \hat{k}' \triangleright Berechne Koeffizienten zu neu approximierten Kapital der nächsten Periode nach 3.5
 - 21: $b^{(i+1)} \leftarrow (1 - \xi)b^{(i)} + \xi \hat{b}$ \triangleright Update die Koeffizienten
 - 22: **end while**
-

Initialisierung

Zunächst wird das dünne Gitter mit gewähltem Approximationslevel nach den Beschreibungen in Kapitel 3 konstruiert.

Die Abbildung Φ wird mithilfe von (4.1) bestimmt, wobei die Simulationswerte des GSSA verwendet werden. Die Inverse Abbildung Φ^{-1} folgt aus (4.2) und mit ihr können aus den Gitterpunkten, die jeweiligen Zustände in den Modellgrößen errechnet werden.

Als Integrationsmethode wird entweder der Gauß-Hermite-Ansatz (4.2.2) oder die in 4.2.3 beschriebene Monomregel gewählt. Abhängig von der Methode werden die jeweiligen Knoten und Gewichte berechnet.

Die Produktivitätsschocks sind unabhängig von anderen Berechnungen und folgen nur (2.7). Deshalb können sie vor dem iterativen Teil berechnet und gespeichert werden.

Zur Berechnung der Anfangsschätzung siehe Abschnitt 5.1.

Iterativer Teil

Die Schleife über n entspricht der Schleife über die Gitterpunkte $x_1, \dots, x_m = \Phi^{-1}(k_1, a_1), \dots, \Phi^{-1}(k_m, a_m)$. In dieser Schleife soll für alle Gitterpunkte das Kapital der nächsten Periode k'_n approximiert werden.

Die Transformation in Zeile 14 wird für die Berechnung des Kapitals zwei Perioden später in Zeile 15 benötigt. Dieses hingegen wird für die Bestimmung des Konsums in der folgenden Periode gebraucht. Damit kann die Approximation des bedingten Erwartungswertes in Zeile 18 berechnet werden.

Bei der Transformation in Zeile 14 kann es vorkommen, dass Werte leicht außerhalb von $[-1,1]^d$ liegen. Auf Grund des autoregressiven Prozesses (2.7) muss der Erwartungswert für die Berechnung des Kapitals der nächsten Periode genommen werden. Hierzu wird der Erwartungswert mithilfe der zuvor berechneten Integrationsknoten und Integrationsgewichte approximiert. Betrachtet sei beispielsweise ein Gitterpunkt, der geringe Produktivität und geringes Kapital widerspiegelt. Bei der deterministischen Quadratur werden Knoten betrachtet, welche die Produktivität noch weiter sinken lassen. Daraus resultiert, dass hierbei die neue Produktivität kleiner ist, als das kleinste Element der Simulation für die Produktivität. Somit wird es durch die Transformation (4.1) nicht mehr in $[-1,1]^d$ abgebildet.

Dementsprechend tritt in Zeile 15 das Problem auf, das Polynom außerhalb von $[-1,1]^d$ auszuwerten. In dieser Arbeit wird der mathematisch nicht korrekte Ansatz verfolgt, das Polynom trotzdem an diesen Stellen auszuwerten, obwohl die Chebyshev-Polynome dort nicht definiert sind. Allerdings ergeben sich gute Ergebnisse, da die Punkte nahe an $[-1,1]^d$ liegen.

Ein mathematisch saubererer Ansatz ist es, eine Extrapolation vorzunehmen, welche allerdings zu höheren Kosten und einer schwierigeren Implementierung führt. Die Berechnung der Koeffizienten in Zeile 20 erfolgt durch den im Abschnitt über Vorberechnung dargestellten Ansatz. Diese kann jedoch auch in jeder Iteration mithilfe der HCFFT gelöst werden.

Das Dämpfungsparameter ξ und die Abbruchbedingung sind analog zum GSSA.

5.3 Änderungen der Algorithmen für das Multi-Country-Modell

Beim Multi-Country-Modell liegen für jedes Land $h = 1, \dots, N$ zwei Zustände vor. Dementsprechend wird das Problem in $2N$ Variablen betrachtet und es werden N landesspezifische Kapital-Policyfunktionen gesucht.

