Der Einsatz evolutionärer Computermodelle bei der Untersuchung historischer und politischer Fragestellungen

von:

Eckhart Arnold

Erfurt, 1. Dezember 2002

Inhaltsverzeichnis

1 Einleitung

Wie viele andere bedeutende wissenschaftliche Entdeckungen auch, hat die Evolutionstheorie seit ihrer Erfindung Wirkungen entfaltet, die weit über den Kreis ihrer ursprünglichen Anwendung hinaus reichen. Daher verwundert es nicht, dass schon bald nach Darwins Entdeckung die wichtigsten Stichworte seiner Theorie (bzw. der durch Herbert Spencer und andere popularisierten Form von Darwins Theorie (Vgl. Koch 1973, S.38ff.)) in den Gesellschaftswissenschaften auftauchten - nicht selten in Form kruder Analogien und Übertragungen von halbverstandenen biologischen Erkenntnissen. So bildete bis etwa zur Mitte des 20. Jahrhunderts der „Sozialdarwinismus", d.i. die Ansicht, dass auch das Zusammenleben der Menschen in der Gesellschaft sowie der Wettbewerb zwischen den Gesellschaften durch einen „Kampf ums Dasein" und das „Überleben des Stärksten" bestimmt sei, die wohl dominanteste Form der Übertragung evolutionstheoretischer Vorstellungen auf den gesellschaftlichen Bereich. Nicht zuletzt wegen seiner verheerenden normativen Konsequenzen gilt der Sozialdarwinismus heutzutage zu Recht als diskreditiert.

Sieht man von solchen Missverständnissen jedoch einmal ab, so erscheint die grundlegende Frage, ob nicht die Entwicklung kultureller und sozialer Systeme oder auch politischer Ordnungen durch evolutionäre Mechanismen erklärt werden kann, nach wie vor interessant. Es ist durchaus denkbar, dass auch im Wettbewerb der Gesellschaftsformen oder der Produktionstechniken die Mechanismen der Mutation und Selektion wirksam sind, die - ähnlich wie in der natürlichen Evolution - unter gegebenen Umweltbedingungen einen evolutionären Entwicklungsprozess herbeiführen.

In diesem Aufsatz soll deshalb einmal der Frage nachgegangen werden, wie evolutionäre Erklärungen historischer, gesellschaftlicher und politischer Vorgänge beschaffen sein könnten, und bei welchen Vorgängen dieser Art ein evolutionärer Erklärungsansatz erfolgversprechend erscheint. Um ein Ergebnis dieses Aufsatzes vorweg zu nehmen: Es gibt eine Reihe von Phänomenen aus dem Bereich der historischen und politischen Wissenschaften, bei denen ein Erklärungsansatz mit Hilfe evolutionärer Modelle sich als fruchtbar erweisen könnte. Dazu gehören z.B. historische Vorgänge, die sich über große Zeiträume hinweg abspielen, wie etwa die Entstehung sozialer und politischer Normen (sowohl solcher, die ethischer Natur sind, als auch solcher, die man eher als bloße Klugheitsregeln einstufen würde). Da die Evolutionstheorie eine kausale Erklärung teleologischer Strukturen darstellt, könnte ein evolutionärer Ansatz besonders dann hiflreich sein, wenn es darum geht die Entstehung leistungsfähiger sozialer Institutionen und wirklichkeitstauglicher Normensysteme zu erklären, die in der Form, in der sie sich schließlich durchsetzen, nicht zuvor geplant gewesen sein konnten. Was die Untersuchung von

Normen betrifft kann dabei interessanterweise nicht nur die empirische Frage, warum diese oder jene Norm sich durchgesetzt hat, sondern auch die ethische Frage, ob eine bestimmte Norm als moralisch gültig anerkannt werden sollte, unter einem evolutionären Gesichtspunkt betrachtet werden.

Im folgenden werde ich zunächst in abstracto beschreiben, wie evolutionäre Erklärungen kultureller Phänomene konstruiert werden können, und welche Bedingungen erfüllt sein müssen, damit eine solche evolutionäre Erklärung als vollständig gelten kann. Dazu werde ich zunächst eine allgemeine Evolutionstheorie vorstellen, die den Rahmen für Erklärungen historischer Vorgänge abgeben soll. Innerhalb dieses theoretischen Rahmens soll dann besonderes Augenmerk auf den Einsatz von Computersimulationen gerichtet werden. Hierfür werde ich in Anlehnung an Robert Axelrods bahnbrechende Untersuchungen (Axelrod 1984) eine Computersimulation vorstellen, die es erlaubt, evolutionäre - in diesem Fall speziell populationsdynamische Vorgänge - am Modell zu studieren.

Schließlich soll an einigen Beispielen wenigstens andeutungsweise gezeigt werden, wie evolutionäre Erklärungen in der Philosophie und den Gesellschaftswissenschaften eingesetzt werden können.

2 Evolutionäre Erklärungen

2.1 Eine allgemeine (axiomatische) Evolutionstheorie als Grundlage

Bevor der Versuch unternommen wird, gesellschaftliche Vorgänge evolutionär zu erklären, ist es notwendig, sich einen möglichst klaren Begriff davon zu bilden, was ein evolutionärer Prozess ist. Nur so wird es später möglich sein zu unterscheiden, ob ein evolutionärer Erklärungsversuch tatsächlich eine Erklärung liefert, oder ob es sich nur um die erzählerische Beschreibung eines historischen Vorgangs in einem evolutionswissenschaftlichen Jargon handelt. Gefordert ist also zunächst eine allgemeine Theorie der Evolution.

Für eine solche Theorie wird hier zunächst auf die bei Schurz (Schurz 2001) beschriebene axiomatische Evolutionstheorie zurückgegriffen. Anschließend soll auf die Frage eingegangen werden, ob eine Evolutionstheorie in dieser Form als Erklärungsgrundlage bereits gehaltvoll genug ist, und wie sie für die Erklärung kultureller Phänomene fruchtbar gemacht werden kann.

Die axiomatische Evolutionstheorie besagt folgendes: Evolution, d.h. die Entwicklung von „Organismen“ zu immer angepassteren („fitteren“) Formen, findet mit einer bestimmten Wahrscheinlichkeit statt, wenn vier grundlegende Bedingungen erfüllt sind, die Schurz mit folgenden Worten beschreibt.:

- „*(V)ariation*: Das Erbmaterial eines Organismus variiert durch verschiedene Prozesse und diese Variationen haben Einfluss auf die Fitness (=Reproduktionsrate) des Organismus bzw. der Spezies.

- *(R)eproduktion*: [Das Erbmaterial] eines Organismus wird von Generation zu Generation reproduziert.

- *(S)elektion*: Die Umgebungsbedingungen üben auf den Organismus bzw. die Spezies einen Selektionsdruck aus, der die Reproduktionsrate begrenzt. Dieser Selektionsdruck führt zu unterschiedlichen Reproduktionsraten unterschiedlicher Spezien bzw. unterschiedlicher Varianten derselben Spezies, mit der Folge, dass sich die Varianten bzw. Spezien mit der höheren Reproduktionsrate langfristig durchsetzen.

- *UST*: Die zeitliche Änderungsrate der selektiv wirksamen Umgebungsparameter ist entweder gering, oder aber regulär-periodisch.“

(Schurz 2000, 329f., 335; vgl. auch Wieser 1994b, 16)

Natürlich sind es im Bereich der Kulturwissenschaften nicht Organismen, die sich entwickeln, sondern soziale Normen und Institutionen. Aber die Gesetzmäßigkeiten sind genau dieselben. Reichen aber diese vier Bedingungen bereits hin, um ein bestimmtes Phänomen evolutionär zu erklären?

Man könnte einwenden, dass ein wesentlicher Aspekt bei dieser axiomatischen Evolutionstheorie eher verschleiert wird, nämlich die Tatsache, dass die Existenzmöglichkeit (und damit auch die Möglichkeit der Evolution) bestimmter Organismen auch von ontologischen Voraussetzungen abhängig ist. Dass sich z.B. ein Vogelflügel in der Evolution hat entwickeln können, war nur möglich, weil das System Luft-Vogelflügel aufgrund der Gesetze der Aerodynamik tatsächlich funktionert. Nun sind derartige ontologische Voraussetzungen zwar implizit in der *Bedingung S* enthalten. Aber es ist nicht unwichtig, sich vor Augen zu halten, dass in die *Bedingung S* zum Teil sehr anspruchsvolle ontologische Voraussetzungen eingehen können. Dies bedeutet, dass das Erfülltsein von S im Einzelfall sehr schwer nachzuweisen sein dürfte, da dies die Klärung der ontologischen Möglichkeitsbedingungen erfordert.

2.2 Die Anwendung der allgemeinen Evolutionstheorie auf kulturwissenschaftliche und historische Fragestellungen

Bei der Erklärung historischer Vorgänge fällt die eben erwähnte Schwierigkeit zum Glück weniger gravierend aus, da es sich hierbei um ex-post Erklärungen handelt, womit die Existenzmöglichkeit des zu Erklärenden zwangsläufig gegeben ist. Dennoch besteht ein wichtiger Unterschied darin, ob erklärt werden soll, warum eine soziale Insitution funktioniert, oder ob erklärt werden soll, wie sie entstehen und sich durchsetzen konnte. Das Funktionieren einer sozialen Insitution kann (ohne Zirkelschluss) nicht allein durch ihre evolutionäre Entstehung erklärt werden. Umgekehrt ist mit dem Nachweis der Funktionstüchtigkeit oder der Zweckmäßigkeit bestimmter sozialer Insitutionen noch nicht erklärt, wie und warum sie entstehen konnten. Denkbar ist aber, dass das die Entstehung generell leistungsfähiger sozialer Institutionen dadurch erklärt werden kann, dass die Bedingungen für evolutionäre Entwicklungsprozesse im Sine der axiomatischen Evolutionstheorie gegeben sind, da derartige Prozesse dort, wo dies möglich ist, vergleichsweise rasch zur Herausbildung „quasi-teleologischer“ Formen führen. Eine solche Erklärung wird weiter unten am Beispiel der von Eric L. Jones (Jones 1981) gebenen Begründung für den historischen „Erfolg“ des europäischen Kontinents erörtert werden.

Zur Beantwortung der Frage, ob eine axiomatische Evolutionstheorie wie die oben skizzierte bereits hinreichend gehaltvoll ist, um die Herausbildung leistungsfähiger sozialer Institutionen wenigstens im Prinzip zu erklären, müsste ausserdem noch genauer untersucht werden, ob evolutionäre Prozesse, die diesen vier Axiomen genügen, bereits mit hinreichender Wahrscheinlichkeit zur Herausbildung „quasi-teleologischer“ Formen führen. Man könnte dazu den durch diese Axiome beschriebenen evolutionären Prozess als einen Optimierungsalgorithmus auffassen und Überlegungen zu dessen Effizienz anstellen (vgl. Axelrod 1997, 10ff., vgl. Wagner 1994). Darüber hinaus könnte man die Frage aufwerfen, ob die oben beschriebene Evolutionstheorie nicht speziell für den Bereich der Kulturwissenschaften noch weiter verfeinert könnte.