GSSA

Die Policyfunktionen werden mit

$$\psi^h(\{k_t^h, a_t^h\}_{t=0, \dots, T}^{h=1, \dots, N}; b^h), \quad h = 1, \dots, N \quad (5.6)$$

approximiert. Die Startschätzung der Koeffizienten wird für jedes Land analog zum repräsentativen Agenten-Wachstums-Modell gewählt. Sie werden in einer Matrix $B = [b^1, \dots, b^N] \in \mathbb{R}^{(n+1) \times N}$ gespeichert. Mit den Koeffizienten wird das Modell T Perioden vorwärts simuliert, woraus sich $\{k_{t+1}^h\}_{t=0, \dots, T}^{h=1, \dots, N}$ ergibt. Danach wird der durchschnittliche Konsum $\{c_t\}_{t=0, \dots, T}$ mit (2.10) und die bedingte Erwartung $\{y_t^h\}_{t=0, \dots, T}^{h=1, \dots, N}$ wird für jedes Land nach (2.18) berechnet. Nachfolgend muss der Fehler in N Regressionsgleichungen

$$y_t^h = \psi^h(\{k_t^h, a_t^h\}_{t=0, \dots, T}^{h=1, \dots, N}; b^h) + \varepsilon_t^h, \quad h = 1, \dots, N \quad (5.7)$$

minimiert werden. Es wird angenommen, dass ψ^h linear in b^h ist, was zum Regressionsmodell

$$y^h = Xb^h + \varepsilon^h, \quad h = 1, \dots, N \quad (5.8)$$

führt, wobei $y^h = (y_0^h, \dots, y_{T-1}^h)^\top \in \mathbb{R}^T$, $\varepsilon^h = (\varepsilon_0^h, \dots, \varepsilon_{T-1}^h)^\top \in \mathbb{R}^T$ und $X \in \mathbb{R}^{T \times (n+1)}$ mit den Basisfunktionen der Zustände rekursiv konstruiert wird.

Die daraus resultierenden Koeffizienten werden, analog zum eindimensionalen Fall, mit dem Dämpfungsparameter gewichtet auf die alten Koeffizienten addiert.

Die Abbruchbedingung wird zu

$$\frac{1}{MN} \sum_{n=1}^M \sum_{h=1}^N \left| \frac{((k_n^h)_{t+1})^{(i)} - ((k_n^h)_{t+1})^{(i-1)}}{((k_n^h)_{t+1})^{(i)}} \right| < \bar{\omega}\xi \quad (5.9)$$

umformuliert.

Dünn-Gitter-Algorithmus

Auch hier werden N Policyfunktionen mit dem Interpolationspolynom auf dem dünnen Gitter approximiert. Das heißt, es werden die Koeffizienten gesucht, für die der Fehler in (2.18) möglichst klein wird.

Alle Operationen des Dünn-Gitter-Algorithmus für das repräsentative Agenten-Wachstums-Modell werden analog für N Länder ausgeführt. Wie beim GSSA wird der gesamte Konsum durch (2.10) berechnet und für den Konsum der einzelnen Länder wird der Durchschnitt genommen.

Es ist sinnvoll zur Quadratur die in 4.2.3 beschriebene Monomregel zu nutzen, da sie um einiges billiger ist als die Gauß-Hermite-Quadratur (4.2.3) und die Ergebnisse ähnlich gut sind.

Die Abbruchbedingung wird analog zum GSSA zu

$$\frac{1}{MN} \sum_{n=1}^M \sum_{h=1}^N \left| \frac{((k_n^h)')^{(i)} - ((k_n^h)')^{(i-1)}}{((k_n^h)')^{(i-1)}} \right| < \bar{\omega} \quad (5.10)$$

angepasst.

Kapitel 6

Ergebnisse der Implementierung

Im Rahmen dieser Bachelorarbeit werden die beschriebenen Modelle mit folgenden Funktionen und Parameterwerten analysiert.

Es wird angenommen, dass $u(c_t) = (c_t^{1-\gamma} - 1)/(1 - \gamma)$, $f(k_t) = k_t^\alpha$ mit $\alpha = \frac{1}{3}$, $\beta = 0,99$, $\rho = 0,95$, $\gamma = 1$, $\delta = 1$ und $\sigma = 0,01$. Zur weiteren Analyse werden δ , γ und σ einzeln variiert.

Zur Evaluierung der Ergebnisse werden mehrere Optimalitätsbedingungen geprüft. Bis auf die, in eine einheitsfreie Form umgeschriebene, Eulergleichung

$$\mathcal{E}(k_t, a_t) \equiv \text{E}_t \left[\beta \frac{u'(c_{t+1})}{u'(c_t)} (1 - \delta + a_{t+1} f'(k_{t+1})) \right] - 1 \quad (6.1)$$

ergibt sich in allen Optimalitätsbedingungen ein Fehler von null nach Konstruktion. Diese wurden nur zum Vergleich mit verwandter Literatur implementiert. Zum Testen der Optimalitätsbedingungen werden T_{test} Punkte $\{k_t, a_t\}_{t=1, \dots, T_{\text{test}}}$ benötigt. Diese können direkt festgelegt werden oder, wie in der Arbeit realisiert, durch Simulation der Produktivität und Auswerten der Policyfunktion erhalten werden. Danach wird der durchschnittliche sowie der maximale Fehler über alle betrachteten Punkte und Optimalitätsbedingungen berechnet.

Für alle Tests wird die beschriebene Monomregel (4.2.3) genutzt, da diese eine sehr ähnliche Genauigkeit wie die Gauß-Hermite-Quadratur (4.2.3) liefert und die Kosten weitaus geringer sind.

6.1 Ergebnisse beim Agenten-Wachstums-Modell

Nachfolgend werden die maximalen Fehler in (6.1) beim repräsentativen Agenten-Wachstums-Modell bei verschiedenen Modellparametern und Dünn-Gitter-Level μ dargestellt. Genauer gesagt wird von den gesetzten Modellparametern δ , γ und σ immer jeweils eins verändert.

Für das Abbruchkriterium wird $\bar{\omega} = 10^{-10}$ verwendet und es wird das Dämpfungparameter $\xi = 0,1$ genutzt.

Resultate bei isotropen dünnen Gittern

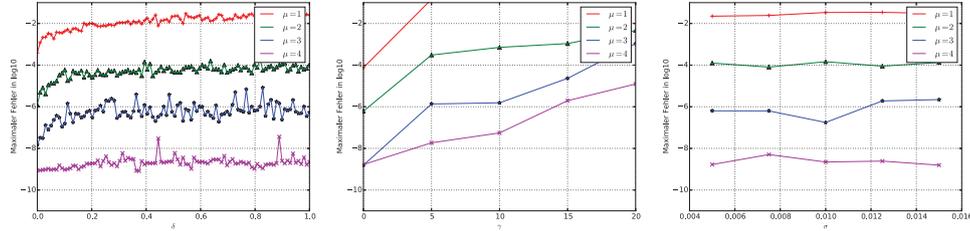


Abbildung 6.1: Vergleich der maximalen Fehler in (6.1) der Approximationen der Level $\mu = 1,2,3,4$ bei $d = 2$ und verschiedenen Modellparametern.

Es ist zu sehen, dass die Genauigkeit, bei den getesteten Modellparametern, bei größerem Level höher ist. Bis auf die Level-1-Approximation bei hohen γ -Werten konvergiert die Fixpunktiteration und liefert vernünftige Ergebnisse. Nach [JMMV14] konvergiert der Algorithmus auch bei hohen γ -Werten und Level 1, wenn eine bessere Startschätzung verwendet wird.

Da die Kosten beim Agenten-Wachstums-Modell aufgrund der kleinen Dimension sehr gering sind, sollten höhere Level verwendet werden. Damit die Fixpunktiteration bei höheren Level konvergiert, müssen gute Startschätzungen verwendet werden. Eine Möglichkeit ist es, für eine μ -Approximation eine $(\mu-1)$ -Approximation durchzuführen und das Ergebnis davon für die Startschätzung zu verwenden. Der Vorgang muss bei einem Level gestartet werden, bei dem die Fixpunktiteration konvergiert, so dass ab diesem Level iterativ die Approximationen der nachfolgenden Level berechnet werden kann.