Für den Bereich der biologischen Evolution ist dies möglich, indem man die Merkmale genetischer Vererbung und Entwicklung durch weitere Axiome näher eingrenzt (vgl. Schurz 1998, 329ff.). Für die Beschreibung kultureller Evolutionsprozesse wäre ein ähnliches Vorgehen denkbar, etwa indem man versucht, die Reproduktions- und Selektionsbedinungen von Kulturtechniken (Normen, Institutionen, Technologien) durch Gesetzmäßigkeiten genauer zu erfassen. Ein erster Ansatz müsste in der Untersuchung der Frage bestehen, wie sich Kulturtechniken durchsetzen. Grundsätzlich kann dies auf zweierlei Weise geschehen: 1. indem die Gesellschaften, die sie angenommen haben, andere Gesellschaften, die nicht darüber verfügen, verdrängen (durch Krieg, Zerstörung ihrer Lebensgrundlagen etc.) 2. indem Gesellschaften, die noch nicht über eine bestimmte Kulturtechnik verfügen, sie von einer anderen Gesellschaft, die sie bereits besitzt, erlernen (was wiederum auf zweierlei Weise geschehen kann: dadurch dass die Kulturtechnik selbst oder dadurch dass Idee ihrer Möglichkeit weitergegeben wird (vgl. Diamond 1998, S.215ff.)). Wenn es nun gelänge, über die Geschwindigkeit der Ausbreitung von Kulturtechniken auf diesen Wegen genauere Aussagen zu treffen, so erscheint es zumindest im Prinzip denkbar - analog etwa zu den populationsdynamischen Gesetzen in der biologischen Evolution - Gesetzmäßigkeiten für die Verbreitung von Kulturtechniken aufzustellen, wenn auch in den Kulturwissenschaften nicht dieselbe Genauigkeit der Vorhersagen erwartet werden darf.

Um einen entsprechenden historischen Entwicklungsvorgang nun mit Hilfe der allgemeinen Evolutionstheorie erklären zu können, müssen im wesentlichen drei Bedingungen erfüllt sein:

1. Die Voraussetzungen für einen evolutionären Prozess müssen gegeben sein, d.h. es muss gezeigt werden, dass für den beschriebenen Vorgang mindestens die vier Grundaxiome der allgemeinen Evolutionstheorie (V, R, S und UST) erfüllt sind und gegebenenfalls noch weitere Bedingungen, die speziell für gesellschaftliche und kulturelle Entwicklungsprozesse gelten.

2. Das Ergebnis des historischen Vorgangs muss nach Maßgabe dieser Axiome, insbesondere der Selektionsbedingungen, auch wahrscheinlich gewesen sein. Sonst wäre die Hypothese, dass bei dem untersuchten Vorgang ein evolutionärer Prozess im Sinne der axiomatischen Evolutionstheorie stattgefunden hat, bereits widerlegt.

3. Der evolutionäre Prozess muss nachgewiesen werden, indem die Zwischenstadien dieses Prozesses identifiziert werden. Falls die historischen Quellen lückenhaft sind, muss zumindest die Möglichkeit solcher Zwischenstadien dargelegt werden können.

Nur wenn alle drei Bedinungen erfüllt sind, kann die evolutionäre Erklärung eines historischen Vorganges als vollständig gelten. Wird eine dieser Bedingungen nicht erfüllt, so ist die evolutionäre Erklärung entweder falsch (wenn Bedingung 1 oder 2 verletzt wird) oder sie bleibt eine bloße Behauptung (sofern Bedingung 3 nicht erfüllt ist).

3 Computermodelle zur Simulation evolutionärer Vorgänge

Soeben wurde umrisshaft gezeigt, wie eine allgemeine Evolutionstheorie durch Axiome beschrieben werden kann, und wie diese Theorie auf historische Vorgänge angewendet werden könnte. Doch gerade die Anwendung dieser Theorie auf bestimmte wissenschaftliche Probleme wirft noch weitere Fragen auf. Denn selbst wenn in einer gegebenen Situation alle wirksamen Selektionsbedingungen fesgestellt worden sind, bleibt noch offen, welche Organismen (bzw. welche sozialen Normen, Insitutionen, Kulturtechniken) nach Maßgabe dieser Selektionsbedingungen die „Fitteren“ sind, und nach welchem Muster und wie rasch dementsprechend der evolutionäre Prozess ablaufen wird. In einigen Fällen mag die Antwort zwar offensichtlich sein, da manche evolutionäre Entwicklungen nur unter ganz bestimmten Umweltbedingungen möglich sind. So konnte sich beispielsweise die Viehzucht nur in solchen geographischen Regionen entwickeln, wo Tiere leben, die domestizierbar sind (Diamond 1998, S.85ff.). In anderen Fällen liegt die Antwort jedoch nicht unmittelbar auf der Hand. Es empfiehlt sich daher evolutionäre Vorgänge zunächst am Modell zu studieren, um so die Bedingungen evolutionärer Vorgänge and typische Entwicklungsmuster festzustellen, nach denen in der Wirklichkeit Ausschau gehalten werden muss.

Während die Untersuchung evolutionärer Prozesse in der Biologie naturgemäß sehr weit fortgeschritten ist, trifft man in den Gesellschaftwissenschaften (von Ausnahmen abgesehen) allenfalls in der Ökonomie auf entsprechend ausgearbeitete Modelle. Letztere werden zumeist auf spieltheoretischer Grundlage und auf mathematisch zum Teil sehr anspruchsvollen Niveau gebildet (z.B. Binmore/Samuelson 1992). Zu der mathematematischen Modellbildung tritt als eine vergleichsweise junge Technik der Einsatz von Computersimulationen hinzu. Computersimulationen sind dabei als eine Ergänzung und Erweiterung der mathematischen Modellbildung zu verstehen. Im Gegensatz zu diesen zeichnen sie sich allerdings häufig durch eine größere Einfachheit und Anschaulichkeit aus. Zudem gibt es bstimmte Probleme, die sich mathematisch nur sehr schwer in den Griff bekommen lassen, die aber mühelos auf einem Computer programmiert werden können.

Eine Pionierleistung auf diesem Gebiet stellt Robert Axelrods „Evolution der Kooperation“ (Axelrod 1984) dar. Axelrod untersuchte in diesem Werk mit Hilfe von Computerturnieren, in denen er eine Anzahl von Spielern mit unterschiedlichen Strategien parweise gegeneinander antreten ließ, welche Strategien im wiederholten Gefangenendilemmaspiel besonders erfolgreich abschneiden, und ob eher kooperative Strategien oder eher destruktive Strategien eine Chance haben, sich auf lange Sicht evolutionär durchzusetzen.

Um das Prinzip derartiger Computersimulationen besser zu verdeutlichen, werde ich im folgenden eine einfache Simulation des iterierten Gefangenendilemmas nach dem Vorbild des Computerturniers in Robert Axelrods „Evolution der Kooperation“ (Axelrod 1984) vorstellen. Dabei werde ich auch einige Varianten von Axelrods Simulation durchspielen, bei denen es um den Einfluss von Missverständnissen bzw. Fehlleistungen und um die evolutionäre

Stabilität erfolgreicher Strategien unter dem Druck degenerativer Mutationen geht.

Es zeigt sich dabei übrigens, dass Axelrods Ergebnisse sehr stark von der Wahl bestimmter Vorgaben wie der Strategiemenge und der Auszahlungsparameter abhängig sind, und dass bei geänderten Bedingungen viele von Axelrods Behauptungen nicht mehr ohne weiteres aufrecht erhalten werden können. Dies bedeutet, dass man die Genauigkeit der Ergebnisse Axelrods und ähnlicher Computersimulationen für die Beurteilung von Vorgängen aus dem gesellschaftlichen oder politischen Bereich nicht überschätzen darf, da in diesen Bereichen die ermittelten Parameter oft nur ungenaue Schätzwerte wiedergeben, so dass die Instabilität des Modells gegenüber der Veränderung der Paramter sich auf die Schlussfolgerungen vom Modell auf die Wirklichkeit überträgt.

Beabsichtigt ist aber weniger die Widerlegung oder Relativierung von Axelrods Ergebnissen - die Schwächen seiner Theorie sind in der einschlägigen Literatur eingehend diskutiert worden (vgl. Binmore 1994, 194ff., vgl. Schüßler 1990, 33ff.) - als vielmehr die Darstellung der Einsatzmöglichkeiten von Computersimulationen an einem hinreichend einfachen Beispiel.

3.1 Ein Beispiel: Die Simulation des iterierten Gefangenendilemmas

Bei dem von Axelrod durchgeführten Computerturnier handelt es sich um eine Simulation des iterierten Gefangenendilemmas. Ein einfaches (nicht iteriertes) Gefangenendilemma kann als ein Spiel beschrieben werden, in dem zwei Spieler die Wahl haben zu kooperieren oder nicht zu kooperieren (also zu „defektieren“). Der Gewinn, den jeder Spieler erhält, hängt vom Verhalten beider Spieler ab. Ein Spieler, der kooperiert, während der andere defektiert, erhält überhaupt nichts, während sein Gegenspieler den höchst möglichen Gewinn einstreicht. Defektieren beide Spieler, so bekommen sie zwar einen Gewinn, doch ihr Gewinn fällt nur sehr gering aus. Kooperieren beide Spieler, so bekommen sie einen recht ansehnlichen Gewinn, der aber nicht dem Höchstgewinn entspricht. Eigentlich wäre es für beide Spieler am besten zu kooperieren, wenn sie nur sicher gehen könnten, dass der andere sich ebenso verhält. Gerade dies ist im einfachen Gefangenendilemma aber nicht der Fall.

Anders verhält es sich beim wiederholten („iterierten“) Gefangenendilemma. Hier spielen die beiden Spieler eine Folge von Gefangenendilemmasituationen durch. Zwar wissen sie wiederum nicht, ob der andere Spieler als nächstes kooperieren oder defektieren wird. Aber sie verfügen über die gesamte Folge der vergangenen Züge als Anhaltspunkt und zudem können sie in der nächsten Runde auf das Verhalten des Gegners in der jetzigen Runde reagieren.

Dass das iterierte Gefangenendilemma (ebenso wie das einfache Gefangenendilemma) ein plausibles Modell vieler typischer sozialer Situationen ist, bedarf kaum einer weiteren Erörterung. Ein gutes Beispiel sind etwa die Verhandlungen in Parlamentsausschüssen, bei denen dieselben Verhandlungspartner immer wieder aufeinander treffen und jeder bestimmte Ziele verfolgt, bei denen er von der Zustimmung der anderen abhängig ist.[1]

Eine wichtige Eigenschaft des iterierten Gefangenendilemmas besteht darin, dass es keine eindeutig beste Strategie gibt. Wie erfolgreich eine bestimmte Strategie ist, hängt immer auch von der Strategie des Gegenspielers ab. Welche Strategie ist dann aber insgesammt, d.h. wenn man sie gegen eine Vielzahl sehr unterschiedlicher Gegenspieler antreten lässt, am erfolgversprechendsten? Um diese Frage zu klären, führte Axelrod sein Computerturnier durch. Eine Implementierung dieses Computerturnieres soll im folgenden beschrieben werden. Sie unterscheidet sich von Axelrods Computerturnier durch die vergleichsweise geringe Anzahl teilnehmender Strategien. Doch geht es hier im wesentlichen um die Verdeutlichung des Prinzips. Abgesehen davon stellt sich das - weiter unten noch anzusprechende - Problem der kontingenten Strategiemenge genauso für die größere Anzahl von Strategien in Axelrods Computerturnier.

3.1.1 Die Implementation des Computerturniers

Insgesamt nahmen 12 unterschiedliche Strategien an dem hier beschriebenen Computerturnier teil. Es lohnt sich nicht, alle Strategien einzeln zu beschreiben, zumal ihre Namen meist selbsterklärend sind (*Random, Tit for Tat, Always friendly*). Die drei Strategien *GraciousTFT, Tester* und *Analyst* sollen jedoch mit Hilfe von Quellcodebeispielen etwas näher erläutert werden. Die Beispiele aus dem Programmcode sind in der Programmiersprache Python geschrieben, einer Interpretersprache, die sich wegen ihrer Einfachheit und der guten Lesbarkeit des Programmcodes für solche Aufgaben empfiehlt.

Die Strategie *GraciousTFT* ist eine Variante von *Tit for Tat*, die jedoch längere Folgen gegenseitiger Bestrafungen erkennt und durch ein Friedensangebot zu beendigen versucht. Damit beseitigt *GraciousTFT* eine Schwäche, die *Tit for Tat* im Zusammentreffen mit böswilligen Varianten des eigenen Typs aufweist. Im folgenden Programmausschnitt steht ein Rückgabewert

[1] Dies gilt natürlich vor allem in Politischen Systemen mit schwach ausgeprägter Fraktionsdisziplin wie den USA.

von 1 für Kooperation und ein Rückgabewert von 0 für eine Defektion.

GraciousTFT:

```
def nextMove(self, round, myMoves, opMoves):
    if round == 1:
        return 1                                      # start friendly
    elif round > 6 and ((opMoves[-5:] == [0,0,0,0,0] and \
                         myMoves[-5:] == [0,0,0,0,0]) or \
                        (opMoves[-5:] == [0,1,0,1,0] and \
                         myMoves[-5:] == [1,0,1,0,1])):
        return 1                                      # peace offer
    else:
        if opMoves[-1] == 1: return 1                 # play tit for tat
        else:                return 0
```

Die Strategie *Tester* ist dem Buch von Axelrod entnommen. Sie versucht zunächst durch eine mutwillige Defektion festzustellen, ob sich der Gegner ausnutzen lässt. Wenn ja, dann defektiert sie bei jedem zweiten Zug. Wenn nicht, spielt sie Tit for Tat.

Tester:

```
def nextMove(self, round, myMoves, opMoves):
    if round <= 2:
        return 0
    elif round == 3:
        if opMoves[-1] == 0:  self.state = "TFT"
        else:                 self.state = "Deceiver"
        return 1
    elif round == 4:
        return 1
    else:
        if self.state == "TFT":
            if opMoves[-1] == 1:  return 1
            else:                 return 0
        else:
            if round % 2 == 1:    return 0
            else:                 return 1
```

Analyst schließlich analysiert jeweils die zehn letzten Spielzüge, um festzustellen, ob die gegnerische Strategie ausnutzbar oder böswillig ist. Erweist sie sich als ausnutzbar, so defektiert *Analyst*. Ist der Gegner böswillig, dann wehrt sich *Analyst* ebenfalls durch Defektion. Wenn beides nicht eindeutig feststellbar ist, dann spielt Analyst Tit for Tat. Die ersten zehn Züge werden zufällig gewählt.

Analyst:

```
def nextMove(self, round, myMoves, opMoves):
    if round <= 10:
        return whrandom.randint(0, 1)   # play random at the beginning
    else:
```

```
# analyse

ex_attempt, ex_success = 0,0
opex_opportunity, opex_attempt = 0, 0

i = -9
while i <= -1:
    if myMoves[i-1] == 0:
        ex_attempt += 1
        if opMoves[i] != 0: ex_success += 1 # opponent did
                                            # not punish exploit!
    else:
        opex_opportunity += 1
        if opMoves[i] == 0: opex_attempt += 1  # opponnent played
                                               # defective without
                                               # reason
    i += 1

# and react accordingly

ret = -1
if (ex_attempt > 0):
    if (float(ex_success) / float(ex_attempt)) >= 0.6:
        return 0                    # keep exploiting
    else: ret = 1                   # try to be friendly again

if opex_opportunity > 0:
    if (float(opex_attempt) / float(opex_opportunity)) <= 0.4:
        return 1            # opponent isn't really bad
    else:
        return 0            # opponent tried to deceive to often
else:
    if ret != -1: return ret      # fallback
    else:
        if opMoves[-1] == 1: return 1   # play TFT if clueless
        else:                return 0
```

3.1.2 Die Ergebnisse des Computerturniers

Das Turnier wurde zunächst mit denselben Parametern wie bei Axelrod (Auszahlungsparameter T=5, R=3, P=1, S=0; Rundenzahl w=200) fünfmal mit allen Spielern durchgeführt. In vier Fällen gewann *Tester* das Turnier, einmal jedoch gewann *Analyst*. Der unterschiedliche Ausgang des Turniers erklärt sich dadurch, dass einzelne Strategien (Pseudo-)Zufallszahlen verwenden. Besonders bei *Analyst* führt dies zu starken Schwankungen zwischen den einzelnen Durchläufen. Bei zwei Durchläufen gelang es *Analyst* daher die Strategie *MassiveResponse* (zwei Defektionen zur Strafe für eine Defektion des Gegners) richtig einzuschätzen und auf ein überwiegend kooperatives Spielverhalten einzuschwenken, während *Tester* gegen *MassiveResponse* regelmäßig schlecht spielte. Dadurch konnte sich *Analyst* in einem Fall den entscheidenden Vorsprung sichern und verfehlte den Sieg im anderen Fall nur knapp.

Die typische Rangfolge sieht jedoch folgendermaßen aus:

Turnier 2

```
Tester:                 6025
GraciousTFT:            5981
Tit for tat:            5912
Analyst:                5750
DelayedTFT:             5567
Tit for two tats:       5557
MaliciousTFT:           5461
Cheater:                5407
Random:                 4893
Always friendly:        4764
MassiveResponse:        4748
Utterly destructive:    3748
```

In allen Durchläufen lag *GraciousTFT* vor *Tit for Tat*, was mit dem oben beschriebenen Problem von *Tit for Tat* zusammen hängen mag. Ein Blick auf den Spielablauf verdeutlicht dies:

Tit for Tat : MaliciousTFT

```
1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1
1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1
1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1 1:0 0:1
1:0 0:1...
```

(1=Kooperation, 0=Defektion)

Im Vergleich dazu der Beginn der Partie *GraciousTFT* gegen *MaliciousTFT*:

GraciousTFT : MaliciousTFT

```
1:0 0:1 1:0 0:1 1:0 0:1 1:0 1:1 1:1 1:1 1:1 1:1 1:1 1:1
1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1
1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1 1:1
1:1 1:1...
```

Dass anders als in Axelrods Turnieren *Tester* und nicht *Tit for Tat* Sieger wurde, ist dadurch zu erklären, dass relativ viele Strategien am Turnier teilnahmen (immerhin ein sechstel), die sich sehr leicht ausbeuten lassen, nämlich *Always friendly* und *Tit for two tats*, was von *Tester* weidlich ausgenutzt werden konnte. Streicht man eine dieser beiden Strategien aus dem Turnier, so gerät *Tester* sofort ins Hintertreffen.

Ändert man nun die Paramter für das Turnier ein wenig ab, indem man als Belohnung für wechselseitige Kooperation R=4 Punkte auszahlt, so ändert sich das Bild: *Tester* fällt zurück, während - wie zu erwarten - *GraciousTFT* den ersten Rang einnimmt. *Tit for Tat* belegt immer noch Platz zwei, was immerhin für eine gewisse Stabilität dieser Strategie gegenüber einer Veränderung der Turnierbedinungen spricht:

Turnier 6 (T=5, R=4, P=1, S=0)

```
GraciousTFT:          7786
Tit for tat:          7524
Tit for two tats:     7284
Tester:               7077
...
```

Wird statt dessen die Bösartigkeit wieder höher belohnt, so schnellt *Tester* sogleich wieder an die Spitze. Erstaunlich ist, dass auch diesmal *GraciousTFT* vor *Tit for Tat* an Platz zwei steht, woraus vermutungsweise die Schlussfolgerung abgeleitet werden kann, dass *GraciousTFT* eine echte Verbesserung von *Tit for Tat* darstellt:

Turnier 9 (T=7, R=4, P=1, S=0)

```
Tester:               8151
GraciousTFT:          7911
Tit for tat:          7834
Tit for two tats:     7368
...
```

Alles in allem zeigt sich jedoch, dass Varianten von *Tit for Tat* durchaus keine schlechte Strategie darstellen, auch wenn Axelrods Bevorzugung von *Tit for Tat* anhand dieser Resultate nicht nachvollzogen werden kann. Natürlich ist dieses Ergebnis mit Vorsicht zu genießen, da - wie bereits erwähnt - die Simulation bei weitem zu wenig Strategien enthält, um aussagekräftige Untersuchungen zu erlauben.

Die Tatsache, dass *GraciousTFT* besser abschneidet als *Tit for Tat* ist intuitiv sehr einleuchtend. Übertragen auf wirkliche Situationen würde *Tit for Tat*, sobald nur ein Missverständnis auftritt, ein Verhalten herbeiführen, dass einer „Blutrache" verglichen werden könnte, die kein Ende mehr nimmt. Ein Verhalten also, dass sich auf Dauer für alle Beteiligten als sehr unvorteilhaft erweist.

3.1.3 Iteriertes Gefangenendilemma mit variablen Auszahlungen: Lohnt sich die *Heiratsschwindler*-Strategie?

Neben der Abwandlung der konstanten Auszahlungsparameter, wäre noch einer weitere Variante denkbar, bei der die Auszahlungsparamter im Verlauf des Turnieres variabel sind. Zum Beispiel könnten hin- und wieder Spielrunden eingestreut werden, bei denen die Auszahlung gegenüber den gewöhnlichen Runden verdoppelt oder verdreifacht wird.

Bei dieser Abwandlung des Simulationsszenarios - für das sich in der Wirklichkeit ebenfalls leicht Beispiele finden lassen - wäre sogar die Überlegung anzustellen, ob durch diese Variante die Ergebnisse der ursprünglichen Simulation nicht wesentlich in Frage gestellt werden. Ein wichtiges Ergebnis der ursprünglichen Simulation besteht darin, dass im iterierten Gefangenendilemma kooperative Strategien leistungsfähiger sind als destruktive Strategien. Man könnte auch davon sprechen, das im iterierten Gefangenendilemma das Dilemma aufgebrochen wird, indem die Spieler durch die Drohung von Sanktionen in den nachfolgenden Spielzügen zum beiderseitgen Vorteil zur Kooperation angehalten werden. Dies führt zu der (erfreulichen) Schlussfolgerung, dass kooperatives Verhalten zwischen Menschen nicht nur in der Moral sondern bereits im Eigennutz eine Stütze findet, was Hoffnungen hinstsichtlich der Durchsetzungsfähigkeit von kooperativem Verhalten weckt.

Aber hat dieses Ergebnis auch dann noch Bestand, wenn die Höhe der Auszahlungen von Fall zu Fall variiert? Sollte man nicht befürchten, dass die Kooperation gerade dann einbricht, wenn die Auszahlungen besonders hoch und damit der Betrug besonders lohnend ist (Schurz 2001, S.356f.)? Die Strategie, die so vorgeht, dass sie bei niedrigen Auszahlungen kooperativ spielt (sofern die Kooperation gegenseitig ist), bei hohen Auszahlungssummen aber überraschend defektiert, soll im folgenden als *Heiratsschwindler* bezeichnet werden. Die zu untersuchende Frage lautet also: Kann die Strategie *Heiratsschwindler* im iterierten Gefangenendilemma mit variablen Auszahlungen kooperative Strategien wie z.B. *Tit for Tat* verdrängen?