Resultate bei anisotropen dünnen Gittern

Es folgt eine Analyse der Fehler abhängig vom Level und ein Vergleich mit der anisotropen Version bei verschiedenen Modellparametern. Sei für den anisotropen Fall $\mu = (\mu_1, \mu_2) = (\text{Kapitallevel}, \text{Produktivitätslevel})$.

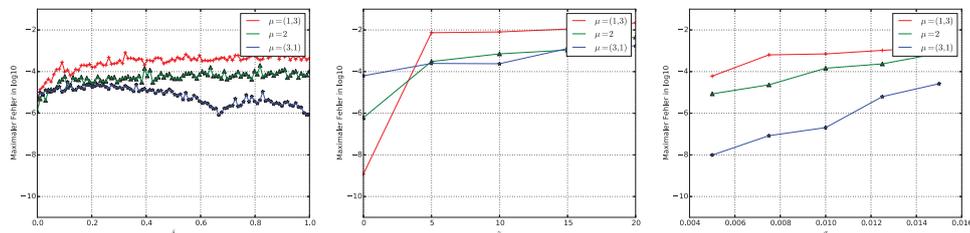


Abbildung 6.2: Vergleich der maximalen Fehler in (6.1) der Approximationen verschiedener (anisotroper) Level bei $d = 2$ und verschiedenen Parametern.

Die Plots zeigen die Nützlichkeit der anisotropen dünnen Gitter bei diesem Modell. Offensichtlich wird für die meisten Modellparameter eine höhere Genauigkeit erreicht, wenn dem Kapital anstatt der Produktivität ein höheres Level zugewiesen wird. Für eine höhere Genauigkeit sollte dementsprechend

eher das Kapitallevel als das Produktivitätslevel erhöht werden. Bei dünnen Gittern höherer Level sollte dementsprechend getestet werden, ob es sinnvoll ist, das Produktivitätslevel zu senken um eine Kostenersparnis ohne Genauigkeitsverlust zu erhalten oder für eine größere Genauigkeit nur das Kapitallevel zu erhöhen. Eine Kombination der Ansätze kann ebenfalls zum Erfolg führen. Es sollte beachtet werden, dass je nach Modellparametern auch schlechtere Ergebnisse resultieren können.

Auswertung der Approximation der Policyfunktion

An dieser Stelle werden die Funktionsauswertungen der Policyfunktion im relevanten Bereich gezeigt. Diese entsprechen dem Kapital der nächsten Periode bei der optimalen Wahl des Konsums für die Maximierung des erwarteten Ge-

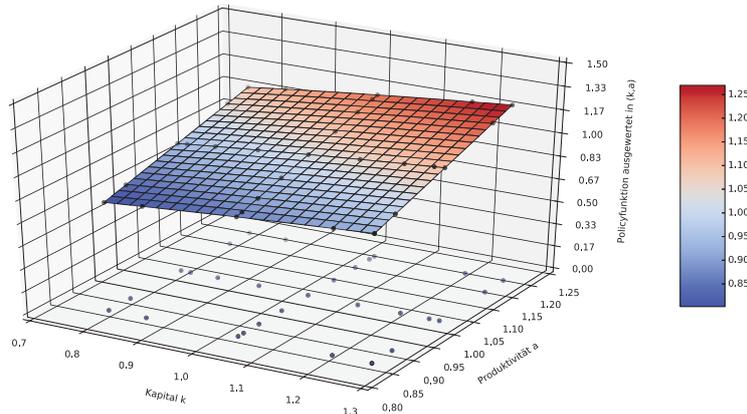


Abbildung 6.3: Auswertung der mit Level 3 approximierten Policyfunktion für betrachtete Zustände (k, a) bei $d = 2$

Zu sehen ist die Level-3-Approximation der auf einem Tensorproduktgitter ausgewerteten Policyfunktion, wobei die Dünn-Gitter-Punkte und ihre Auswertung als Punkte dargestellt sind. Das Kapital der nächsten Periode wird hoch gewählt, wenn das Kapital und die Produktivität der aktuellen Periode groß ist und gegensätzlich bei niedrigem Kapital und niedriger Produktivität in der aktuellen Periode.