Um die Erfolgsaussichten einer Strategie wie *Heiratsschwindler* richtig einzuschätzen, müsste eine neue Simualtion unter entsprechenden Bedingungen durchgeführt werden. Aber auch ohne eine weitere Simulation lassen sich einige grundsätzliche Überlegungen zu der Frage anstellen, ob kooperative Strategien dem Eindringen von *Heiratsschwindler* hilflos ausgeliefert sind. Dafür kann auf den Begriff der kollektiven Stabilität von Strategien zurückgegriffen werden (vgl. Axelrod 1984, Anhang B).[2]

Eine Strategie A ist kollektiv stabil, wenn jede beliebige andere Strategie B im Wettkampf mit ihr nicht mehr Punkte erhält als Strategie A im Wettkampf mit sich selbst gewinnt. Wenn wir die Punkte, die eine Strategie X in einem Wettkampf mit der Strategie Y erhält, mit V(X/Y) bezeichnen, dann ist Strategie A kollektiv stabil, wenn gilt: $V(B/A) \leq V(A/A)$ für jede beliebige Strategie B.

Da diese Definition der kollektiven Stabilität unabhängig von der Höhe der Auszahlungen ist, lässt sie sich auch auf das iterierte Gefangenendilemma

[2]Der Begriff der *kollektiven Stabilität* ist einer der schwächsten von einer Reihe möglicher Begriffe der *evolutionären Stabilit.* Ein vergleichsweise stärkerer Begriff ist der Begriff der *evolutionären Stabilität* nach Maynard-Smith, der fordert, dass $V(B/A) < V(A/A)$. Auf die Problematik der Begriffsbildung kann an dieser Stelle leider nicht näher eingegangen werden (vgl. dazu Binmore 1998, 319ff., Binmore/Samuelson 1992, 277ff.).

mit variierenden Auszahlungen anwenden. Unsere Frage lautet dann: Gibt es auch bei variierenden Auszahlungen kooperative Strategien, die kollektiv stabil sind? Unter einer kooperativen Strategie verstehe ich dabei eine Strategie, die nicht unmotiviert defektiert, so dass jede hinreichend freundlich gesonnene Gegnerstrategie jederzeit mit ihr kooperieren kann. *Tit for Tat* ist in diesem Sinne eine kooperative Strategie. Wie Axelrod bewiesen hat (Axelrod 1984, Anhang B), ist *Tit for Tat* im iterierten Gefangenendilemma mit gleichbleibenden Auszahlungen kollektiv stabil.

Zu untersuchen ist also die Behauptung, dass sich auch bei variierenden Auszahlungen immer eine kooperative Strategie konstruieren lässt, die kollektiv stabil ist. Um die Wahrheit dieser Behauptung zu zeigen, soll der Einfachheit halber zunächst angenommen werden, dass in jeder fünften Spielrunde die Auszahlung deutlich höher ausfällt als in jeder anderen Spielrunde. Ansonsten sollen die Auszahlungen allerdings immer von gleicher Höhe sein. Nun betrachte man folgende aus zwei Teilstrategien zusammengesetzte Strategie:

1. Für jede Runde mit gewöhnlicher Auszahlung spiele *Tit for Tat*. Nimm dabei Bezug auf die jeweils letzte Runde mit gewöhnlicher Auszahlung (d.h. in der dritten Runde bezieht sich die Reaktion auf das Gegnerverhalten der zweiten Runde, aber in der sechsten Runde auf das der vierten Runde).

2. Für jede durch fünf teilbare Runde spiele ebenfalls *Tit for Tat*, aber beziehe Dich dabei immer auf die jeweils letzte Runde mit durch fünf teilbarer Rundenzahl. (Hat der Gegner also in der fünften Runde defektiert, so folgt die Bestrafung dafür erst in der zehnten Runde, unabhängig davon, wie der Gegner in der neunten Runde gespielt hat.)

Nennen wir nun diese Strategie A und die beiden Teilstrategien der Einfachheit halber A_1 und A_2. Tritt Strategie A im Wettkampf gegen sich selbst an, so gilt für die Gesmthöhe aller Auszahlungen offensichtlich: $V(A/A) = V_1(A/A) + V_2(A/A)$. Dabei bezeichnet V_1 die Summe der Auszahlungen in den Runden 1..4, 5..9, 11..14 usw. und V_2 die Summe der Auszahlungen in den Runden 5, 10, 15, 20, 25 usw. Da A_1 und A_2 unabhängig voneinander sind, gilt ebenso $V(A/A) = V_1(A_1/A_1) + V_2(A_2/A_2)$.

Lassen wir Strategie A nun gegen eine beliebige Strategie B antreten, dann lässt sich die Auszahlung $V(B/A)$ auf folgende Weise zerlegen: $V(B/A) = V_1(B/A) + V_2(B/A)$. Da aber A_1 und A_2 jeweils kollektiv stabil sind (denn es handelt sich ja in beiden Fällen um die Strategie *Tit for Tat* bei gleichbleibenden Auszahlungen) so gilt: $V_1(B/A) \leq V_1(A_1/A_1)$

und $V_2(B/A) \leq V_2(A_2/A_2)$. Dann gilt insbesondere aber auch: $V(B/A) = V_1(B/A) + V_2(B/A) \leq V_1(A_1/A_1) + V_2(A_2/A_2) = V(A/A)$. Nach der Definition ist Strategie A damit kollektiv stabil. Insbesondere kann dann auch die Strategie *Heiratsschwindler* nicht in Strategie A eindringen. Entscheidend für die Konstruktion der kollektiv stabilen kooperativen Strategie A ist lediglich, dass vorher bekannt ist, wann eine besonders hohe Auszahlung erfolgt. Dies müssen wir allerdings voraussetzen, da sonst auch die *Heiratsschwindler*-Strategie nicht konstruiert werden könnte. Weiterhin dürfen die besonders hohen Auszahlungen nicht zu selten erfolgen, da es sonst zu demselben Phänomen kommt, das auch bei einer nur sehr geringen Anzahl von Spielrunden auftritt, und das darin besteht, dass Defektionen nicht mehr wirksam bestraft werden können.

Aufgrund der vereinfachenden Annahmen, dass es nur zwei unterschiedliche Auzsahlungsniveaus gibt, wobei die höhrere Auszahlung periodisch in jedem fünften Zug auftritt, behandelt der eben geführte Beweis bisher lediglich einen Spezialfall. Eine Verallgemeinerung des Beweises ist folgendermaßen möglich: Wir nehmen statt zwei unterschiedlicher Auszahlungshöhen eine Menge von k unterschiedlichen Auszahlungshöhen $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$ an. Dabei steht jedes $\mathcal{T}_i$ mit $(1 \leq i \leq k)$ für ein Tupel (T_i, R_i, P_i, S_i) von Auszahlungsparametern, die den beiden Bedingungen $T_i > R_i > P_i > S_i$ und $T_i + S_i < 2R_i$ genügen.[3]

Weiterhin nehmen wir an, dass die Folge der Spielrunden $\mathcal{R} = r_1, r_2, \ldots, r_n$ aus Teilfolgen $\mathcal{R}_i$ mit $(1 \leq i \leq k)$ von hinreichender Länge (mindestens zwei Züge) zusammengesetzt ist, in denen die Auszahlung jeweils entsprechend den Auszahlungsparametern $\mathcal{T}_i$ erfolgt. Es wird voraussgesetzt, dass die Strategien in jeder Runde wissen, nach welchem Tupel $\mathcal{T}_i$ die Auszahlungen erfolgen, d.h. welcher Teilfolge diese Runde zugeordnet ist. Die Strategien verfügen jedoch nicht über die Information, wann die letzte Runde einer Teilfolge erreicht ist.

Nun lässt sich ganz analog zu dem oben gegebenen Beweis für den vereinfachten Fall eine Strategie A konstruieren, die über die Teilfolgen $\mathcal{R}_i$ jeweils *TitForTat* spielt. Für die Auszahlung, welche die Strategie $V(A/A)$ im Spiel gegen sich selbst erhält, gilt $V(A/A) = \sum_{i=1}^{k} V_i(A_i/A_i)$, wobei V_i für die Gesamtauszahlung über die Teilfolge $\mathcal{R}_i$ steht und die Teilstrategie A_i als

[3] Die erste Bedingung ist die Voraussetzung dafür, dass es sich um ein Gefangenendilemma habndelt. Die zweite Bedingung stellt sicher, dass eine dauerhafte wechselseitige Kooperation gewinnbringender für die Spieler ist als eine Folge von Spielzügen, bei der immer abwechselnd ein Spieler kooperiert, während der andere defektiert. Letzteres würde das Modell grundlegend verändern und Rahmenbedingungen schaffen, in der sich die abwechselnde gegenseitige Beschädigung der Spieler als optimale kooperative Struktur herausbilden würde.

TitForTat über die Teilfolge $\mathcal{R}_i$ der Spielrunden definiert ist. Wie im vereinfachten Fall lässt sich für jede denkbare Angreiferstrategie B die Ungleichung aufstellen:

$$V(B/A) = \sum_{i=1}^{k} V_i(A/B) \leq \sum_{i=1}^{k} V_i(A_i/A_i) = V(A/A)$$

womit A kollektiv stabil ist. Folglich existiert auch bei nicht periodisch und in unterschiedlicher Höhe variierenden Auszahlungen keine *Heiratsschwindler*-Strategie, die in A eindringen kann.

3.2 Erweiterung zur populationsdynamischen Simulation

Der Turniererfolg einer bestimmten Strategie sagt noch nicht zwingend etwas über den langfristigen Erfolg dieser Strategie im evolutionären Wettbewerb aus. Um diese Frage zu untersuchen, soll nun das ursprüngliche Computerturnier in einem zweiten Schritt zu einer populationsdynamischen Simulation ausgebaut werden.

Dazu wird mit einer Population von zehntausend Spielern, die auf die gegebene Menge von Strategien gleichmäßig verteilt werden, eine Serie von Turnieren durchgespielt, wobei diejenigen Strategien, die sich als besonders erfolgreich erweisen, in der jeweils folgenden Runde von einer größeren Anzahl von Spielern angewendet werden, während die weniger erfolgreichen Strategien eine geringere Anzahl von Spielern zugewiesen bekommen, bis sie möglicherweise irgendwann ganz „aussterben".[4] Es würde an dieser Stelle zu weit führen, auf alle technischen Einzelheiten dieser Simulation detailliert einzugehen.

[4] Die Neuverteilung der Spielerpopulation auf die Strategien wird dabei nach folgender Vorschrift bestimmt: Zunächst wird der Mittelwert der Resultate aller Strategien berechnet. Dann wird für jede Strategie der Quotient aus ihrer Punktzahl und dem Mittelwert der Punktzahlen aller Strategien gebildet. Der ermittelte Wert dient als Faktor für die Größenänderung der Population der Strategie. In einem letzten Schritt werden die Populationen aller Strategien soweit skaliert, dass ihre Summe wieder einer vorgegebenen Gesamtpopulation von 10.000 Individuen entspricht, wobei kleine Abweichungen zugelassen werden, um Rundungsfehler möglichst zu vermeiden. Strategien, von denen nur noch weniger als zwei Individuen übrig geblieben sind, werden aus dem Rennen gezogen.