6.2 Ergebnisse beim Multi-Country-Modell

Nun werden die Ergebnisse des Algorithmus beim Multi-Country-Modell für das Level $\mu = 2$ dargestellt. Für das Abbruchkriterium wird $\bar{\omega} = 10^{-7}$ genutzt und es wird das Dämpfungsparameter $\xi = 0,05$ verwendet.

Zu sehen ist der maximale Fehler in der einheitsfreien Eulergleichung des Multi-Country-Modells und daneben der durchschnittliche Fehler über alle Optimalitätsbedingungen.

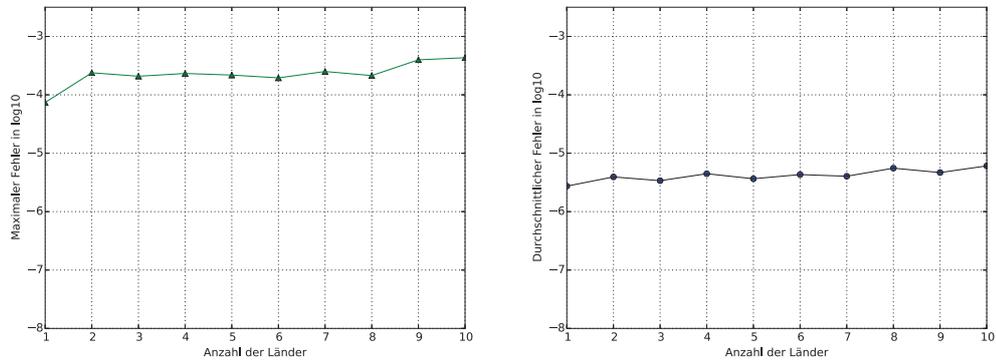


Abbildung 6.4: Maximaler und durchschnittlicher Fehler der Level-2-Approximation für $N=1, \dots, 10$ Länder

Es ist zu erkennen, dass sowohl der maximale als auch der durchschnittliche Fehler bei verschiedenen Dimensionen ähnlich bleibt. So kann bei den gewählten Parametern in kleiner Dimension die Genauigkeit getestet werden und wenn diese zufriedenstellend ist, das Approximations-Level in großer Dimension verwendet werden.

Es folgt ein Vergleich der maximalen Fehler bei verschiedenen Level. Dieser zeigt, dass die Genauigkeit bei unterschiedlicher Anzahl von Ländern ähnlich ist. Des Weiteren gibt es die Tendenz, dass bei höherem Level in höheren Dimensionen schlechtere Genauigkeiten, als im niedrigdimensionalen Fall erreicht werden.

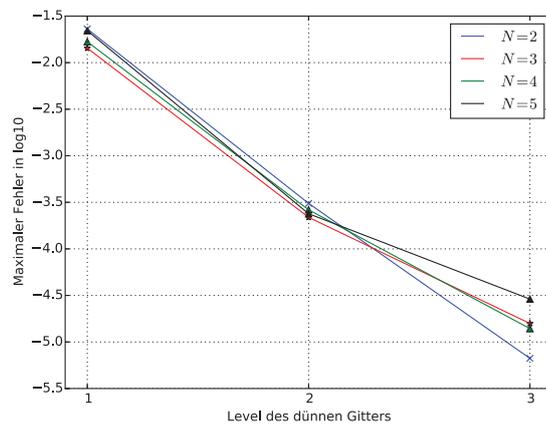


Abbildung 6.5: Vergleich der maximalen Fehler bei Approximationen verschiedener Level

Laufzeitvergleich

Es folgt ein Vergleich der Laufzeit des Matlab-Codes aus [JMMV14] mit der für diese Arbeit geschriebenen C-Implementierung.

Für die Tests wurde ein *Samsung Serie 9* mit *Intel Core i5-3317U* mit 1,7GHz, 4GB RAM und einer SSD verwendet, wobei nur ein Kern für die Berechnungen

genutzt wurde.

Zu sehen ist, dass die Laufzeit bei beiden Implementierungen nicht exponentiell mit der Dimension steigt. Zudem ist die *HCFIT* in höheren Dimensionen langsamer als die Matlab-Variante.

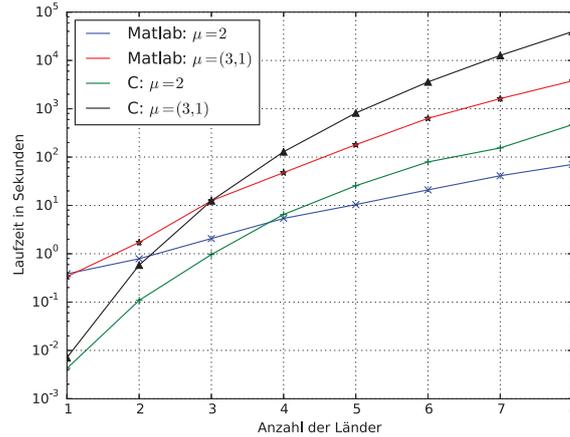


Abbildung 6.6: Vergleich der Laufzeit des Matlab-Codes aus [JMMV14] mit der C-Implementierung bei verschiedenen Dünn-Gitter-Level und bis zu 8 Ländern

Nachfolgend wird die Anzahl der Gitterpunkte bei den oben verwendeten Dünn-Gitter-Level gezeigt, welche ebenfalls nicht exponentiell steigt.

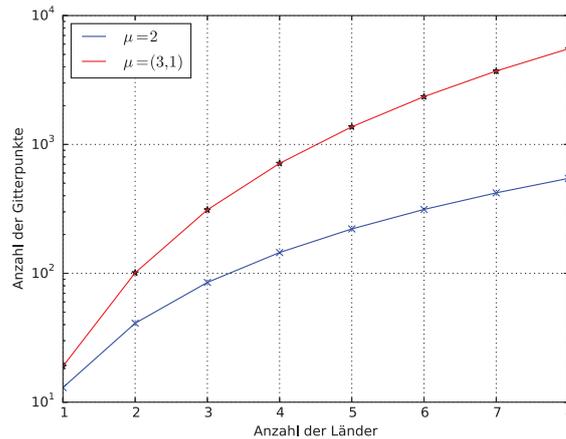


Abbildung 6.7: Anzahl der Gitterpunkte bei verschiedenen Levels und N Ländern, wobei die Dimension $d = 2N$ ist

Die Ergebnisse zeigen, dass die Matlab-Variante bei einer großen Zahl von Gitterpunkten besser abschneidet.

Kapitel 7

Zusammenfassung und Ausblick

Zusammenfassung

In dieser Arbeit wird der Ansatz der dynamischen Programmierung zur Lösung von Optimierungsproblemen in der Wirtschaft dargestellt. Durch die Verwendung von dünnen Gittern und Interpolation auf diesen, ist es möglich, die Kapital-Policyfunktion bei geringen Kosten zu approximieren. Simulation liefert den relevanten Zustandsraum, was die Lösung auf diesem verbessert, da kein Aufwand in unnötige Zustände gesteckt wird. Die Verwendung von einer Monomregel, erlaubt es den Erwartungswert auch in hohen Dimensionen zu approximieren, ohne unter dem Fluch der Dimensionen zu leiden.

Ausblick

In der *HCFIT*-Bibliothek wird aktuell eine Parallelisierung der Funktionen implementiert. Diese ermöglicht eine Verbesserung der Laufzeit, besonders wenn ein Supercomputer mit vielen Prozessoren genutzt wird.

In der Bibliothek gibt es noch keine Funktion zur Berechnung der Matrix der in den Gitterpunkten ausgewerteten Chebyshev-Basisfunktionen. Diese kann nur über einen Trick mit konstruierten Koeffizienten und der Auswertung bei diesen erhalten werden (siehe Code-Dokumentation). Dieser Umweg ist zwar sehr ineffizient, jedoch findet die Berechnung außerhalb des iterativen Teils für die Vorberechnung statt und ist somit akzeptabel. Eine Erweiterung um diese Funktion würde die Laufzeit des Algorithmus jedoch verbessern.

In dieser Arbeit wird dem Ansatz über die Eulergleichung nachgegangen. Es existieren auch Ansätze über die Bellmangleichung, welche bei einigen Problemen sinnvoller sind. Vor- und Nachteile dieser Ansätze können noch weiter analysiert werden.