Dieses Verfahren zur Ermittlung der Neuverteilung der Individuen ist mehr oder weniger willkürlich gewählt, genügt aber der Bedingung, dass eine Strategie sich umso erfolgreicher vermehrt, je größer ihre Punktzahl ist. Natürlich könnte man ebenso gut auch jede andere streng monoton steigende Abbildung der erreichten Punktzahlen auf die Größenänderungen der Populationen wählen. In den meisten Fällen dürfte dies nur zu einer Beschleunigung oder Verlangsamung der zu beobachtenden Phänomene führen.

Lediglich eine der (gegenüber der im letzten Abschnitt beschriebenen Simulation geringfügig erweiterten) Strategiemenge neu hinzugefügte Strategie soll kurz vorgestellt werden, da sie in der populationsdynamischen Simulation unter bestimmten Bedingungen sehr erfolgreich ist.

Die Strategie *Pawlow* beginnt mit zwei Defektionen und ändert nur dann ihr Verhalten, wenn sie vom Gegner bestraft wird (d.h. wenn der Gegner mit einer Defektion antwortet), wobei es keine Rolle spielt, ob sie für kooperatives oder unkooperatives Verhalten „bestraft" worden ist. Als Sicherheitsmechanismus (um „Missverständnisse" zu vermeiden) ändert sie ihre Strategie aber nur höchstens jede zweite Runde.

Pawlow:

```
def nextMove(self, round, myMoves, opMoves):
    if round <= 2:
        return 0                                    # be naughty
    else:
        if opMoves[-1] == 0 and myMoves[-1] == myMoves[-2]:
            if myMoves[-1] == 0:    return 1        # learned something
            else:                   return 0
        else:
            return myMoves[-1]
```

3.2.1 Die Ergebnisse der populationsdynamischen Simulation

Das Ergebnis der populationsdynamischen Simulation entspricht den Erwartungen (Abbildung 1). Die Strategie *GraciousTFT* setzt sich bald an die Spitze und dominiert das Feld, dicht gefolgt von *Tit for Tat*. Dass zwischen beiden Strategien ein Abstand bleibt, obwohl sie nach dem Verschwinden der meisten anderen Strategien stets die gleiche Durchschnittspunktzahl erziehlen, hängt mit der Verteilungsfunktion zusammen, die Verschiebungen zwischen den Strategien nur bei unterschiedlicher Punktzahl erlaubt und damit einen einmal errungenen Vorteil einer gleich starken Strategie konserviert.

Auffällig ist das kontinuierliche Absinken der anfangs sehr starken Strategie *Tester*. Dies ist dadurch zu erklären, dass nach dem Verschwinden der meisten Strategien keine gegnerische Strategie mehr vorhanden ist, die von *Tester* ausgebeutet werden kann. Durch seinen Ausbeutungsversuch in den ersten beiden Runden schneidet *Tester* dann immer etwas schlechter ab als *Tit for Tat* und *GraciousTFT*.

Ein ähnliches Problem betrifft auch *Pawlow*, wobei es allerdings erstaunlich ist, das die im Turnier eher mittelmäßige Strategie *Pawlow* überhaupt so lange durchhalten kann. Letzteres dürfte darauf zurückzuführen sein, dass die Strategie *Pawlow* über die Fähigkeit verfügt, sich relativ gut an ihr Millieu anzupassen.

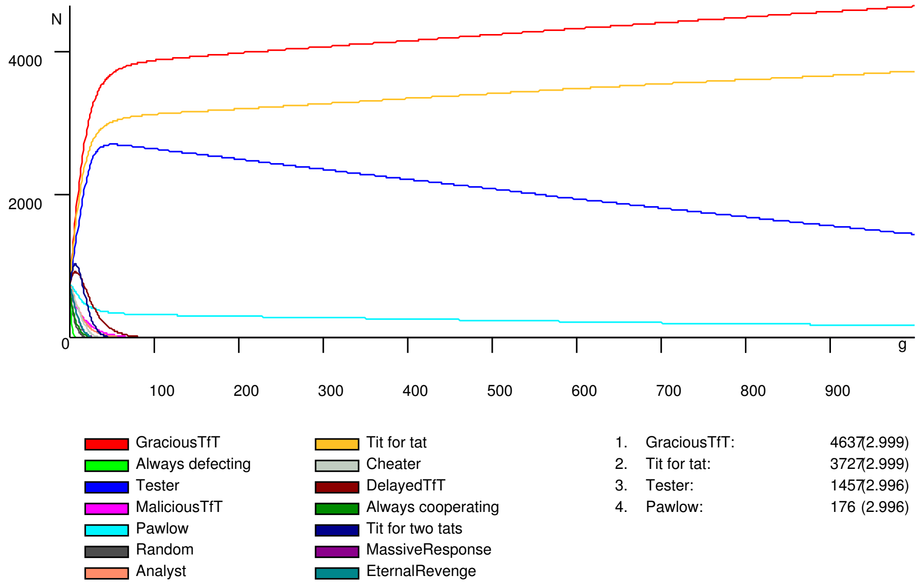

Abbildung 1: Populationsdynamische Simulation ohne Mutationen oder Rauschen.

Diese Fähigkeit bewährt sich besonders dann wenn die populationsdynamische Simulation unter erschwerenden Bedingungen durchgeführt wird, wie die folgenden Beispiele zeigen.

3.2.2 Der Einfluss von Rauschen auf die Populationsdynamik

Es ist unrealistisch anzunehmen, dass bei einer so großen Anzahl von Interaktionen zwischen den Spielern, wie sie bei der populationsdynamischen Simulation angenommen wird, nicht gelegentlich auch Fehlleistungen und Missverständnise auftreten sollten. Um diesen Aspekt zu simulieren wird ein Rauschparameter in die Simulation eingefügt. Dabei werden mit einer bestimmten Wahrscheinlichkeit die Züge der Spieler ins Gegenteil verkehrt, d.h. einem Spieler, der kooperativ spielen wollte, wird eine Defektion untergeschoben und umgekehrt. Alles kommt nun darauf an, ob die Strategien in der Lage sind solche Fehler wieder auszugleichen.

Bei einem 1%-igen Rauschen (Abbildung 2) können sich nur *GraciousTFT* und *Pawlow* längerfristig erfolgreich behaupten können. Beide Strategien verfügen über Mechanismen (siehe dazu die Beschreibungen der Strategien weiter oben), die ihnen eine gewisse Fehlertoleranz verleihen. Das Ergebnis bestätigt auch die eben ausgesprochene Vermutung über die Anpassungsfähigkeit von *Pawlow*. Zu Erwähnen ist allerdings, dass bei einer Erhöung des

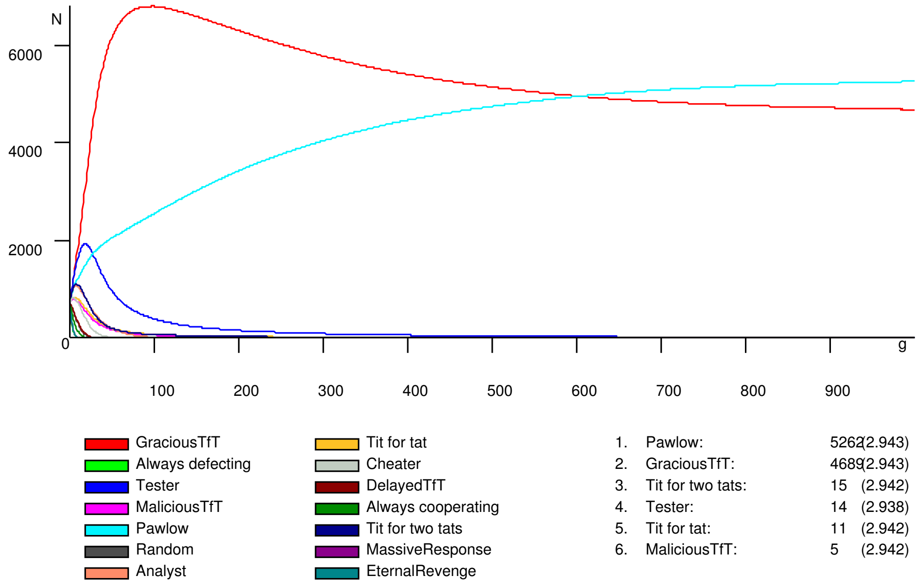

Abbildung 2: Populationsdynamische Simulation mit Rauschen (1%).

Rauschparameters auf 5% der Vorteil für *Pawlow* verschwindet, so dass diese Strategie kaum besser abschneidet als die anderen Strategien mit Ausnahme von *GraciousTFT*.

3.2.3 Der Einfluss von Mutationen

Obwohl die populationsdynamische Simulation einer evolutionären Entwicklung schon nahe kommt, fehlt in diesem Modell noch ein wichtiger Aspekt. Zwar ändert sich die Strategiemenge im Laufe der Simulation durch das Wegfallen erfolgloser Strategien, aber es kommen keine neuen Strategien hinzu. Ein erster Ansatz, um diesen wesentlichen Aspekt evolutionärer Entwicklungen in das Modell zu integrieren, besteht darin, einen Mutationsparameter einzuführen, der nach jedem Durchgang die Mutation eines bestimmten Prozentsatzes der Strategien bewirkt. Dabei wird der Einfachheit halber zunächst angenommen, dass durch Mutationen nur vereinfachte Strategien entstehen können, nämlich *AlwaysD* (defektiere immer) und *AlwaysC* (kooperiere immer), die beide jeweils die Hälfte der mutierten Strategien stellen.

Im Vergleich zur ursprünglichen populationsdynamischen Simulation ohne Mutationen ergibt sich mit Mutationen ein dramatisch verändertes Bild (Abbildung 3). Diesmal wird die Szene von den beiden Strategien *Tester* und *Pawlow* beherrscht, die von dem kontinuierlichen Zustrom des ohne weiteres ausbeutbaren Mutanten *AlwaysC* profitieren. Demgegenüber geraten

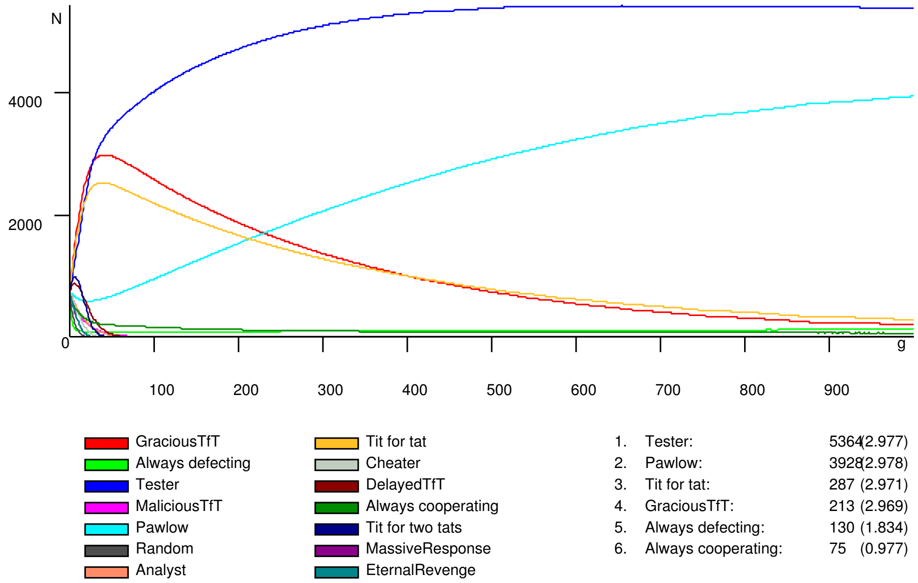

Abbildung 3: Populationsdynamische Simulation mit Mutationen (1%).