Verwandte Modelle können mit ähnlichen Ansätzen gelöst werden. Dementsprechend können weitere Modelle mit beschriebenen Methoden analysiert werden.

Kapitel 8

Notationsverzeichnis

Kapitel 2

t	Zeit-Periode
β	Diskontfaktor
c_t	Konsum in Periode t
k_t	Kapital in Periode t
u	Nutzenfunktion
f	Produktionsfunktion
V	Wertfunktion
E_t	Operator der bedingten Erwartung
a_t	Produktivität in Periode t
δ	Abschreibungsrate
ρ	Autokorrelationskoeffizient
ϵ_t	Schock in Periode t
σ	Standardabweichung
λ	Wohlfahrtsgewicht
0_N	Null-Vektor der Länge N
Σ	Kovarianzmatrix
ζ_t	Landesspezifischer Schock in Periode t
ζ_t^h	Weltweiter Schock in Periode t
\hat{k}_{t+1}	neue Lösung für das Kapital der nächsten Periode
K	Interpolationspolynom für Kapital-Policyfunktion

Kapitel 3

d	Dimension
s	Anzahl der eindimensionalen Gitterpunkte
f	Zu interpolierende Funktion (nur in diesem Kapitel)
\tilde{f}	Approximation von f
F_d^k	Raum der stetigen Ableitungen der Ordnung k
$c_{d,k}$	Konstante, von d und k abhängig
m	Anzahl der Gitterpunkte
S_i	Menge von eindimensionalen Punkten

$m(i)$	Anzahl der Elemente von S_i
$T_i(x)$	Chebyshev-Basisfunktion
q_j	Extrema des Chebyshev-Polynoms
A_i	Menge von eindimensionalen Punkten, disjunkt konstruiert
i_j	Index der Menge A_i in Dimension j
μ	Level des dünnen Gitters
$H^{d,\mu}$	dünnes Gitter mit Level μ in d Dimensionen
Ψ	Basisfunktion in $[-1, 1]^d$
b	Koeffizientenvektor
\hat{f}	Interpolationspolynom
x_i	Gitterpunkte
\mathcal{O}	Ordinäre Polynombasis
B_i	Menge von eindimensionalen Chebyshev-Basisfunktionen, disjunkt konstruiert
X	Matrix der in den Gitterpunkten ausgewerteten Basisfunktionen

Kapitel 4

T	Anzahl der simulierten Perioden
ε_t^I	absoluter Integrationsfehler in Periode t
y_t	Approximation des bedingten Erwartungswerts in der Fixpunkt-Eulergleichung
J	Anzahl der Knoten der Integrationsmethode
$\varepsilon_{t,j}$	Knoten der Integrationsmethode, Realisierungen der Zufallsvariable ε_t des Modells
$\omega_{t,j}$	Gewichte der Integrationsmethode für alle Perioden t und Knoten $\varepsilon_{t,j}$
f	Integrand (nur in diesem Kapitel)
$\mathcal{W}(\varepsilon)$	Dichtefunktion der multivariaten Normalverteilung
ε	(mehrdimensionale) normalverteilte Zufallsvariable
μ	Vektor der Erwartungswerte (nur in diesem Kapitel)
Σ	Kovarianzmatrix
L	Matrix der Cholesky-Zerlegung
ν_j	Knoten der Gauß-Hermite-Quadratur
ω_j	Gewichte der Gauß-Hermite-Quadratur

Kapitel 5

ψ	Polynom ersten Grades
A	Normalisierungskonstante
\bar{k}	Gleichgewichtszustand des Kapitals
\bar{a}	Gleichgewichtszustand der Produktivität
$\bar{\omega}$	Abbruchkriterium
ξ	Dämpfungsparameter
Φ	Abbildung, die die lineare Transformation beschreibt

Kapitel 6 T_{Test}

Anzahl der für den Test simulierten Perioden

Literaturverzeichnis

- [Bel57] BELLMAN, R. E.: *Dynamic programming*. Princeton University Press, 1957
- [Bel61] BELLMAN, R. E.: *Adaptive Control Processes*. Princeton University Press, 1961
- [BNR00] BARTHELMANN, Volker ; NOVAK, Erich ; RITTER, Klaus: High dimensional polynomial interpolation on sparse grids. In: *Advances in Computational Mathematics* 12 (2000), Nr. 4, S. 273–288
- [Cao08] CAO, Jin: *Chapter 5 Dynamic Programming*. {http://www.sfm.econ.uni-muenchen.de/lehre/advanced_macro/jin_cao/ch5.pdf}. Version: 2008
- [Ell08] ELLISON, Martin: *The Neoclassical Growth Model and Dynamic Programming*. http://users.ox.ac.uk/~exet2581/recursive/sg_mat.pdf. Version: 2008
- [For08] FORSTER, Otto: *Analysis 2*. Vieweg+ Teubner, 2008. – 110ff S.
- [Ger06] GERVAIS, Martin: *The Neoclassical Growth Model and Dynamic Programming*. <http://www.wiwi.uni-regensburg.de/lee/links/IHS%20teaching08/lecture2.pdf>. Version: 2006
- [GH14] GRIEBEL, M. ; HAMAEEKERS, J.: Fast Discrete Fourier Transform on Generalized Sparse Grids. In: *Sparse grids and Applications* Bd. 97. Springer, 2014, S. 75–108
- [JMM09] JUDD, Kenneth ; MALIAR, Lilia ; MALIAR, Serguei: Numerically Stable Stochastic Simulation Approaches for Solving Dynamic Economic Models. (2009), August, Nr. 15296
- [JMM10] JUDD, Kenneth L. ; MALIAR, Lilia ; MALIAR, Serguei: A Cluster-Grid Projection Method: Solving Problems with High Dimensionality / National Bureau of Economic Research. Version: 2010. <http://www.nber.org/papers/w15965.pdf>. 2010 (15965). – NBER Working Papers
- [JMMV14] JUDD, Kenneth L. ; MALIAR, Lilia ; MALIAR, Serguei ; VALERO, Rafael: Smolyak method for solving dynamic economic models. In:

- Journal of Economic Dynamics and Control* 44 (2014), Nr. C, S. 92–123
- [JS11] JUDD, Kenneth ; SKRAINKA, Benjamin: High performance quadrature rules: how numerical integration affects a popular model of product differentiation / Centre for Microdata Methods and Practice, Institute for Fiscal Studies. Version: 2011. <http://EconPapers.repec.org/RePEc:ifs:cemmap:03/11>. 2011 (CWP03/11). – CeMMAP working papers
- [Jud98] JUDD, K.L.: *Numerical Methods in Economics*. MIT Press, 1998 (Scientific and Engineering)
- [Kru12] KRUEGER, Dirk: *Macroeconomic Theory*. <http://www.ssc.wisc.edu/~aseshadr/econ714/MacroTheory.pdf>. Version: 2012
- [Mar88] MARCET, Albert: The Parameterized Expectation Approach. (1988)
- [MCWG95] MAS-COLELL, Andreu ; WHINSTON, Michael ; GREEN, Jerry: *Microeconomic Theory*. Oxford University Press, 1995. – 964–966 S.
- [Pam09] *Kapitel 15.3*. In: PAMPEL, Thorsten: *Mathematik für Wirtschaftswiss.* Springer Verlag, 2009
- [SB94] SIMON, Carl P. ; BLUME, Lawrence: *Mathematics for Economists*. W. W. Norton, 1994. – 453–457 S.
- [SJ13] SCHMEDDERS, K. ; JUDD, K.L.: *Handbook of Computational Economics*. Elsevier Science, 2013 (Handbook of Computational Economics Bd. 3)
- [Ski08] SKIENA, Steven S.: *The Algorithm Design Manual*. 2nd. Springer Publishing Company, Incorporated, 2008
- [Smi99] SMITH, Gregor W.: *Dynamic Equilibrium Models III: Business-Cycle Models*. {<http://qed.econ.queensu.ca/pub/faculty/smithgw/econ815A/815-5.pdf>}. Version: 1999
- [Smo63] SMOLYAK, S. A.: Quadrature and interpolation formulas for tensor products of certain class of functions. In: *Dokl. Akad. Nauk SSSR* 148 (1963), Nr. 5, S. 1042–1053. – Transl.: Soviet Math. Dokl. 4:240-243, 1963