Tit for Tat und *GraciousTFT*, die keine unmotivierten Defektionen unternehmen und daher *AlwaysC* auch nicht auszubeuten versuchen, deutlich ins Hintertreffen. Wie man sieht, ist der evolutionäre Erfolg von „gutwilligen“ Strategien wie *Tit for Tat* oder *GraciousTFT* daran geknüpft, dass sie in einer Umwelt agieren, in der keine „degenerativen Mutationen“ auftreten, die die Population unterwandern. Wie wahrscheinlich eine solche Annahme ist, hängt letztlich von dem Sachbereich ab, der mit einem evolutionären Modell beschrieben werden soll. Bereits so lässt sich aber festhalten, dass einige der von Axelrod getroffenen generellen Feststellungen wie diejenige, dass die Strategie *Tit for Tat* immer eine sichere Wahl sei, oder dass erfolgreiche Strategien in aller Regel „gutwillig“ sein müssen, sich in dieser Allgemeingültigkeit nicht aufrecht erhalten lassen.

3.3 Möglichkeiten und Grenzen von Computermodellen bei der Untersuchung evolutionärer Prozesse

Um die Erklärungskraft von Computermodellen bei der Untersuchung evolutionärer Vorgänge im Bereich der Gesellschaftswissenschaften richtig einzuschätzen, muss man berücksichtigen, dass in diesem Bereich die relevanten Einflüsse selten mit hinreichender Genauigkeit bestimmt oder auch nur vollständig benannt werden können.[5] Von Ausnahmefällen abgesehen dürfte es daher kaum möglich sein, Computermodelle zu erstellen, die genaue Prognosen ermöglichen.

Dennoch kann der Einsatz von Computermodellen sinnvoll sein, um die Muster zu studieren, nach denen evolutionäre Prozesse ablaufen. Eine gewisse Vorsicht ist jedoch geboten, wenn aus einer Computersimulation verallgemeinernde Schlussfolgerungen gezogen werden sollen. Sonst besteht die Gefahr, dass zufällige, d.h. von der Wahl bestimmter Paramter abhängige Simulationsergebnisse voreilig zu allgemeinen Regeln hypostasiert werden.

Aus heutiger Sicht erscheint der Ansatz von Axelrods „Evolution der Kooperation“, der hier einmal als Beispiel nachvollzogen wurde, gerade in dieser Hinsicht noch als zu unvorsichtig, um nicht zu sagen naiv. So beruht die Tatsache, dass in den beiden aufeinanderfolgenden Turnieren, die Axelrod in seinem Buch beschreibt (Axelrod 1984, Kap. 2), jedesmal die Strategie *Tit for Tat* als Sieger hervorging, wahrscheinlich nur auf Zufall. Denn der Erfolg von *Tit for Tat* hängt nicht zuletzt von der Auswahl der Strategiemenge ab, die bei Axelrod hochgradig kontingent ist, indem sie durch eine Art von Preisausschreiben bestimmt wurde. Versucht man systematischer vorzugehen und als Strategiemenge alle Strategien mit einer gewissen Komplexität zu Grunde zu legen, also beispielsweise alle Strategien, die sich als endliche Automaten darstellen lassen, die bis zu zwei Zustände speichern können, dann gewinnt mitnichten *Tit For Tat*. Wie die Abbildung 4 zeigt, geht in diesem Falle vielmehr die Strategie *Grim* (Ewige Vergeltung) als Sieger aus dem Wettkampf hervor (vgl. auch Binmore 1998, 322). Allerdings ist auch damit noch nicht das letzte Wort gesprochen, denn es könnte nun wiederum eingewandt werden, dass die Menge aller ein- und zweistufigen Automaten keine sinnvolle Ausgangsbasis ist, da sie eine ungewöhnlich große Anzahl ausgesprochen „dummer“ Strategien enthält, die auch auf ständige Defektion mit fortgesetzter Kooperation reagieren, was kaum realistisch erscheint.

Diese Einwände verdeutlichen, dass die Interpretation der Ergebnisse von Computersimulation keineswegs trivial ist und durch theoretische Überlegungen abgestützt werden muss.

[5] Wie wollte man beispielsweise die Agressivität eines Staates beziffern? Agressivität kann sich in militärischer Aufrüstung ebenso ausdrücken wie in rhetorischem „Säbelrasseln“ der Regierung. Wie das Beispiel zeigt, ist es oft noch nicht einmal möglich, eine auch nur halbwegs genaue Ordnungsrelation („agressiver als“) aufzustellen.

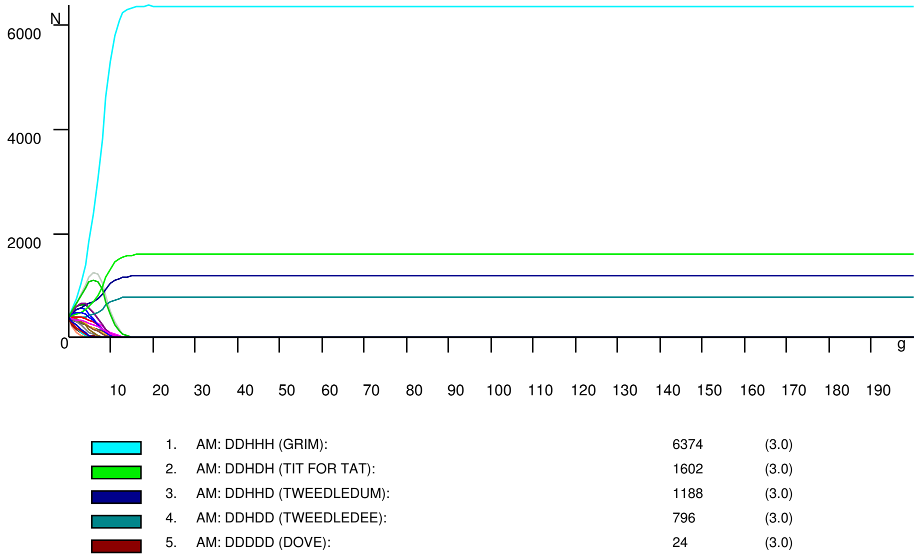

Abbildung 4: Populationsdynamische Simulation mit endlichen Automaten.

4 Beispiele für evolutionäre Erklärungsansätze im Bereich der Kulturwissenschaften

4.1 Die evolutionäre Erklärung historischer Prozesse

Auch wenn, wie in der Einleitung bereits angemerkt, Metaphern aus der Biologie und besonders der Evolutionstheorie Eingang in die Geschichtsschreibung gefunden haben, stehen die langsam ablaufenden Prozesse evolutionärer Entwicklungen für gewöhnlich nicht im Mittelpunkt des Interesses von Fachhistorikern. Die Gründe dafür sind zweierlei: Zum einen befasst sich die Historie tradionellerweise vorwiegend mit der politischen Geschichte und teilt die Epochen daher nach politischen Großereignissen wie Kriegen, Revolutionen, der Gründung und dem Untergang von Staaten und Reichen ein. Solche Ereignisse lassen sich genauer datieren und sind leichter fassbar als die schleichend verlaufenden evolutionären Entwicklungsprozesse. Historiker betrachten die Ergebnisse solcher Prozesse wie z.B. das technische Niveau in einer bestimmten Epoche daher eher als historische Rahmenbedinungen, die als gegeben vorausgesetzt werden müssen. Zum anderen scheint es häufig an einer geeigneten konzeptionellen Grundlage zu fehlen, um evolutionäre Prozesse erfassen und sie als solche verstehen zu können. Dieser Mangel äußert sich manchmal darin, dass evolutionäre Prozesse gewaltsam zu zeitlich leicht datierbaren historischen Schlüsselereignissen umgebogen werden, wie

Eric Jones dies am Beispiel der sogenannten „industriellen Revolution“ kritisiert (Jones 1988).

Im folgenden werde ich kurz zwei erfolgreiche Beispiele für evolutionäre Erklärungen in der Geschichtswissenschaft anführen. Eines davon stammt charakteristischerweise nicht von einem Historiker sondern von einem Biologen.

4.1.1 Die neolithische „Evolution“ (J. Diamond)

In seinem Buch „Guns, Germs and Steel“ (Diamond 1998) unternimmt Jared Diamond den anspruchsvollen Versuch, die zivilisatorische Entwicklung menschlicher Gesellschaften (und insbesondere den Entwicklungsvorsprung der eurasischen Gesellschaften) aus natürlichen Umweltbedingungen ihrer geographischen Region zu erklären. So erklärt Diamond beispielsweise die Geschwindigkeit, mit der Ackerbau und Viehzucht entwickelt wurden und sich ausbreiten konnten, aus dem Vorhandensein (oder eben Nicht-Vorhandensein) von Pflanzen und Tieren die zur Domestikation geeignet waren, und damit ob sich diese Kulturtechniken entlang einer Ost-West-Achse (Eurasien) oder entlang einer Nord-Süd-Achse (Amerika), die sich über unterschiedliche Klimazonen ersteckt, ausbreiten mussten. Diamonds Erklärung erscheint deshalb so zwingend, weil sie zeigt, dass die Entwicklung grundlegender zivilisatorischer Errungenschaften von Umweltbedingungen abhängig ist, deren Fehlen auch der größte menschliche Erfindungsgeist nicht ausgleichen kann.

Inwiefern handelt es sich dabei um einen evolutionären Erklärungsansatz? Die Wege, auf denen sich Kulturtechniken ausbreiten und selektiv gegen andere Kulturtechniken durchsetzen (kriegerische Verdrängung, kulturelles Lernen), wurden bereits angesprochen. Entscheidend ist außerdem, dass es ein hinreichend großes Versuchfeld gab, indem unterschiedliche Kulturtechniken erprobt und verfeinert werden konnten. Doch auch dies ist unproblematisch, da Diamond relativ lange Zeitspannen betrachtet, und zudem geographische Räume behandelt, in denen oft eine Vielzahl untereinander mehr oder weiniger lose verbundener Gesellschaften existierten. Insbesondere gelingt es Diamond auch die Zwischenstadien der evolutionären Entwicklung zu identifizieren, indem er z.B. nachweist, dass auch Völker, die noch keinen Ackerbau betreiben, in manchen Fällen als mögliche Vorstufe des Ackerbaus bereits Nutzgärten pflegen.

Von besonderem Interesse ist Diamonds Darstellung der Evolution der politischen Verbände vom Stammesverband bis zum großen Flächenstaat. Großstaaten stützen sich auf eine Reihe von politischen, religiösen, wirtschaftlichen, rechtlichen und militärischen Insitutionen, die in ihrem Zusammenspiel so komplex sind, dass sie in ihrer Gesamtheit kaum von genialen Reichsgründern oder Zivilisationsstiftern erfunden worden sein können. Obwohl Diamond für die schrittweise Entwicklung der großen Staatswesen eine relativ schlüssige Typologie anbietet, bleibt dieses Gebiet innerhalb seiner Theorie am ausbaufähigsten. Die Nachzeichnung der evolutionären Prozesse die zur Entwicklung der komplizierten Institutionengefüge von Staaten geführt haben, ist so gesehen noch eine Herausforderung. (Auch wenn in dieser Hinsicht z.B. von Max Weber und dessen Nachfolgern einiges geleistet worden ist. Aber gerade diese Vorleistungen ließen sich mit Hilfe evolutionärer Erklärungsansätze womöglich noch verfeinern.)

4.1.2 „Das Wunder Europas“ (Eric Lionel Jones)

Ähnlich wie Diamond versucht auch Eric Jones den historischen „Erfolg“ Europas zu erklären (Jones 1991). Anders als dieser schreibt er jedoch nicht die Geschichte der Menschheit seit den letzten 13.000 Jahren, sondern er beschränkt sich auf die jüngste Zeit vom Mittelalter bis zur Gegenwart. Bei seiner Erklärung misst er dem schleichenden Prozess stetiger wirtschaftlicher und technischer Entwicklung wesentlich größere Bedeutung bei als vermeintlichen historischen Durchbrüchen wie der Entdeckung der Gravitationskraft oder der industriellen Revolution. Ohne dies immer explizit zu erwähnen, legt Jones dabei ein evolutionäres Erklärungsschema zu Grunde. Demnach kam dem europäischen Kontinent gerade seine politische Kleinräumigkeit zu Gute. Diese ermöglichte es, neue Errungenschaften gleich welcher Art gewissermaßen evolutionär auszuprobieren. Wenn ein Erfinder oder ein Entdecker bei einem bestimmten Herrscher kein Gehör fand, so brauchte er nicht weit zu reisen, um einen anderen zu finden, der vielleicht größeres Interesse zeigte, und dafür im günstigsten Fall mit einer Erfindung belohnt wurde, die ihm einen wichtigen Vorteil gegenüber seinen Rivalen sichern konnte. Aus demselben Grund konnten es sich die Herrscher in Europa auch nicht erlauben den wirtschaftlich potenten Schichten, deren Angehörige nur eine Grenze überschreiten mussten um sich der Willkür eines bestimmten Herrschers zu entziehen, bis auf den letzten Blutstropfen Steuern abzupressen. In zentralisierten Staaten wie dem chinesischen Kaiserreich verhielt sich dies anders: Hier konnte der Herrscher eine neue Errungenschaft ohne Mühe per Dekret verbieten, wenn sie ihm nicht zusagte, und Jones führt zahlreiche Beispiele auf, wo dies auch der Fall gewesen ist (Jones 1991, 77, 231ff.).

Natürlich bildet dieser Gesichtspunkt nur einen Teilaspekt von Jones Erklärung. Die Nationalstaatsbildung und der Aufstieg der Handel treibenden Klassen waren für das „Wunder Europa“ von kaum gringerer Bedeutung. Dennoch läßt sich aus Jones Darstellung etwas vereinfachend die Quintessenz ziehen, dass Europa gerade deshalb so erfolgreich war, weil hier evolutionäre Trial and Error-Prozesse wirksam werden konnten.[6]

4.2 Evolutionäre Stabilität ethischer Normen

Zu guter Letzt soll noch eine Anwendungsmöglichkeit evolutionärer Theorien im Bereich der Ethik diskutiert werden. Dass evolutionäre Theorien herangezogen werden können, um die Entstehung von moralischen und rechtlichen Normen empirisch zu erklären, bedarf nach dem bisher gesagten keiner weiteren Begründung mehr. Dass ein evolutionärer Ansatz aber auch bei der Diskussion der philosophischen Frage weiter helfen kann, welche Normen denn nun gelten *sollen*, erscheint auf den ersten Blick alles andere als einleuchtend. Schließlich gilt es seit David Hume als ein ehernes Gesetz, dass man nicht von Tatsachen auf Normen schließen kann, so daß insbesondere die Tatsache, dass sich irgendwelche Normen evolutionär durchgesetzt haben, keineswegs besagt, dass diese Normen auch gut und richtig sind.

Dennoch kann der Gegensatz zwischen Sein und Sollen überbrückt werden, wenn man ein (zwangsläufig) normatives Prinzip einführt, das einen Zusammenhang zwischen diesen beiden Ebenen ausdrücklich herstellt. In der Tat enthalten fast alle vorkommenden ethischen Systeme mit Ausnahme solcher, die man im Sinne Max Webers als rein „gesinnungsethisch" charakterisieren müsste, explizit oder implizit derartige Prinzipien. Gerade im Bereich der politischen Moral wird gemeinhin ein Prinzip angenommen, das besagt, dass keine politisch-ethische Norm Gültigkeit beanspruchen kann, deren Befolgung der Selbstaufgabe des eigenen Landes gleichkommt.[7] Man könnte dieses Prinzip das *Prinzip der Realitätsadäquatheit* nennen.

Der Begriff der evolutionären (bzw. kollektiven) Stabilität könnte nun zur näheren Konkretisierung dieses Prinzips verwendet werden. Dadurch wäre zweierlei gewonnen: Zum einen würde der Fehlschluss vermieden werden, dass dem Realitätsadäquatheitsprinzip nur durch ein besonders rücksichtsloses und nationalegoistisches Verhalten genüge getan werden könnte (ein Fehlschluss zu dem die Schule des „Politischen Realismus" manchmal neigt), da es eine gewisse Bandbreite evolutionär stabiler Verhaltensweisen gibt, darunter

[6] In ähnlicher Weise wird übrigens oft auch der Erfolg des US-amerikanischen Föderalismus gegenüber dem europäischen (und erst recht dem bundesdeutschen) Föderalismus gedeutet. Im amerikanischen Föderalismus herrscht gerade das richtige Maß an Unabhängigkeit der einzelnen Staaten, welches das Ausprobieren unterschiedlicher Wege ermöglicht. Zugleich gibt es kaum Barrieren (wie etwa die Sprachbarrieren in Europa), die der raschen Verbreitung erfolgreicher Vorbilder im Wege stehen.

[7] Das soll nicht bedeuteten, dass es nicht auch Extremfälle gibt, in denen die Selbstaufgabe des eigenen Landes das kleinere von zwei Übeln ist.

- wie zuvor gezeigt - auch solche die grundsätzlich eher kooperativ ausgerichtet sind.[8] Zum anderen bleibt es möglich, allzu illusorische oder idealistische Normforderungen wie z.B. einen kategorischen Pazifismus wohlbegründet abzuweisen.

[8] Durch das Kriterium der evolutionären Stabilität wird also die politsch-ethische Entscheidung nicht schon erübrigt, womit den Entscheidungsträgern freilich auch die Übernahme von Verantwortung (für ihre normative Wahl) nicht erspart bleibt.

5 Zitierte Literatur

Axelrod, Robert (1984): Die Evolution der Kooperation, Oldenbourg, München (5 Aufl. 2000; engl. Original 1984).

Axelrod, Robert (1997): The Complexity of Cooperation. Agent-Based Models of Competition and Collaboration, Princeton University Press, Princeton.

Binmore, Ken / Samuelson, Larry (1992): Evolutionary Stability in Repeated Games Played by finite Automata, in: Journal of Economic Theory 57 (2/1992), 278-305.

Binmore, Ken (1994): Game Theory and the Social Contract I. Playing Fair, MIT Press, Cambridge (Massachusetts), London (England) (4. Nachdruck 2000).

Binmore, Ken (1998): Game Theory and the Social Contract II. Just Playing, MIT Press, Cambridge (Massachusetts), London (England).

Diamond, Jared (1998): Guns, germs and steel: a short history of everybody for the last 13000 years, Vintage Random House, London.

Jones, Eric L. (1981): Das Wunder Europa : Umwelt, Wirtschaft und Geopolitik in der Geschichte Europas und Asiens, J.C.B. Mohr, Tübingen (deutsch 1991, engl. Original 1981).

Jones, Eric L. (1988): Growth Reccurring. Economic Change in World History, Oxford.

Koch, Hannsjoachim (1973): Der Sozialdarwinismus. Seine Genese und sein Einfluß auf das imperialistische Denken, C.H. Beck, München.

Maynard-Smith, John (1982): Evolution and the Theory of Games, Cambridge Univ. Press, Cambridge (8. Aufl. 2000).

Schurz, Gerhard (2001): Natürliche und kulturelle Evolution: Skizze einer verallgemeinerten Evolutionstheorie, in: Wickler, Wolfgang / Salwiczek, Lucie: Wie wir die Welt erkennen. Erkenntnisweisen im interdisziplinären Diskurs, München.

Schüssler, Rudolf (1990): Kooperation unter Egoisten: vier Dilemmata, R.Oldenbourg Verlag, München (2.Aufl. 1997)

Wagner, Günter P. (1994): Der Dialog zwischen Evolutionsforschung und Computerwissenschaft, in: Wieser (1994a, Hg.), 221-233.

Wieser, Wolfgang (1994a, Hg.): Die Evolution der Evolutionstheorie. Von Darwin zur DNA, Wissenschafttliche Buchgesellschaft, Darmstadt.

Wieser, Wolfgang (1994b): Gentheorien und Systemtheorien. Wege und Wandlungen der Evolutionstheorie im 20. Jahrhundert, in: Wieser (1994a, Hg.), 15-48.

6 Anhang: Programmcode des Computerturniers

Das Programm für das Computerturnier (Kapitel 3.1) ist in zwei Module (*Tournament.py* und *Strategy.py*) aufgeteilt. Das Modul *Tournament.py* enthält die beiden Klassen *Match* und *Tournament*, wobei *Match* ein Spiel zwischen zwei Spielern steuert, während *Tournament* den Ablauf des gesamten Turniers festlegt und dazu für jede mögliche Spielerpaarung ein Objekt der Klasse *Match* erzeugt (siehe die Methode *runTournament* der Klasse *Tournament*).

Das Modul *Strategy.py* enthält die Strategien, die am Turnier teilnehmen können. Alle Strategien sind als abgeleitete Klassen der abstrakten Basisklasse *Strategy* definiert. Sie enthalten als einzige Methode die Methode *nextMove*, der die Nummer des gegenwärtigen Zuges sowie die Listen aller bisherigen eigenen Züge und aller bisherigen gegenerischen Züge übergeben werde müssen. Der Rückgabewert ist entweder 0 oder 1, wobei 0 einer Defektion entspricht, während 1 Kooperation bedeutet.

Aus Platzgründen wird das wesentlich umfangreichere Programm für die populationsdynamische Simulation (ebenso wie der Programmcode für die Benutzschnittstelle) hier nicht mehr mit abgedruckt.

6.1 Tournament.py

```
"""Classes for matches and tournaments.

The the instances of the classes Match and Tournament can be
connected to visualizing objects by appending a signal function to
the signalXXXXXX lists of the class instance. Signal functions have
the general form:

    function(originating_object, *argument_list)

See the source code to find out which signals are defined and which
arguments are passed to the signals.

"""



############################################################
#
#  class Match
#
############################################################
```

```
class Match:
    """Let two strategies play one series of iterated prisoners
    dilemmas and record the results.

    signals:
        signalROUND_FINISHED (match)   - sent when one round of the match
                                         has been finished

    flags (read the "state" field, e.g.  m = Match(p1, p2); flag=m.state):
        MATCH_READY             - match has not yet started
        MATCH_RUNNING           - match is currently running
        MATCH_FINISHED          - match has been finished
    """


    def __init__(self, player1, player2, T=5, R=3, P=1, S=0, w=200):
        """Initialize class with player strategies."""

        self.signalROUND_FINISHED = []  # dummy function with
                                        # argument list

        self.player1 = player1          # strategy of player 1
        self.player2 = player2          # strategy of player 2
        self.moves1 = []                # sequence of moves from player 1
        self.moves2 = []                # sequence of moves from player 2
        self.score1 = 0                 # absolute score of player 1
        self.score2 = 0                 # absolute score of player 2
        self.round = 0                  # number of round
        self.state = "MATCH_READY"

        self.T = T  # reward for defecting if the other player cooperated
        self.R = R  # reward for mutual cooperation
        self.P = P  # "reward" for mutual defection
        self.S = S  # "reward" for cooperation if the other player defected
        self.w = w  # number of rounds


    def score(self, m1, m2):
        """Return the score of player one for the moves m1 and m2 of
        player one and player two respectively. In order to determine the
        score of player two just swap the parameters m1 and m2."""

        if (m1, m2) == (0, 1):      return self.T
        elif (m1, m2) == (1, 1):    return self.R
        elif (m1, m2) == (0, 0):    return self.P
        else:                       return self.S


    def nextRound(self):
        """Let the players meet and update the respective variables with
        the results. Then, determine if iteration stops after this round
        (random > w). Return 0 if it does, 1 otherwise."""

        self.round += 1

        m1 = self.player1.nextMove(self.round, self.moves1, self.moves2)
        m2 = self.player2.nextMove(self.round, self.moves2, self.moves1)

        self.moves1.append(m1)
        self.moves2.append(m2)
```

```
        self.score1 += self.score(m1, m2)
        self.score2 += self.score(m2, m1)

        if self.round >= self.w:    return 0
        else:                       return 1


    def match(self):
        """Play one match. Call notifier function after each round."""

        self.state = "MATCH_RUNNING"

        cont = 1
        while cont:
            cont = self.nextRound()
            for s in self.signalROUND_FINISHED: s(self) # callback

        self.state = "MATCH_FINISHED"



############################################################
#
#  class Tournament
#
############################################################


class Tournament:
    """Set up and play a tournament.

    signals:
        signalENLIST_PLAYER (tournament, player)
            a new player has been enlisted
        signalNEW_MATCH (tournament, match)
            a new match has just begun
        signalMATCH_COMPLETED (tournament, match)
            match "match" has been completed
        signalTOURNAMENT_FINISHED (tournament)
            the tournament has been finished

    flags (read the "state" field, e.g.  t = Tournament(); flag=t.state):
        TOURNAMENT_READY        - match has not yet started
        TOURNAMENT_RUNNING      - match is currently running
        TOURNAMENT_FINISHED     - match is finished
    """


    def __init__(self, name = "Tournament", T=5, R=3, P=1, S=0, w=200):
        """Initialize class with match notifier and
        identification string."""

        self.signalENLIST_PLAYER       = []
        self.signalNEW_MATCH           = []
        self.signalMATCH_COMPLETED     = []
        self.signalTOURNAMENT_FINISHED = []

        self.name = name    # identification string
        self.player = []    # list of players
        self.match  = []    # list of matches
```

```
        self.score = {} # dictionary of player scores (indexed by players)
        self.state = "TOURNAMENT_READY"

        self.T = T  # Parameters defining the reward (or punishment resp.)
        self.R = R  # for cooperation or defection.
        self.P = P  # These parameters are directly passed to Match.__init__()
        self.S = S
        self.w = w  # number of rounds



    def __cmpfunc(self, p1, p2):
        if self.score[p1] > self.score[p2]:     return -1
        elif self.score[p1] == self.score[p2]:  return 0
        else:                                   return 1


    def enlistPlayer(self, newPlayer):
        """Adds a new Player to the tournament."""

        self.player.append(newPlayer)
        for s in self.signalENLIST_PLAYER: s(self, newPlayer)


    def runTournament(self):
        """Run the tournament.

        Let each player play once against every other player.
        Invoke signalNEW_MATCH before a new new match is started and
        signalMATCH_COMPLETED after it is finished. When all matches are
        completed, calculate the scores of the players and sort
        self.player according to the players rank in the tournament."""

        self.state = "TOURNAMENT_RUNNING"

        self.match = []
        for p in self.player: self.score[p] = 0

        for i in range(len(self.player)):
            for k in range(i+1, len(self.player)):
                p1 = self.player[i]
                p2 = self.player[k]

                m = Match(p1, p2, self.T, self.R, self.P, self.S, self.w)

                self.match.append(m)
                for s in self.signalNEW_MATCH: s(self, m)

                m.match()
                self.score[p1] += m.score1
                self.score[p2] += m.score2
                for s in self.signalMATCH_COMPLETED: s(self, m)

        self.player.sort(self.__cmpfunc)

        self.state = "TOURNAMENT_FINISHED"
        for s in self.signalTOURNAMENT_FINISHED: s(self)
```

6.2 Strategy.py

```
"""Some basic strategies."""


import whrandom


class Strategy:
    """An abstract class for a player strategy.

    Any concrete player strategy in the tournament is a child
    class of this one."""

    name = "no strategy"


    def nextMove(self, round, myMoves, opMoves):
        """Determine the next move (either 1 to coperate or 0 to defect)
        based on the sequences of all previous moves.

        Parameters:
        round     - Number of the current round starting with 1
        myMoves   - List of all previous moves of this strategy in this match
        opMoves   - List of all previous moves of the opponent strategy
        """
        pass



###########################################################
#
#   Trivial strategies
#
###########################################################


class AlwaysFriendly(Strategy):
    """This strategy never defects."""

    name = "Always friendly"

    def nextMove(self, round, myMoves, opMoves):
        return 1                                        # never defect



class UtterlyDestructive(Strategy):
    """This strategy always defects."""

    name = "Utterly destructive"

    def nextMove(self, round, myMoves, opMoves):
        return 0                                        # always defect



class Random(Strategy):
    """This strategy chooses its moves at random."""
```

```
    name = "Random"

    def nextMove(self, round, myMoves, opMoves):
        return whrandom.randint(0, 1)                   # play randomly



############################################################
#
#   simple strategies
#
############################################################


class TitForTat(Strategy):
    """Play cooperatively only if the other player did
    so in the previous round. Start friendly."""

    name = "Tit for tat"

    def nextMove(self, round, myMoves, opMoves):
        if round == 1:
            return 1                                    # start friendly
        else:
            if opMoves[-1] == 1: return 1
            else:                return 0



class TitForTwoTats(Strategy):
    """Play friendly if the oponent did not defect in the
    two previous rounds. Start friendly."""

    name = "Tit for two tats"

    def nextMove(self, round, myMoves, opMoves):
        if round <=  2:
            return 1                                    # start friendly
        else:
            if (opMoves[-2:] == [0,0]): return 0
            else:                       return 1



class MassiveResponse(Strategy):
    """Play 'Tit for Tat' but punish twice for every single defection of
    the opponent. Start friendly though."""

    name = "MassiveResponse"

    def nextMove(self, round, myMoves, opMoves):
        if round == 1:
            return 1                                    # start friendly
        elif round == 2:
            if opMoves[-1] == 0:      return 0          # Tit for Tat
            else:                     return 1
        else:
            if opMoves[-2:] != [1,1]: return 0          # Massive response
            else:                     return 1
```

```
class Cheater(Strategy):
    """Play friendly if the oponent did not defect in the two
    previous rounds (like 'Tit for two Tats'). But try to cheat
    (play destrcutive) every 7th round."""

    name = "Cheater"

    def nextMove(self, round, myMoves, opMoves):
        if round <= 2:
            return 1                                    # start friendly
        else:
            if (round % 7) != 0:                        # play Tf2T usually
                if (opMoves[-2:] == [0,0]): return 0
                else:                       return 1
            else:                                       # but cheat sometimes
                return 0



class GraciousTfT(Strategy):
    """Play 'Tit for Tat', but play cooperatively (as an offer of peace),
    if there have already been five rounds of mutual defection (or
    alternating defection and cooperation) in sequence."""

    name = "GraciousTfT"

    def nextMove(self, round, myMoves, opMoves):
        if round == 1:
            return 1                                      # start friendly
        elif round > 6 and ((opMoves[-5:] == [0,0,0,0,0] and \
                             myMoves[-5:] == [0,0,0,0,0]) or \
                            (opMoves[-5:] == [0,1,0,1,0] and \
                             myMoves[-5:] == [1,0,1,0,1])):
            return 1                                      # peace offer
        else:
            if opMoves[-1] == 1: return 1                 # play tit for tat
            else:                return 0




class MaliciousTfT(Strategy):
    """Play 'Tit for Tat' but start unfriendly."""

    name = "MaliciousTfT"

    def nextMove(self, round, myMoves, opMoves):
        if round == 1:
            return 0                                    # start unfriendly
        else:
            if opMoves[-1] == 1: return 1
            else:                return 0



class DelayedTfT(Strategy):
    """Play friendly if the opponent did so three moves before, otherwise
    do not cooperate. Play friendly at the beginning."""

    name = "DelayedTfT"
```

```
    def nextMove(self, round, myMoves, opMoves):
        if round <= 3:
            return 1
        else:
            if opMoves[-3] == 1: return 1
            else:                return 0



############################################################
#
#   slightly more comples strategies
#
############################################################


class Tester(Strategy):
    """Defect in the first round in order to test
    the opponents reaction.  Based on the opponents reaction, play either
    Tit for Tat (starting friendly) or try to deceive opponent by playing
    cooperatively in the second and third round and then defecting every
    second round. (See Axelrod, ch. 2, p.40)"""

    name = "Tester"

    def nextMove(self, round, myMoves, opMoves):
        if round <= 2:
            return 0
        elif round == 3:
            if opMoves[-1] == 0:  self.state = "TFT"
            else:                 self.state = "Deceiver"
            return 1
        elif round == 4:
            return 1
        else:
            if self.state == "TFT":
                if opMoves[-1] == 1:  return 1
                else:                 return 0
            else:
                if round % 2 == 1:    return 0
                else:                 return 1



class Analyst(Strategy):
    """Play random for the first ten rounds. Then try to analyse
    the opponents strategy based on the opponents reactions. If either
    the opponent could be exploited very  well or if the opponent
    did attempt to exploit this strategy to often, play non-cooperatively.
    Play cooperativly, if it wasn't possible to exploit the opponent
    and if the opponent played fair as well."""

    name = "Analyst"

    def nextMove(self, round, myMoves, opMoves):
        if round <= 10:
            return whrandom.randint(0, 1)   # play random at the beginning
        else:

            # analyse
```

```
ex_attempt, ex_success = 0,0
opex_opportunity, opex_attempt = 0, 0

i = -9
while i <= -1:
    if myMoves[i-1] == 0:
        ex_attempt += 1
        if opMoves[i] != 0: ex_success += 1 # opponent did
                                            # not punish exploit!
    else:
        opex_opportunity += 1
        if opMoves[i] == 0: opex_attempt += 1  # opponnent played
                                               # defective without
                                               # reason
    i += 1

# and react accordingly

ret = -1
if (ex_attempt > 0):
    if (float(ex_success) / float(ex_attempt)) >= 0.6:
        return 0                    # keep exploiting
    else: ret = 1                   # try to be friendly again

if opex_opportunity > 0:
    if (float(opex_attempt) / float(opex_opportunity)) <= 0.4:
        return 1            # opponent isn't really bad
    else:
        return 0            # opponent tried to deceive to often
else:
    if ret != -1: return ret      # fallback
    else:
        if opMoves[-1] == 1: return 1   # play TfT if clueless
        else:                return 0
```